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

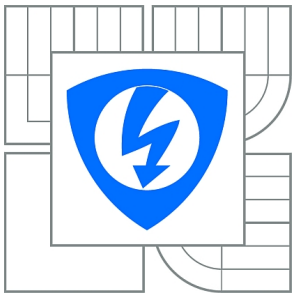
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE DAVID TROJÁK
AUTHOR

BRNO 2012

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

EXTRACTING TEXT DATA FROM THE WEBPAGES

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

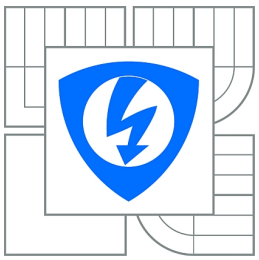
AUTOR PRÁCE — DAVID TROJÁK

AUTHOR

VEDOUCÍ PRÁCE — Ing. RADEK ČERVENEC

SUPERVISOR

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	David Troják	***ID:***	116992
Ročník:	3	***Akademický rok:***	2011/2012

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 navrhněte koncept algoritmu pro extrakci 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
[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: www2009.org/proceedings/pdf/p1169.pdf
[3] ATKINSON, J.A. Institute for Communicating and Collaborative Systems [online]. 2003. Text Mining: Principles and applications. Dostupné z URL: http://labs.rightnow.com/colloquium/papers.php.

Termín zadání:	6.2.2012	***Termín odevzdání:***	31.5.2012

Vedoucí práce: Ing. Radek Červenec
Konzultanti bakalářské práce:

prof. Ing. Kamil Vrba, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Tato práce se zabývá dolováním textových údajů z webových stránek, přehledem možných programů a způsoby extrakce textu. 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, získání textových dat z webu, problémy při extrakci dat, wrappery, Java program.

ABSTRACT

This work deals with text mining from web pages, an overview of available programs and its methods of text extraction. Part of this work is the program created in Java language, which allows text to obtain data from specific web pages and save them into XML file.

KEYWORDS

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

TROJÁK, David *Databáze textových dat z internetových stránek*: bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2012. 47 s. Vedoucí práce byl Ing. Radek Červenec

PROHLÁŠENÍ

Prohlašuji, že svou bakalářskou práci na téma „Databáze textových dat z internetových stránek“ jsem vypracoval samostatně pod vedením vedoucího bakalářské 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 bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské 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

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

ÚVOD

V dnešním světě je Internet největší síť, která je hojně využívána pro vyhledávání a sdílení. Většina textů a informací má svou elektronickou podobu, která je prostřednictvím sítě Internet distribuována. Jsou to nejrůznější zpravodajské servery,ale i knižní nakladatelství, která vydávají vedle tištěné i elektronickou podobu téhož textu.

Uživatelé pro nalezení informací v drtivé většině používají různé internetové vyhledávače, které jsou většinou omezeny indexací (oblast stránek, kterou vyhledávač dříve navštívil nebo byla přidána autorem) a klíčovými slovy. Kromě výsledků zobrazuje např. reklamy a jiné nedůležité věci. Výsledky jsou navíc řazeny podle relevance klíčového slova a hodnocení stránek vyhledávačem tzv. ranku. V dnešní době vyhledávače mají stále několik problémů. Ve většině případů vyhledávač nabízí spousty výsledků hledání, ale najít zde relevantní informaci je mnohdy složité a časově náročné. Dalším problémem je nemožnost indexovat veškeré množství dat na internetu, čímž může dojít ke ztrátě možných relevantních informací. Z tohoto důvodu stoupá obliba systémů pro automatické dolování užitečných znalostí z textu (Text Mining).

Tyto systémy získávají své informace v mnoha formátech a z různých zdrojů. Je tedy žádoucí, aby získané informace byly uloženy strukturovaně a byly obohaceny o důležité informace, jako je zdroj, význam atd. Další systémy mohou díky jednotné struktuře dále informace zpracovávat, učící algoritmy se na nich mohou dále zdokonalovat apod. Proto byl prostřednictvím této bakalářské práce započat návrh zmíněného systému.

Cílem této bakalářské práce je prostudování stávajících systémů, implementování nového systému pro automatické dolování textu z Internetu a ukládání ve strukturované podobě. Očekává se vznik rozsáhlé databáze obsahující kromě textu v jednotném formátu i přidružené informace.

1 TEORETICKÁ ČÁST

1.1 Data Mining

Data Mining neboli dolování z dat je definováno jako „netriviální extrakce implicitních, dříve neznámých a potenciálně užitečných informací z dat“.[1] Informace je implicitní tedy skrytá a neznámá. Hlavní vlastností je analýza vycházející z obsahu dat, nikoli od uživatele. Pro získání musí být použito automatických technik pro dolování dat. Jedná se o shromáždění dat a jejich následné vyhodnocení podle požadovaného klíče.

Techniky dolování dat mohou pomoci při řešení problému „zahlcení informacemi“. Výzkum dolování zasahuje a překrývá se s oblastmi jako jsou databáze, získávání informací, extrakce informací a zpracování přirozeného jazyka.

Dolování dat se skládá z pěti hlavních částí:

- Extrakt, transformace a načítání dat transakcí do systému databáze.
- Ukládání a správa dat v databázi systému.
- Analýza dat aplikací softwaru.
- Prezentace dat v požadovaném formátu, například graf nebo tabulka.

Data Mining vychází z velké řady matematických a statistických technik. Nyní si uvedeme některé z hlavních technik:

- **Rozhodovací stromy** – Prediktivní model, který data prezentuje v podobě stromu, kde každý uzel náleží jedné vlastnosti nebo kritériu. Z tohoto uzlu poté vychází konečný počet nových uzlů. Data jsou tedy pomocí stromu rozdělena na určité segmenty, které se vyznačují stejnými vlastnostmi.
- **Neuronové sítě** – Jsou založeny na obdobné logice, jakou má organizace nebo způsob chování lidského mozku respektive neuronů. Využívají se hlavně v oblasti umělé inteligence. Hlavní síla Neuronové sítě spočívá v paralelním zpracovaní dat. Síť se skládá z umělých neuronů, které jsou vzájemně propojeny a mohou si tak předávat signály. Neuron může mít libovolný počet vstupů, ale pouze jeden výstup.
- **Genetické algoritmy** – Jedná se o simulaci evolučního vývoje. Aplikují se zde procesy jako dědičnost, mutace, křížení nebo přirozený výběr, díky kterým můžeme určit vývoj či modifikaci atributů.
- **Shlukování a klasifikace** – Technika shlukování slouží k roztřídění dat do skupin s podobným obsahem. Pomocí klasifikace nalezneme hlavní atributy skupin. Tato metoda používající dvě techniky nám umožňuje identifikovat a charakterizovat odlišné segmenty z dat.[1]

1.1.1 Vývoj dolování dat

Jedna z počátečních forem dolování dat se nazývala Data Dredging neboli "bagrování z dat". Z názvu můžeme odvodit, že se jednalo o velice primitivní prohledávání dat bez vytvoření počáteční hypotézy. I přes tuto skutečnost je tato forma oblíbená, protože dopomohla k objevení velmi hodnotných informací.

Postupem času se začala zajímat o dolování z dat odvětví jako jsou pojišťovnictví, telekomunikace, veřejné služby, bankovnictví, maloobchod, lékařství, vědecké výzkumy a mnoho dalších. Manažerům Data Dredging přináší informace, díky kterým se mohou zaměřit na správné faktory podnikání či objevit skryté korelace mezi ekonomickými proměnnými v neustále se zrychlujícím tempu obchodu. Jedním z hlavních lákadel k investicím jsou speciální algoritmy, které automaticky hledají v datech strategické informace. Jedná se tedy o analytickou techniku zpracování, která je přímo závislá na skladech dat.[1]

1.1.2 Typy dat

Systémy dolování dat mohou zpracovávat různé data. Mohou to být typy dat ve strukturované formě, v semi-strukturované formě a nebo v úplně nestrukturované formě.

Strukturovaná data

Data se strukturou nalezneme například u XML dokumentů s přiloženým DTD nebo s XML schématem.

Semi-strukturovaná data

Pod tímto pojmem můžeme chápat třeba HTML stránky. Zpracovávání takovýchto dat není založeno na NLP (přirozené zpracovaní jazyka), ale vzniká pomocí vzorů vytvořených například posloupností HTML (HyperText Markup Language) tagů.

Nestrukturovaná data

Jedná se o čistá textová data, která nemají žádnou strukturu. Extrakce takovýchto dat spadá pod techniky NLP(přirozené zpracování jazyka).

1.2 Text Mining

Dolování textu je důležité pro získání potřebných údajů. Jejich využití je různorodé. Mohou sloužit podnikům k efektivnějšímu hospodaření, tak podnikatelům jako podklady pro rozhodování. Dolování textu obvykle obsahuje procesy pro strukturovanost

vstupního textu (jako třeba rozložení, přidání či odebrání lingvistických funkcí nebo konečné uložení do příslušné selektované databáze), následné vyhodnocení a prezentaci na výstupu[8]. V mnoha aplikacích dolování textu jsou údaje brány jako potenciálně užitečné, ale nevedou většinou k pravému účelu dolování. To je nutné především v případě, když se jedná o výsledky určené k lidskému užití, nikoli jako základ pro další automatické operace. Výsledek se srozumitelným výstupem je základ pro sumarizaci obsáhlých textů.[14]

1.2.1 Vývoj dolování textu

Dříve bylo dolování informací z textu velice složité, protože text byl psaný a vyhledávání v něm časově náročné. Později se proto zavedly rejstříky a obsahy, které toto vyhledávání dosti usnadnily, ale stále bylo potřeba velké množství času pro nalezení potřebných údajů.

Až s rozmachem počítačové technologie přišla revoluce v dolování textu. Všechen text se přepsal do elektronické podoby a dostal se do globální sítě, kam má dnes přístup téměř každý. Dále se vyvíjely nejrůznější nástroje pro usnadnění vyhledávání správných údajů.

Různé firmy daly na vývoj těchto technologií dost peněz, aby mohly analyzovat své data. V dnešní době je trend vytvořit programy, které dokáží vyhledávat důležitá data i v cizích jazycích.

Jednou z firem byla například DARPA (Defense Advanced Research Projects Agency), která financovala MUC (Message Understanding Conferences). Jednalo se o soutěž zaměřenou na vývoj nových metod získávání informací. Na každou konferenci bylo vypsáno zadání a vývojové skupiny jej musely co nejpřesněji splnit. Poté skupiny s nejlepšími řešeními byly vybrány pro účast na konferenci.

1.2.2 Metody dolování

Problém vyhledávání správných údajů a informačního zahlcení se pokouší řešit sémantický web. Ten oproti klasickým internetovým stránkám, u nichž je nejdůležitější dobrá čitelnost a srozumitelnost pro uživatele, poskytuje srozumitelná data pro strojové zpracování, tzn. dává údajům strukturu a provazuje je. Základní technologie pro sémantické internetové stránky je RDF (Resource Description Framework), která dokáže popisovat zdroje pomocí tzn. trojic (zdroj, vlastnost, hodnota). Tyto trojice bývají označovány jako „tvrzení“. V rámci každého tvrzení je zdroj subjektem, vlastnost predikátem a hodnota vlastnosti objektem. Hodnota vlastnosti může nabývat opět dalšího zdroje. RDF data neboli obsah stránek je možné vytvořit přímo

na serveru pomocí již vytvořených databází. Na takové úkony jsou již předpřipraveny příslušné aplikace, tudíž většina webu může být lehce převedena i do strojové verze.[21, 3]

Metody dolování textu:

- **Kategorizace** – Jedná se o začlenění podobných dokumentů do předem předdefinovaných tříd.
- **Shlukování** – Dokumenty, které jsou si dosti podobné, se bez testování dávají do shluků.
- **Sumarizace** – Obsáhlý text je zestručněn na pár vět.
- **Extrakce** – Získávání požadovaných informací z textu (emaily, ceny aj.).
- **Selekce informací** – Hledávání klíčového slova ve více dokumentech.
- **Vizualizace** – Grafické zobrazení požadovaných dat.

Souhrn metod představuje hlavní nástroj pro analýzu textových informací. Pokud metody vhodně použijeme, mohou nám usnadnit práci. V dnešním světě s obrovským množstvím dat je časová náročnost zpracování velká, a to ve většině případů není žádoucí.

Kategorizace

Problém kategorizace je složitý a místy i neřešitelný. Na začátku je potřeba si vytvořit množiny trénovacích dat obsahující příklady správného přiřazení. Následně pomocí těchto množin už je možné nová data kategorizovat. Kategorizování dokumentů je dlouhodobě využívanou technikou pro vyhledávání informací v knihovnách.

Shlukování

Tento název v sobě zahrnuje velké množství výpočetních postupů. Cílem je rozklad daného dokumentu na relativně stejné shluky. Princip spočívá v tom, že jednotlivé shluky obsahují objekty, které si jsou co nejvíce podobné a objekty náležící do rozdílných shluků se musí co nejvíce lišit. Každý objekt je popsán skupinou proměnných (znaků). Shlukovací analýza závisí na zvolených proměnných, míře vzdálenosti mezi objekty a shluky a na algoritmu výpočtu. Důležitá část analýzy je popis jednotlivých shluků. Největší úspěch efektivity lze dosáhnout, pokud objekty mají tendenci se shlukovat do přirozených tříd.

Sumarizace

Pokud uživatel potřebuje v krátkém čase pochopit obsah textu, je dobré využít metodu automatické sumarizace dokumentu. Princip spočívá ve vytvoření souhrnu důležitých částí. Máme 2 hlavní typy:

- **Sumarizace extrakcí** – Souhrn je vytvořen z původního textu pomocí statistických metod, heuristických metod nebo pomocí obou. Části, které byly vyjmuty z textu, již nejsou dále upravovány.
- **Obsahová sumarizace** – Souhrn je výkladem původního textu. Princip spočívá v nahrazování příliš dlouhých částí jejich kratšími verzemi. Například věta „Navštívil jsem Prahu, Brno a Olomouc.“ Bude nahrazena verzí „Navštívil jsem města.“

Extrakce

Metoda získává data z dokumentu na základě pravidel, která jsou popsána na trénovacích stránkách. Zde je přesně definováno, co se má extrahovat. Například u HTML kódu se používá extrakce obsahu v hranatých závorkách (%požadovaný text%). Nejvíce se tedy tato metoda používá pro značkovací jazyky jako jsou například HTML a XML (Extensible Markup Language). Nevýhoda této metody spočívá v nepřizpůsobivosti. Při změně systému stránky přestává systém fungovat a vyžaduje zásah programátora, aby ho opět doladil.[7]

Selekce informací

Jde o vyhledávací způsob většiny vyhledávačů jako například Seznam.cz, Google atd. Uživatel přes formulář zadá klíčová slova a obdrží seznam dokumentů, které nejvíce vyhovují výchozím parametrům. Bohužel samotné nalezení požadovaných dokumentů nestačí k úspěšnému vyhledávání. Vzniká zde pojem relevance, která je definována jako vlastnost vztahu mezi dotazem uživatele a jednotlivým dokumentem, tedy jako prvek množiny všech nalezených dokumentů. Efektivnost vyhledávání je hodnocena atributy přesností a úplností, které jsou získány na základě relevance výsledků.

Vizualizace

Tato metoda se používá jako doplňková pro metody předchozí. Zejména u shlukování a kategorizace. U shluků ve 2D modelu můžeme ohodnotit, jak správně shluky vznikly, a u kategorizace můžeme zobrazit rozhodovací stromy.

1.2.3 Aplikace jednotlivých metod

Při rozhodování, kterou metodu použít, se nemusíme omezit na jednotlivé metody, ale můžeme využít také jejich kombinací. Například kombinace selekce informací a textové kategorizace. Při dobré volbě procedury je možné celkové vyhledávání mnohokráte urychlit.

1.2.4 Hodnocení výsledků

Abychom mohli hodnotit různé systémy je nezbytné zavést standardizovaná kritéria hodnocení výsledků. Dnes se využívají dvě kritéria, a to již zmíněné atributy přesnost a úplnost.

- Přesnost - je definována jako poměr počtu získaných relevantních dokumentů a počtu všech relevantních dokumentů.
- Úplnost - je definována jako poměr počtu získaných relevantních dokumentů a počtu všech získaných dokumentů.

Dosažení vysoké hodnoty obou parametrů je značně obtížné. Při vysoké úplnosti klesá šance, že budou nalezeny všechny vyhovující dokumenty. Naopak při vysoké úspěšnosti hrozí, že nehledaný dokument bude vyhodnocen jako hledaný.

1.2.5 Vlastnosti textu

Při automatickém zpracování textu vznikl model vlastností textu. Jedná se o ohodnocení postavení textu oproti celé stránce. Pro tyto účely byly zavedeny atributy zřetelnost (markedness) a váha (weight). Zřetelnost nám udává, jak je text vůči ostatnímu výrazný. Váha zase bere postavení textu v hierarchii nadpisů.

- zřetelnost $= (F \cdot \Delta f + b + o + u + c) \cdot (1 - z)$
- váha $= [(F \cdot \Delta f) + (b + o + u + c) \cdot l + W \cdot p] \cdot (1 - z)$

Pokud je text tučným písmem, kurzívou, podtržený, barevně odlišený nebo přeškrtnutý, tak hodnoty b, o, u, c a z jsou rovny 1. V opačném případě jsou rovny 0. Parametr Δf je roven hodnotě rozdílu mezi velikosti písma textu a standardní velikostí v dokumentu. Proměnné l a p mají hodnotu 1, pokud za textem následuje zalomení řádku nebo dvojtečka. Konstanty F a W definují váhu velikosti písma a dvojtečky.

1.3 Wrappery

Takto nazvaná skupina obsahuje programy, které se zabývají čistě automatickou extrakcí dat z webových stránek. Jejich 3 základní úkoly jsou jasné:

- Stáhnutí HTML stránky z Internetu,
- vyhledávání a rozpoznávání požadovaných dat,
- strukturované uložení do formátu dat pro další možné použití.

Většinou se pro každý web musí vytvořit individuální wrapper, který je schopný na podobných stránkách tohoto webu provádět svou činnost. Jedná se třeba o různé e-shopy, kde se stránky generují podle určité šablony. Nevýhoda tedy spočívá ve statickém řešení dolování textu a případné problémy musí být vyřešeny opětovným

vytvořením wrapperu[12]. Hlavními vlastnostmi wrapperů jsou typy vstupních stránek, hloubka extrakce, závislost na HTML aj. Wrappery se dělí do 6 základních tříd:

- **LR** (left-right) — Pro všechny extrahované hodnoty existuje pravidlo, které se skládá z dvojice řetězců. První odděluje hodnotu od zbývajícího textu zleva a druhý zprava. Aplikace prochází celý dokument a aplikuje na hodnoty zvolené pravidlo.
- **HLRT** (head-left-right-tail) — Tato třída je modifikací předchozí třídy LR. Používá stejný algoritmus, ale omezuje se na určitou část dokumentu. Využívá se zejména pro rozsáhlé dokumenty, aby byla zvýšena efektivnost.
- **OCLR** (open-close-left-right) — Opět třída, která modifikuje třídu LR. Oddělovače *open* a *close* oproti vymezení oblasti u předchozí třídy určují specifickou extrakci pro jednotlivé n-tice.
- **HOCLRT** (head-open-close-left-right-tail) — Jedná se o třídu, která kombinuje třídy **HLRT** a **OCLR**. Extrakce probíhá ve vymezené části dokumentu a vynechává místa ležící mimo oddělovače *open* a *close*.
- **N-LR** (nested-left-right) — Čtyři předchozí třídy wrapperů řeší pouze opakující se strukturu. Příkladem jsou např. tabulky. Tato třída vyhodnocuje data, která mohou být vnořena do sebe. Typickým příkladem mohou být víceúrovňové seznamy.
- **N-HLRT** (head-left-right-tail) -– Třída kombinující třídy **N-LR** a **HLRT**.[10]

1.4 Yahoo! Query Language

(http://developer.yahoo.com/yql/)

Firma Yahoo vyvinula pro správce webů aplikaci na bázi dotazovacího jazyka SQL, která dokáže vyhledávat a filtrovat data. Mimo jiné můžeme službu využít pro stažení obsahu HTML/XML dat. Samozřejmě se program netýká pouze Yahoo stránek, ale pomocí technologie Open Data Tables je možné si nadefinovat propojení s jiným poskytovatelem dat. Služba také nabízí klasické SQL příkazy jako INSERT,UPDATE a DELETE. Příkazy pracují s tabulkami, kde mohou vkládat, měnit a mazat záznamy, ale pouze po autorizaci pomocí protokolu OAuth.

1.5 Struktura HTML stránek

Mezi základní HTML tagy, které vymezují oblast souboru patří HTML, HEAD a BODY.

- **HTML** – Vymezuje konec a začátek každého dokumentu. Je to sice dáno normou, ale v dnešní době si prohlížeč tento tag domyslí a není nutné ho používat.
- **HEAD** – Tzv. „hlavička“ dokumentu, která se nezobrazuje. Obsahuje tagy jako TITLE, STYLE, SCRIPT aj.
- **BODY** – Neboli tělo celé stránky obsahuje vizuální část HTML stránky. Tento tag má své atributy jako třeba barva pozadí, barva textu nebo odkazů celého dokumentu aj., které jsou v dnešní době zastaralé a nahrazují se pomocí CSS stylu. V těle dokumentu se nalézá text, který nás zajímá, a tagy jako font, odkaz, obrázek aj.[9]

Další tag, který se může objevit v souboru je komentář, který je označen začátkem $<!--$a koncem $-->$. Tyto poznámky se nezobrazují návštěvníkům stránky a slouží spíše pro autory stránek, aby se v nich vyznali a orientovali.[13]

1.6 RSS kanály

RSS kanály představují velice populární odnož XML formátu pro čtení zpravodajských i jiných novinek z webu, případně pro čtení dalších informací. Tato technologie umí stručně informovat uživatele Internetu o novinkách v dané oblasti, za předpokladu předchozí registrace vyžadovaných informací. V principu se celková zpráva zmenší na nejdůležitější sdělení, které uživatel potřebuje vědět, a pokud potřebuje více, tak je nutná návštěva příslušného serveru, který sdílí informaci v celku. Jednotlivé RSS (Really Simple Syndication) zprávy obsahují jako hlavní atributy: *description* neboli popis dané zprávy, *title* neboli titulek, *link* neboli odkaz na celou zprávu na serveru a časovou značku ve standardizovaném formátu. Co se týká zpravodajských serverů¨, je zde navíc přidáván obrázek aktuality nebo další metadata. Ukázka RSS kanálu verze 2.0 je možné nalézt v příloze A.1.4.

Samotný princip distribuce aktualit dříve předcházelo používaní *newsletterů* tedy emailů s novinkami. Tato metoda však měla značný problém s rostoucí nevyžádanou poštou (SPAM) s používaním antispamového filtru, kdy se stávalo, že tyto emaily byly chybně označeny za SPAM. Další rozdíl oproti RSS kanálům byla jejich statičnost. U velmi frekventovaných serverů s aktualitami by se dalo říci, že už při odeslání byl email neaktuální. V neposlední řadě patří k výčtu záporných vlastností také fakt, že rozesílání emailů velmi zatěžovalo síťové prostředky.

1.6.1 Vývoj RSS

Původně se jednalo o určité aktualizace mezi jednotlivými servery, aby mohly odkazovat na aktuální události druhého serveru. Již hodně dávno bylo běžné, že se

objevovaly odkazy na různé servery s aktuálními novinkami. Původním cílem tohoto formátu bylo jednoduché a lehce pochopitelné zpracování informací pro širokou veřejnost, který vyvinula firma Netscape. Tato velice silná myšlenka se nesetkala v počátcích s velkým ohlasem, ale postupný rozvoj webů, blogů a celkově Internetu prokázal nutnost sumarizace a selekce informací do stručných zpráv. Velmi průlomový okamžik nastal, když jako první zařadil deník New York Times do své online verze RSS kanál.

První verze s označením *RDF Site Summary* (RSS) 0.9 vznikla již v roce 1999. V brzké době následovala verze 0.91, která se odklonila od RDF elementů, a proto bylo nutné změnit název na *Rich Site Summary*. Tato verze se stala nejčastěji používanou.

Do dalšího vývoje se vložili dvě strany a vznikly tedy dvě větve vývoje. První strana vyvinula RSS 1.0, která se vrací zpět k RDF, přičemž navíc přidává další vlastnost XML jako třeba jmenný prostor. Druhou stranou byla firma UserLand, která se držela linie vývoje. Dodržela také na rozdíl od první strany zpětnou kompatibilitu až do verze RSS 2.0, kde dochází opět ke změně názvu technologie na *Really Simple Syndication*. Jedná se o asi v současnosti nejrozšířenější verzi, která je od roku 2003 ve správě Harvardské University.

Co se samotného zpracování RSS týče, zprvu byly na rozmachu nejrůznější programy. Nejvíce se zapsal do dějin program Feedreader. Programy byly postupně zdokonalovány pro potřeby uživatelů například implementace filtrace zpráv. Tuto funkci posléze použili i tvůrci RSS kanálu, když začali už samotné kanály třídit podle kategorii.

Nyní se to opět trend obrací k původní myšlence. Vše se vrací zpátky na webové stránky a samozřejmě také do webových prohlížečů ve formě různých pluginů neboli zásuvných modulů. Hlavní podíl na této skutečnosti má fakt, že se Internet rozšířil téměř všude a pro všechny nové uživatele není jednoduché obsluhovat tento software. Proto je jednodušší vložit čtečku do prohlížeče, který je v každém počítači a který se dokáže o vše postarat sám.

1.7 XML a parsery

Extensible Markup Language neboli rozšiřitelný značkovací jazyk patří do rodiny tagových jazyků jako třeba HTML. Hlavní využití tohoto jazyka spočívá ve smyslu databáze. Pro jeho jednoduchost a účelnost je podporován v řadě programovacích jazyků a nástrojů. Spolu se stylovými jazyky je možné z formátu XML snadno konvertovat na jiný formát. Ten pak lze snáze zobrazit, případně vyextrahovat z dokumentu jen podstatné části pro zobrazení. Pro tyto účely se nejčastěji využívá

kaskádové styly (CSS) nebo mnohem účinnější XSL (eXtensible Stylesheet Language).

Jelikož XML nemá předem předdefinované značky, tvůrce může do souboru DTD (Document Type Definition) definovat vlastní značky, které posléze slouží k automatické kontrole struktury. Pokud soubor neexistuje, tak kontrola probíhá také, ale pouze v základní rovině, jako například kontrola uzavřených tagů.

Na zpracování XML pracují programy, které se nazývají parsery, a zpracování, které se nazývá parsování, probíhá postupně přes analýzu dat, kontrolu syntaxe, kontrolu s případným DTD či doplnění výchozích hodnot, až po převedení na datovou reprezentaci, která se dále zpracovává.

Rozlišujeme 2 základní typy zpracovávání XML dokumentu: proudové nebo objektové. Každý druh zpracování má své výhody a nevýhody, které jsou zachyceny v této tabulce [2] :

Vlastnost	**DOM**	**SAX**	**StAX**
Typ zpracování	Stromové	Proudové - Push	Proudové - Pull
Procesorová a paměťová náročnost	Velká	Malá	Malá
Čtení XML	Ano	Ano	Ano
Zápis XML	Ano	Ne	ANO
Změna XML	Ano	Ne	Ne
Sekvenční zpracování	Ne	Ano	Ano

Tab. 1.1: Vlastnosti parserů

1.7.1 Typ zpracování

Největší rozdíl parserů spočívá v jejich způsobu zpracování dokumentu.

Rozlišujeme zpracování :

- **Proudové** – Nazývaný také událostmi řízené zpracování. Princip spočívá v postupném čtení XML dokumentu, kdy u každého uceleného bloku je vyvolána událost. Existují dva způsoby přístupu k této metodě :
  - *Push* – Čtení XML dokumentu probíhá automaticky a parser pouze generuje různé typy událostí, na které musí být program schopný reagovat. Představitelem této metody je SAX (Simple API for XML).
  - *Pull* – Tuto metodu využívá StAX (Streaming API for XML), spočívá v žádostech programu, respektive ve vytváření událostí, které poté parser vykoná.

- **Práce se stromovou strukturou reprezentující dokument** – Pro práci je nutné celý XML soubor přečíst a vytvořit v paměti strom reprezentující jej. Parser využívající tuto metodu se nazývá DOM (Document Object Model).[2]

1.8 Java

Java je jeden z nejpoužívanějších programovacích jazyků na světě a to hlavně proto, že je snadno přenositelný na různých platformách počínaje čipovými kartami, mobilními telefony, aplikacemi pro desktopové počítače až po velké distribuované systémy, kde spolupracují počítače po celém světě.

Java byla vyvinuta firmou Sun Microsystems, ale ta ji nechala se dále vyvíjet jako open source. Velká nevýhoda tohoto jazyka spočívá v rychlosti, protože se vždy při startu programu provádí nová kompilace neboli přeložení zdrojového kódu. Další nevýhoda je paměťová náročnost, kdy se kromě samotného programu se musí spustit celé prostředí.

2 PRAKTICKÁ ČÁST

2.1 Popis a vymezení bakalářské práce

Cílem této práce je vytvoření programu, kdy na vstupu je webová stránka a na výstupu se objeví pouze holý text v XML dokumentu, který se použije jako databáze pro následné zpracování. Program ze zdrojového kódu stránky musí být schopen vyextrahovat pouhý text, a ten následně uložit podle příslušného nastavení do databáze s náležitými atributy jako je časová známka, odkaz na získaný text nebo jazyk, v jakém je text psaný. Hlavním přínosem by mělo být vytvoření více přístupů k extrakci textu. Bude se jednat o extrakci stránky podle zadaného odkazu, odposlouchávání RSS kanálů, extrakce článku z jednoho serveru, získávání textu ze sociální sítě a použití programu Rapidminer pro extrakci.

Problém nastává hlavně u různých struktur HTML stránek. Každý vývojář nebo web designér snaží se o osobitost a vkládá do svého díla sám sebe. Z toho důvodu vznikají rozdílná rozvržení a typy stránek. To má za následek problematičtější extrakci údajů z těchto stránek.

Program bude běžet ve dvou režimech :

- **Uživatelský** – Uživatel sám volí, co si přeje stáhnout a uložit.
- **Automatický** – Samostatné vlákno programu automaticky kontroluje aktuálnost RSS kanálů a v případě změny se nová zpráva uloží.

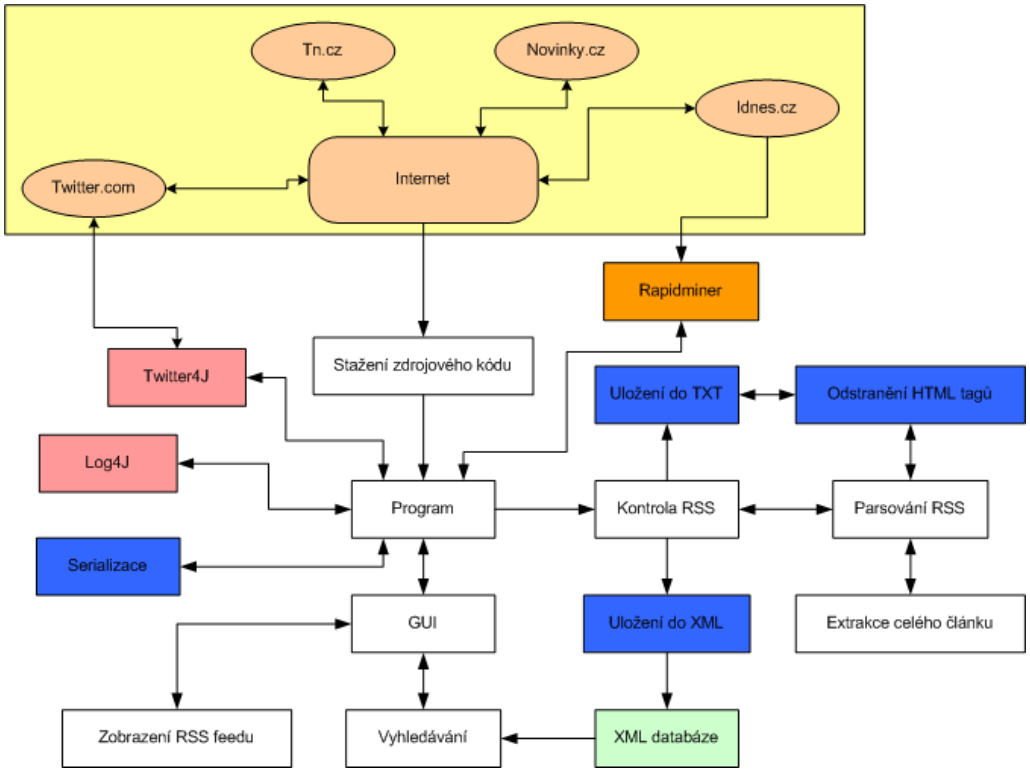
2.2 Koncepce navrženého systému

V rámci této kapitoly jsou představeny hlavní moduly (části) systému včetně jejich stručného popisu a vzájemné komunikace. Významnější bloky budou dále v práci podrobněji rozebrány.

Stručný popis blokového schématu 2.1:

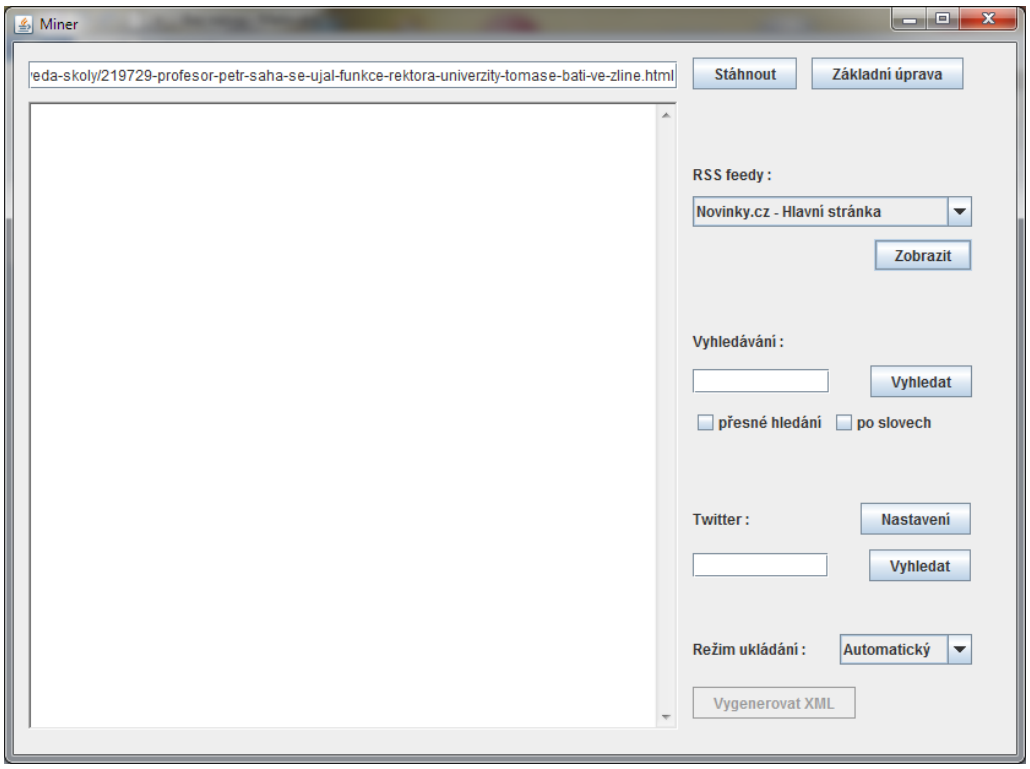
- **Stažení zdrojového kódu** – Obslužná funkce starající se o navázání spojení a stažení požadované stránky.
- **Program** – Blok představující jádro celého programu.
- **Twitter4J** – Knihovna do Javy, která zajišťuje komunikaci se serverem Twitter.com.
- **Log4J** – Knihovna do Javy umožňující jednoduché logování zpráv z aplikace.
- **Rapidminer** – Alternativní cesta získání obsahu stránky Idnes.cz pomocí programu Rapidminer.
- **Serializace** – Zálohování objektů z programu se provádí pomocí knihovny Serializace, která má v Javě přímou podporu.

- **GUI** – Označení pro grafické vlákno programu, díky kterému může uživatel komunikovat s programem.
- **Zobrazení RSS feedu** – Funkce sloužící k nalezení staženého RSS kanálu a jeho zobrazení v GUI.
- **Vyhledávání** – Pomocí klíčového slova či slov je možné vyhledávat v databázi vytvořené programem s možností vyhledávání po slovech; přesně nebo nepřesně, tedy nerozlišovat malá a velká písmena.
- **Kontrola RSS** – Blok prezentující další vlákno programu, které každou minutu kontroluje RSS kanály.
- **Uložení do TXT** – Nové zprávy z RSS kanálu se před vytvořením XML souboru uloží do textového souboru.
- **Odstranění HTML tagů** – Funkce sloužící pro odstranění nedůležitých znaků v textu jako jsou např. HTML tagy.
- **Parsování RSS** – Zpracování získané XML podoby RSS kanálu.
- **Extrakce celého článku** – Pomocí této funkce je možné získat celý článek nacházející se na serveru Novinky.cz.
- **Uložení do XML** – Blok slouží k vytvoření XML souboru z uložených dat v textovém souboru.
- **XML databáze** – Adresář obsahující XML soubory s uloženými daty.



Obr. 2.1: Blokové schéma programu

2.3 Základní uživatelské rozhraní programu



Obr. 2.2: Vzhled programu

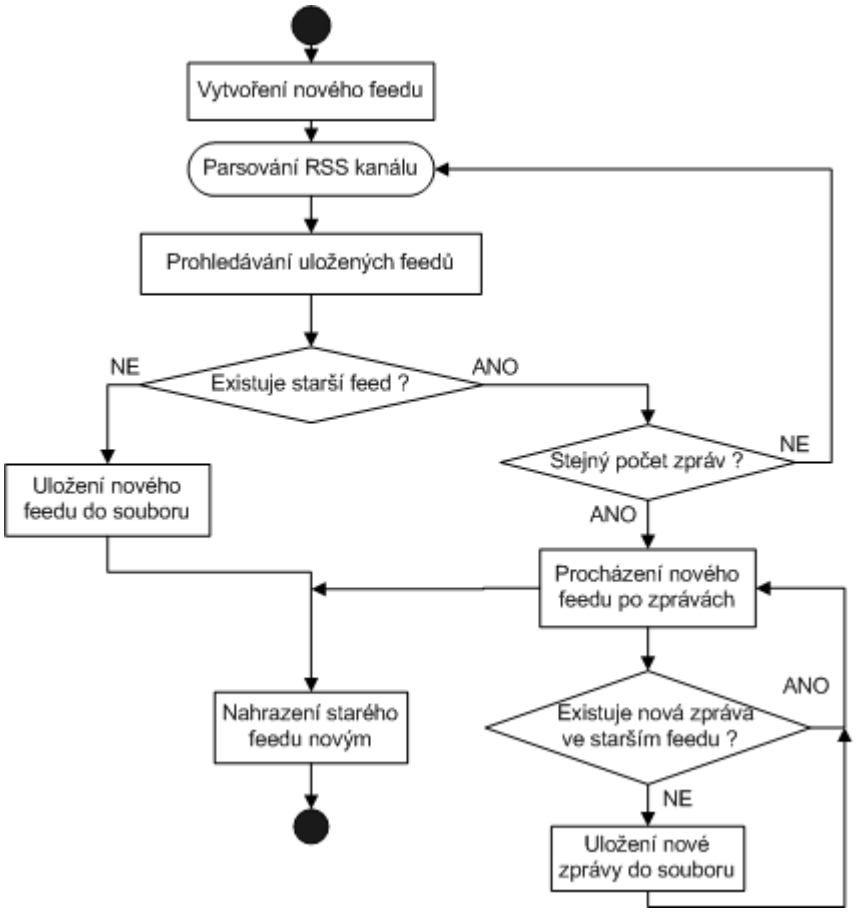
Rozhraní můžeme rozdělit do pěti oblastí:

- **Základní mód** – Zde je možnost zadat URL (Uniform Resource Locator) stránky a pomocí prvního tlačítka se stáhne zdrojový kód. Druhé tlačítko slouží k základní úpravě zdrojového kódu (odstranění HTML tagů) a uložení získaného textu.
- **RSS čtečka** – Vysunovací roletka slouží k výběru jednoho RSS feedů a jeho zobrazení.
- **Vyhledávání** – Uživatel zde může zadat klíčové slovo nebo slova a nastavit přesnost vyhledávání.
- **Twitter** – Pomocí tlačítka *Nastavení* je možné nastavit, co přesně má program hledat. Může buď hledat uživatele podle zadaného klíčového slova, nebo si může nastavit hledání klíčového slova ve zprávách. U zpráv je možné nastavení zeměpisné šířky a délky, které specifikují zprávy pro určitou oblast.
- **Režim** – Pomocí vysunovací rolety je možné nastavit dva režimy generování XML souboru. Automatický režim generuje XML soubor při přechodu na nový den. Pro ruční generování slouží tlačítko pod roletou.

2.4 Kontrola RSS kanálů

Pro kontrolu RSS kanálů bylo vytvořeno samostatné vlákno, aby se grafické vlákno mohlo plně věnovat uživateli a druhé vlákno zase naslouchání a zpracování RSS kanálů. Vlákno se stará kromě kontroly také o uložení nových údajů do textového souboru a o vygenerování XML souboru při překročení časového horizontu, v našem případě jednoho dne.

Princip kontroly je zobrazen na obrázku 2.3. Na začátku je aplikováno parsování RSS kanálu (viz 2.7). Poté dochází k prohledávání uložených kanálů. Pokud se nenajde shoda, tak se jedná o nový feed (zpravidla při prvním startu či smazání souboru *backupFeedu.ser*, kam se ukládají všechny feedy pro znovunačtení), který se celý uloží do textového souboru. Při shodě se naopak ukládají jen zprávy, které se ve starém feedu nevyskytují. Kontrola probíhá pro každý článek se všemi články ze starého feedu, a to z toho důvodu, že se nové zprávy ne vždy zařazují pouze na začátek. Samotné vyhodnocování shody článků probíhá na základě URL nebo titulku.



Obr. 2.3: Stavový diagram kontroly RSS kanálů

2.5 Volba parseru XML dokumentů

Volba parseru probíhala na základě požadavků, které jsou kladeny na zpracování XML souborů. Pro naši práci se nejlépe hodí zpracování proudové, respektive StAX (viz tabulka 1.1), jelikož se očekává zpracovávání velkých objemů dat, nutnost rychlého zápisu a čtení. Díky použitému parseru je dosaženo malé paměťové náročnosti což znamená, že zpracování nezatěžuje tolik procesor jako jiné metody. Pro zápis a čtení jsou použity třídy XMLOutputFactory, XMLStreamWriter, XMLInputFactory a XMLStreamReader z knihovny javax.xml.stream.

2.6 Návrh databáze a formátu XML dokumentů

Při návrhu struktury databáze bylo důležité stanovit potřebné informace pro možné další zpracování. Proto byla struktura systému navržena následujícím způsobem zachyceným v tabulce:

Název tagu	Popis
databaze	Rodičovský tag zahrnující všechny záznamy získaného textu
zaznam	Jednotlivý záznam obsahující potřebné informace
titulek	Stručný popis článku či textu
text	Získaný text
casovaznacka	Časový údaj vzniku textu
url	Odkaz na získaný text

Tab. 2.1: Význam tagů v XML databázi

Možné následné zpracování umožňuje díky přidruženým informacím lépe vyhodnotit uložený text. Například díky URL je možné vyhodnotit o jaký text se jedná a do jaké kategorie patří (sport, věda, kultura…). Důležitost časové známky spočívá v aktuálnosti údaje, případně v možné časové platnosti.

Zde je příklad reálného záznamu z databáze v XML formátu:

Kód 2.1: Příklad reálné podoby záznamu v databázi

```
1 <zaznam>
2  <titulek>
3   Apple v roce 2011 prodal více iOS zařízení než počítačů Mac za 28 let
4  </titulek>
5  <text>
6   Horace Dedi, analytik společnosti Asymco, si nedávno vzal na mušku
       statistiky společnosti Apple. Krom standardně zajímavých čísel...
7  </text>
```

```
 8 <casovaznacka>
 9  Thu, 23 Feb 2012 09:24:00 GMT
10 </casovaznacka>
11 <url>
12  http://www.zive.cz/bleskovky/apple-v-roce-2011-prodal-vice-ios-
        zarizeni-nez-pocitacu-mac-za-28-let/sc-4-a-162466/default.aspx
13 </url>
14 </zaznam>
```

2.7 Parsování RSS

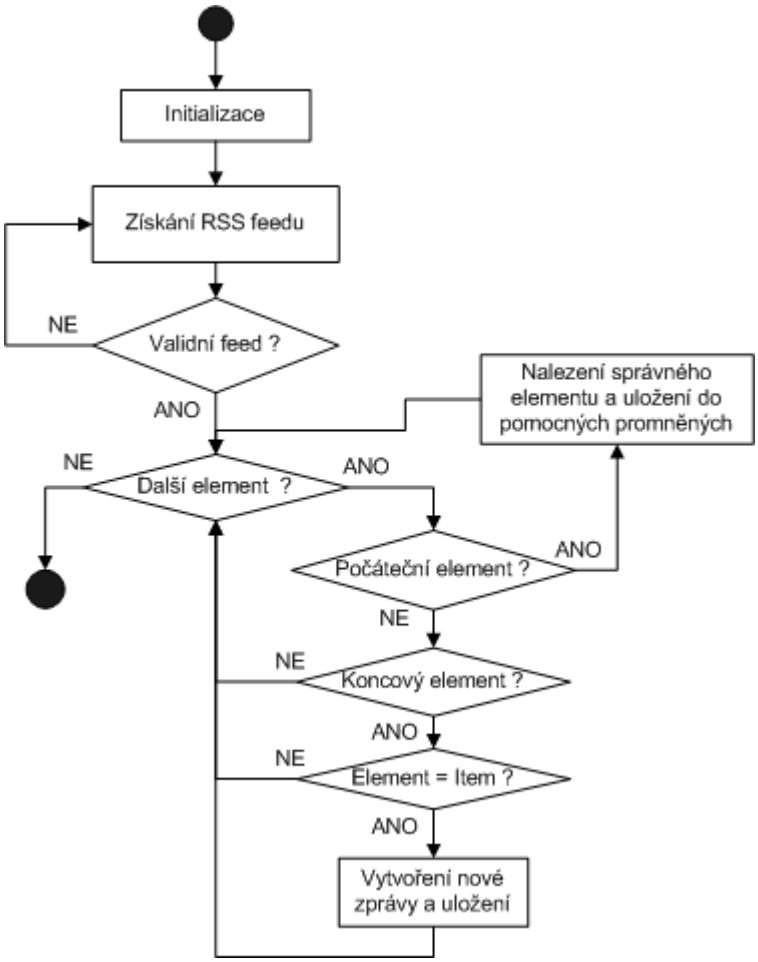
Zpracování RSS kanálu probíhá v podobném duchu jako zpracování jakéhokoliv jiného druhu XML (viz obrázek 2.4 a zdrojový kód 2.2). Na začátku procesu se kontroluje celistvost dokumentu. V našem případě se provede validace a pokud není dokument validní, tak se program pokusí získat obsah kanálu znovu. Po úspěšné validaci již probíhá samotné čtení struktury pomocí metody StAX (viz 1.7.1). Při každém nalezeném počátečním tagu se otestuje jeho název a uloží se do příslušné pomocné proměnné. Po nalezení ukončovacího tagu celé zprávy (každá zpráva je ohraničena tagem *item*) se vytvoří nová zpráva ve feedu s parametry získanými díky pomocným proměnným. Cyklus čtení se opakuje dokud se nepřečte celý obsah kanálu.

2.8 Vyhledávání v databázi

Celá databáze je tvořena z textů získaných buď uživatelem ze základního módu, případně přes Twitter API (Application Programming Interface), nebo se vytváří automaticky díky naslouchání RSS kanálům (viz 2.4). Ukládanému textu je přidělena časová značka (vygenerovaná či získána ze stránek) a odkaz, kde byl nalezen. Generování databáze probíhá při startu, pokud jsou zjištěny staré záznamy, nebo při přechodu na nový den, jelikož systém databáze je tvořen soubory patřícími jednotlivým dnům. K tomuto kroku bylo přistoupeno z důvodu nemožnosti připisování do existujícího souboru (viz 1.1).

Princip vyhledávání spočívá v parsování XML dokumentů, které program dříve vytvořil. V počátku vyhledávání si program obstará pole s názvy XML souborů. Poté začne sekvenčně parsovat jednotlivé dokumenty a hledat začáteční a koncové tagy viz 2.2. Při hledání není možné se vrátit v čtení dokumentu, proto je nutné si nejprve načíst jeden celý záznam, který se následně otestuje podle uživatelem nastavených pravidel, a pokud se najde shoda, tak program záznam vypíše.

Metoda vyhledávání pomocí klíčového slova podporuje 2 druhy specifikace. Jedná se buď o vyhledávání přesné, kdy program hledá řetězec znaků přesně tak jak byl



Obr. 2.4: Stavový diagram parsování RSS kanálu

zadán, a nebo nepřesné. V tomto případě program převádí klíčové slovo a prohledávaný obsah do malých písmen. Druhou specifikací je hledání po slovech nebo celého zadaného řetězce. Zde program jednoduše rozdělí slova podle mezer a hledá každé slovo zvlášť.

Vyhledávání kořenů slov či podobných slov zde není možné, protože to není předmětem této bakalářské práce.

Kód 2.2: Vyhledávací funkce

```
1 XMLInputFactory vstup = XMLInputFactory.newInstance();
2 XMLStreamReader ctecka = vstup.createXMLStreamReader( new FileReader("
    data//" + list[i]));
3   while(ctecka.hasNext()){
4     ctecka.next();
5     if(ctecka.getEventType() == XMLStreamReader.START_ELEMENT){
6       if (ctecka.getLocalName() == (TITUL) ) {
7         ctecka.next();
8         titulek = "";
```

```
9           while (ctecka.getEventType() != XMLStreamReader.END_ELEMENT) {
10            titulek = titulek + ctecka.getText();
11            ctecka.next();
12          }
13          continue;
14        }
15        ...
16      }
17      else if (ctecka.getEventType() == XMLStreamReader.END_ELEMENT) {
18        if (ctecka.getLocalName() == (ZAZNAM)) {
19          n++;
20          if (jCheckBox2.isSelected()) {
21            klice = klic.split(" ");
22            for (String item : klice) {
23              if (KeyTest(item,obsah)) {
24                textArea1.setText(textArea1.getText() + cas + " -
                      Titulek: " + titulek + "\nZprava: " + obsah + "\nURL
                      : " + odkaz + "\n");
25              }
26            }
27          }
28          else {
29            if (KeyTest(klic,obsah)) {
30              textArea1.setText(textArea1.getText() + cas + " - Titulek:
                      " + titulek + "\nZprava: " + obsah + "\nURL: " +
                    odkaz + "\n");
31            }
32          }
33        }
34      }
```

2.9 Ukládání stavu programu

Při hledání způsobů uložení dat programu pro následné znovunačtení byly vyzkoušeny různé varianty jako jsou textové soubory, databáze a serializace. Následně byla zvolena knihovna java.io.Serializable, jelikož umožňuje uložit instanci objektu a opět ji zrekonstruovat. Mimo to je navíc pro tuto knihovnu v Javě přímá podpora.

Uložení probíhá do souboru *backupFeedu.ser* každou celou hodinu, aby se zabránilo ztrátě dat při pádu systému nebo aplikace, a samozřejmě před ukončením programu. Ukládaná data jsou RSS feedy, hodina uložení a datum posledního vytvoření XML dokumentu. Proměnná s datem informuje program o aktuálnosti uložených záznamů v textovém souboru *backup.txt*. Pokud je datum při spuštění programu neaktuální, je vygenerován nový XML soubor. Do něj jsou uložena stará data

ze zmíněného textového souboru, respektive převedena do strukturované podoby.

2.10 Zdroje dat

Pro dolování textu z Internetu byly vybrány dva druhy textu. Spisovný text psaný redaktory na zpravodajských serverech jako jsou Novinky.cz, Idnes.cz, Živě.cz, Aktuálně.cz, Lidovky a další. Druhý druh textu je získáván ze sociální sítě Twitter. API pracující se serverem disponuje mnoha druhy specifikace vyhledání. Je možnost např. hledání uživatele samotného nebo ve zprávách uživatelů. U zpráv je možnost nastavit geologickou polohu, kde zprávy vznikly.

2.11 Extrakce celého článku

Na blok parsování navazuje blok extrakce celého článku, který dokáže ze serveru Novinky.cz extrahovat jakýkoliv článek pouze na základě odkazu. Tento blok je tedy používán při práci s RSS kanálem daného serveru, aby místo stručné zprávy obsaženou ve feedu, vyextrahoval celý článek. Používá k tomu znalost HTML kódu, respektive značek, díky kterým je schopen přesně najít jak titulek, tak i samotný obsah článku (viz příloha A.1.2).

2.12 Extrakce textů ze sociální sítě

Server Twitter je sociální síť, která poskytuje uživatelům takzvané mikroblogy, kde mohou vkládat své tweety neboli příspěvky. Mají přesně definovanou délku, a to je 140 znaků. Tyto tweety se po zadání uživatelem objeví na jeho profilové stránce a na stránkách jeho *followerů* neboli odběratelů těchto příspěvků. Tato služba se také nazývá „SMS internet“, protože podporuje zasílání tweetů do mobilu pomocí krátkých textových zpráv (SMS). V současné době chytrých telefonů se více uplatňují různé aplikace do mobilu než do stolních počítačů. Velká popularita Twitteru je zapříčiněna hlavně díky jednoduchosti a slavným uživatelům jako je Lady Gaga, různí politici či jinak známý lidé.

Hlavní důvod zvolení této sociální sítě jako dalšího zdroje je existující podpora v podobě dostupných API, které umožňují různé druhy přístupů.

2.12.1 Twitter API

Během jediného dne se na Twitteru objeví přes 200 milionů zpráv. Toto ohromné množství dat si vyžaduje nástroje, které pomohou uživatelům v orientaci. Tyto

nástroje (API) pomáhají hlavně vývojářům, aby mohli propojit své programy se serverem. Vývoj nástrojů jde neustále dopředu a není tedy zaručeno, že bude vše stále fungovat.

- **Twitter for Websites** neboli Twitter pro weby je soubor nástrojů, který nabízí jednoduchou a rychlou integraci Twitteru na webovou stránku. Jedná se například o tlačítko, kterým uživatelé mohou pomocí jednoho kliknutí vytvořit odkaz na svůj profil.
- **Search API** je nástroj pro vyhledávání. Zahrnuje jak klíčové vyhledávání, tak i zprávy od daného uživatele.
- **REST API** umožňuje přístup k základním funkcím Twitteru jako je například časové pásmo, aktualizace statusu, informace o uživateli apod.
- **Streaming API** je nástroj pracující v reálném čase. Je vhodný pro vývojáře s velkými datovými nároky, kteří se zajímají o nejrůznější analýzy nebo dolování dat. Základem tohoto nástroje je vytvoření dlouho trvajícího HTTP spojení a jeho stále udržování.

2.12.2 Twitter4J

Twitter4J je neoficiální knihovna do Javy, která umožňuje komunikaci s Twitter API. Pomocí této knihovny se lehce integrují do Java programu Twitter služby.

Tato knihovna patří pod Apache License 2.0. Knihovna je dělána jen v jazyce Java a běží pod jakoukoliv Java platformou od verze 1.4.2. Pro běh není potřeba žádná další knihovna. Vestavěná podpora OAuth přináší možnost využití v Android platformě či Google APP Engine.

Pro používání Twitter API je nutné se zaregistrovat ve vývojářské části, kde získáte přístup. Bez tohoto přístupu není možné jakékoliv používání Twitter služeb.

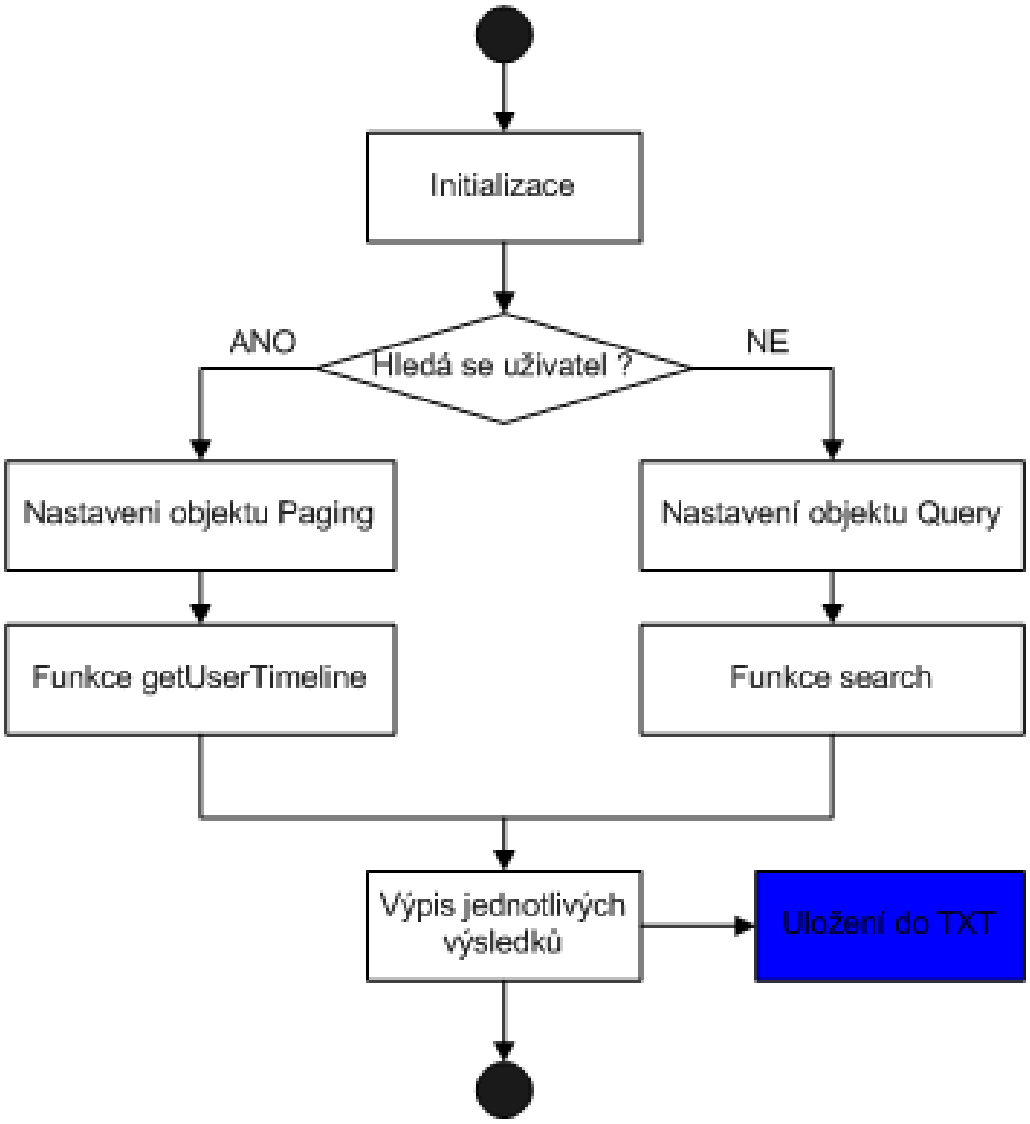
Obr. 2.5: Blokové schéma používání Twitter4J

Funkce pro práci s knihovnou Twitter4J při startu inicializuje třídy pro úspěšné připojení k serveru (ConfigurationBuilder, TwitterFactory a Twitter), a poté přechází k vytvoření požadavku podle uživatelského nastavení (viz obrázek 2.5). Pokud je hledán uživatel a jeho tweety, nastaví se objekt Paging, který nese dva parametry.

První představuje číslo stránky. Druhý definuje počet zobrazených záznamů. Možnost nastavení těchto parametrů uživatelem se zdála zbytečná. Při hledání tweetu dle zadaného slova se musí nastavit objekt Qeury, který nese jediný parametr, a tím je klíčové slovo. Je zde však možnost použití funkce setGeoCode, která umožňuje specifikaci hledaných tweetů podle zeměpisné polohy. Jednotlivé parametry je možné nastavit pomocí grafického rozhraní. Odpovídají zeměpisné šířce, zeměpisné výšce a radiusu hledání v kilometrech. Pro představení této možnosti byli předdefinovány 3 druhy nastavení: vypnutí (nastavení parametrů na 0), přibližná oblast České republiky a možnost vlastního nastavení. Po úspěšném nalezení se systém postará o zobrazení a uložení do textového souboru, ze kterého se poté vytváří strukturovaný XML soubor. Proto je nutné kromě textu uložit také časovou známku a příslušný odkaz (viz obrázek 2.6 a zdrojový kód 2.3).

Kód 2.3: Funkce pro práci s Twitter4J

```
1 String klic = jTextField3.getText();
2 SimpleDateFormat stamp = new SimpleDateFormat(Main.STAMP_FORMAT);
3 ConfigurationBuilder cb = new ConfigurationBuilder();
4 cb.setDebugEnabled(true)
5         .setOAuthConsumerKey("FffMuLOdbsBRBjlZKobqw")
6         .setOAuthConsumerSecret("
              tj8Txpef8JFsYsC1mJwj7EvUH6PelChZ5ACgpDY0Lk")
7         .setOAuthAccessToken("488405530-
              FU54hGfjWqb7rUDg1RUTYL68C56Hr1zR9h8G5fBK")
8         .setOAuthAccessTokenSecret("
              G3bSiNCOL3q8lj86rKSBxeGUmL9JoH5QznKdNyX84");
9 TwitterFactory tf = new TwitterFactory(cb.build());
10 Twitter twitter = tf.getInstance();
11 try {
12         if (userSearching) {
13     Paging paging = new Paging(1, 100);
14     List<Status> statuses = twitter.getUserTimeline(klic,paging);
15     textArea1.setText("Zpravy uzivatele : @" + klic + "\n");
16     for (int x = 0;x<statuses.size();x++) {
17             textArea1.setText(textArea1.getText()+ (x+1) + ". " +
                   statuses.get(x).getText() + "\n");
18             RssListener.Zaloha(stamp.format(statuses.get(x).
                   getCreatedAt()), "https://twitter.com/#!/@" + klic,
                   statuses.get(x).getUser().getName(), statuses.get(x).
                   getText());
19     }
20   }
21   else {
22     Query query = new Query(klic);
23     if (searching == 1) {
```



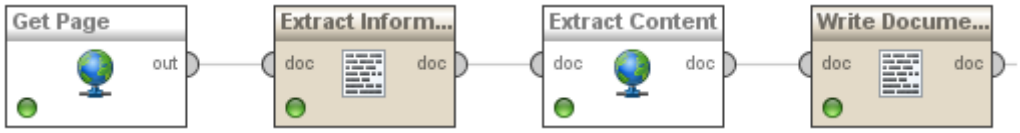
Obr. 2.6: Stavový diagram funkce pro práci s Twitter4J

```
24        textArea1.setText("Uzivatel : zprava \n");
25        query.setGeoCode(new GeoLocation(0,0), 0, Query.KILOMETERS);
26      }
27      else if (searching == 2) {
28        textArea1.setText("Uzivatel : zprava (Stred Ceske republiky s
              radiusem 150 km)\n");
29        query.setGeoCode(new GeoLocation(49.802541,15.688477), 150, Query
              .KILOMETERS);
30      }
31      else {
32        textArea1.setText("Uzivatel : zprava (Zvolene souradnice s
              radiusem " + jTextField6.getText() + " km)\n");
33        query.setGeoCode(new GeoLocation(Double.parseDouble(jTextField4.
              getText()),Double.parseDouble(jTextField5.getText())), Double
```

```
            .parseDouble(jTextField6.getText()), Query.KILOMETERS);
34      }
35      QueryResult result = twitter.search(query);
36      Main.log.info("Twitter extraction");
37      for (Object tweet1 : result.getTweets()) {
38        Tweet tw = (Tweet) tweet1;
39        textArea1.setText(textArea1.getText() + tw.getFromUser() + " : "
              + tw.getText() + " (" + tw.getLocation() +")\n");
40        RssListener.Zaloha(stamp.format(tw.getCreatedAt()), "https://
              twitter.com/#!/@" + tw.getFromUser(), tw.getFromUser(), tw.
              getText());
41      }
42    }
```

2.13 Možnost rozšíření pomocí RapidMiner

Mezi jedny z nejpoužívanějších opensource prostředí pro datovou analýzu patří program RapidMiner. Je dostupný jako samostatná aplikace pro datovou analýzu nebo jako dolovací systém pro integraci do jiné aplikace. Bylo by tedy žádoucí využít jeho existujících operátorů a rozšíření (Web mining), aby byl navýšen počet zdrojů dat pro databázi.

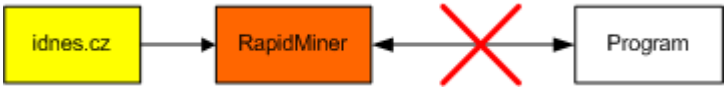


Obr. 2.7: Schéma v Rapidmineru

Byla prozkoumána možnost začlenit prostředí Rapidminer do navrhovaného programu, nicméně se nepodařilo navázání vzájemné komunikace. Byly aplikovány dva způsoby řešení. První řešení spočívalo v návrhu v samotné programu pomocí již vytvořených operátorů, které slouží ke stažení zdrojového kódu, extrahování článku a uložení (viz obrázek 2.7). Nebyla však nalezena cesta pro předávání parametrů nesoucích URL požadovaných stránek (viz obrázek 2.8). Možné řešení by bylo ve vytvoření pluginu pro práci s Java programem, ale to je nad rámec této bakalářské práce. Z tohoto důvodu nebyly vzniklé problémy dále detailněji řešeny. Nicméně se jedná o jedno z možných zajímavých rozšíření programu, které by v budoucnu přineslo nový způsob získání textu do databáze.

Druhý způsob bylo použití knihovny Rapidmineru přímo v Javě. Byly použity stejné bloky, ale v prostředí Java již bylo možné předat parametr s URL. Ovšem

nastal problém s výstupem, protože se extrahovaná data uložila do metadat zpracovávaného dokumentu, ke kterým nebyla nalezena předávací funkce.



Obr. 2.8: Blokové schéma ukazující výskyt problému

2.14 Zhodnocení programu

Tato kapitola se bude zabývat srovnáním navrženého programu vůči existujícím řešením, jenž budou nyní představeny.

2.14.1 Happy Harvester 2

Placený program pro dolování textu. Cena plné verze je 89 €. Ovladatelnost a vzhled programu jsou výborně pojaty, a proto práce s programem je velmi jednoduchá. Dolovaným údajům se přidělují různá kritéria, která se ale definují stylem „co je před“ a „co je po“ požadovaném údaji. Lze vyhledávat na více URL adresách. Výstup programu je vyveden do tabulky, která dovoluje i řazení. Vzhled programu viz příloha A.1.

2.14.2 Web Data Extractor 7.2

Další placený program. Jeho cena je 112.20 €. Trochu složitější nastavení pro vyhledávání, ale za to program dokáže rekurzivně prohledávat další webové stránky z nalezených URL adres. Bohužel dokáže vyhledávat pouze údaje jako email, telefonní číslo, fax nebo uloží celou stránku. Nějaká vlastní definice vyhledávání bohužel není možná. Vzhled programu viz příloha A.2.

2.14.3 WebExtractor360

Velmi jednoduchý a volně šiřitelný program, který má předem definované vyhledávací klíče, ale nabízí možnost je upravovat. Vyhledávací klíč obsahuje regulární výrazy, ovšem jiná možnost specifikace není možná. Rekurzivní vyhledávání lze lehce nastavit. Uložení dat v programu není podporována a po ukončení se data ztratí. Lze však export do TXT souboru. Vzhled programu viz příloha A.3.

2.14.4 Web-Harvest 2.0

Další volně šiřitelný program, který je vytvořený v programovacím jazyce Java, což se asi nejvíce blíží k navrženému programu. Extrakce dat je založena na již ověřených technikách pro práci s HTML dokumenty a jsou to Xpath, Xslt, Xquery a regulární výrazy. Klíč k vyhledávání se zadává ve speciálním XML jazyce programu. Proto program není určen pro všechny uživatele a nastavení hledání je značně složité. Po úspěšném nastavení dokáže program hledat a ukládat prakticky cokoliv. XML konfigurační soubor obsahuje sekvenci procesů, které řeší svou vlastní podúlohu. Kromě složité manipulace tedy není co programu vytknout. Export dat je možný do XML dokumentů. Vzhled programu viz příloha A.4.

2.14.5 Srovnání

Níže uvedená tabulka srovnává existující programy s navrhovaným:

	Web Extractor 360	Happy Harvest 2.0	Web-Harvest	Web Data Extractor	Můj program
Česky	Ne	Ne	Ne	Ne	Ano
Prohledávání na základě nalezených URL	Ano	Ano	Ano	Ano	Ne
Nutná znalost jazyka k práci s programem	Ne	Ne	Ano	Ano	Ne
Automatický režim	Ne	Ne	Ne	Ne	Ano
Práce s RSS	Ne	Ano	Ano	Ne	Ano
Extrakce pouze textu stránky	Ne	Ne	Ano	Ne	Ano
Práce s různými znakovými sadami	Ne	Ne	Ano	Ne	Ano
Předdefinované regulární výrazy	Ano	Ne	Ne	Ano	Ne
Komunikace se sociální sítí	Ne	Ne	Ne	Ne	Ano
Spolupráce s externí API	Ne	Ne	Ne	Ne	Ano
Tvorba XML databáze	Ne	Ano	Ano	Ne	Ano
Zpoplatněno	Ne	Ano	Ano	Ne	Ne

Tab. 2.2: Tabulka srovnání existujících programů

2.15 Možnosti rozšíření programu

Z tabulky 2.2 lze odvodit možnosti rozšíření programu, ale také jeho hlavní výhody jako je třeba čeština, samostatnost aj.

Jedna z možností, jak program rozšířit, by byla implementace prohledávání na základě nalezených URL. Bohužel by si toto rozšíření vyžádalo vytvoření databáze nalezených odkazů, která by postupně více a více zpomalovala chod programu. Bylo by také nutné definovat jak hluboké prohledávání by se provádělo, aby nedošlo k absolutnímu zacyklení programu.

Další oblast rozšíření, která vyplývá z tabulky 2.2 ve srovnání s existujícím programům, je absence předdefinovaných regulárních výrazů. Tyto výrazy jsou používány hlavně pro extrakci dat z internetových stránek. Tato vlastnost se pro náš program nehodí. Možnost využití těchto výrazů by našla uplatnění při prohledávání naší XML databáze. Případné zlepšení prohledávání by bylo možné začleněním knihovny pro vytvoření kořenů slov.

2.16 Spuštění programu

Pro úspěšné spuštění je nutné mít nainstalován kompilátor Java a mít ve složce s programem také soubor **log4j.properties**. Také je nutno mít složku **lib** s knihovnami (viz příloha B.1). Program je možné spustit pomocí souboru **Miner.jar** nebo pomocí příkazového řádku po zadání příkazu **java -jar Miner.jar**.

3 ZÁVĚR

Tato bakalářská práce se zabývá vytvořením databáze extrahovaných dat z Internetu. Hlavní přínosem práce je implementace systému, který umožňuje získávání textů ze síťových serverů, a uložení získaných dat do strukturované databáze, s kterou je možné dále jednoduše a rychle pracovat.

Jako strukturovaná forma uložení dat byl vybrán jazyk XML, který je vhodný pro tvorbu databáze a je v mnoha systémech podporován.

Z možných programovacích jazyků se přistoupilo k jazyku Java. Největší výhody jazyku, které požadujeme, spočívají v snadné práci se sítí, se soubory, s textem a s XML. Mezi další podstatné výhody patří mimo jiné nezávislost na programovacím prostředí a platformě.

Princip programu je zobrazen v blokovém schématu, které je možno nalézt v praktické části. Dále jsou v praktické části rozebrány vybrané body blokového schématu, které podrobněji vysvětlují funkci daného bloku. Například kapitola věnující se bloku vyhledávání obsahuje zdrojový kód a popis vysvětlující jednotlivé kroky.

Hlavní výhoda programu spočívá ve využití více druhů způsobů získání textu. Dokáže v automatickém režimu kontrolovat řadu RSS kanálů a získávat z nich krátké články. Dále je schopen na základě zadaného URL vyextrahovat pouhý text nebo po zadání klíčového slova vyhledat uživatelské zprávy či zprávy obsahující hledané slovo na serveru Twitter.com. V neposlední řadě dokáže extrahovat celé články ze serveru Novinky.cz na základě odkazů získaných z RSS kanálu. Snaha využít více způsobů získání textu vedla k začlenění programu RapidMiner do této práce. Bohužel se nezdařilo vytvořit komunikaci potřebnou k výměně dat.

Další výhody je možné nalézt v tabulce srovnání již existující řešení dolování dat s vytvořeným programem. Z tabulky lze vyčíst, že se vytvořený program liší oproti ostatním hlavně v automatickém režimu, českém jazyce aj.

Výsledky získávání textů mohou být dále zlepšeny několika kroky popsaných v předposlední kapitole teoretické části 2.15.

Jako vedlejší produkt v rámci této práce vznikla možnost uživatelského zobrazování aktuálních RSS kanálů.

V první příloze jsou uvedeny další ukázky zdrojových kódů použitých v programu, ukázka části zpracovávaného RSS kanálu a grafické rozhraní existujících programů. V druhé příloze je uveden obsah přiloženého CD.

LITERATURA

[1] PETR, Pavel. *Data Mining.* Vyd. 2. Pardubice, 2008. ISBN 978-807-3950-989.

[2] HEROUT, Pavel. *Java a XML.* 1. vyd. České Budějovice: Kopp, 2007, 313 s. ISBN 978-80-7232-307-4.

[3] 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>.

[4] *BST download* [online]. 2004 [cit. 2010-12-17]. Dostupné z URL: <http://bstdownload.com/reviews/happy-harvester-2/>.

[5] 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 URL: http://www.2009.org/proceedings/pdf/p1169.pdf

[6] 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>.

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

[8] 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_Manazerske 20rozhodovani_VS_2006.pdf>.

[9] KUČERA , Miroslav. *Základní struktura dokumentu* [online]. 1999 [cit. 2010-12-17]. Miroslav Kučera. Dostupné z URL: <http://interval.cz/clanky/kurz-html-zakladni-struktura-dokumentu/>.

[10] 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>.

[11] LIU, Bing. *Web Data Mining.* Amazon.com : Hardcover , 2006. 236 s.

[12] NEKVASIL, Marek. *VYUŽITÍ ONTOLOGIÍ PŘI INDUKCI WRAPPERU.* Praha, 2006. 58 s. Diplomová práce. Vysoká škola ekonomická v Praze.

[13] 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>.

[14] 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/x -» 21:50:40: sedlac5.htm>.

[15] *SmElis* [online]. 2006 [cit. 2010-12-17]. Dostupné z URL: <http://www.smelis.com/>.

[16] *Tvorba-Webu.cz* [online]. 2008 [cit. 2010-12-15]. DOM: Document Object Model. Dostupné z URL: <http://www.tvorba-webu.cz/dom/>.

[17] *W3C* [online]. 1994 [cit. 2010-12-17]. Dostupné z URL: <http://www.w3.org/>.

[18] *W3C* [online]. 1999 [cit. 2010-12-15]. XML Path Language (XPath). Dostupné z URL: <http://www.w3.org/TR/xpath/>.

[19] *W3SCHOOLS* [online]. 2006 [cit. 2010-12-15]. HTML DOM Introduction. Dostupné z URL: <http://www.w3schools.com/htmldom/dom_intro.asp>.

[20] *Web-Harvest* [online]. 2006 [cit. 2010-12-15]. Dostupné z URL: <http://web-harvest.sourceforge.net/>.

[21] 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 PŘÍLOH

A PRVNÍ PŘÍLOHA

A.1 Zdrojový kód

A.1.1 Funkce pro stažení zdrojového kódu

```
1 public static String Stazeni(String page){
2 String strana = new String();
3 try {
4         URL u;
5         u = new URL(page);
6         URLConnection spojeni = null;
7         String s;
8         spojeni = u.openConnection();
9         spojeni.connect();
10        BufferedReader reader = new BufferedReader(new
              InputStreamReader(spojeni.getInputStream()));
11        while ( (s = reader.readLine()) != null) {
12                strana += s;
13                strana += '\n';
14        }
15        reader.close();
16        }
17        catch (Exception e) {
18                Main.log.error(e);
19        }
20        return strana;
```

Funkce se volá s požadovanou URL a vrácí zdrojový kód stránky typu String.

A.1.2 Funkce extrakce zprávy ze serveru Novinky.cz

```
1 private static String Novinky(String url){
2         boolean titulek = true;
3         String str = new String();
4         String obsah = MainFrame.Stazeni(url);
5         int zacatek=obsah.indexOf("<!-- Sklik-kontext-start -->");
6         int konec=obsah.indexOf("<!-- Sklik-kontext-stop -->");
7         while (zacatek != -1) {
8     if (titulek) {
9         titulek = false;
10    }
11    else {
12        str=str + MainFrame.OdstranitTagy(obsah.substring(zacatek + 28,
              konec)) + "\n";
13    }
14    zacatek = obsah.indexOf("<!-- Sklik-kontext-start -->",zacatek+1);
```

```
15      konec=obsah.indexOf("<!-- Sklik-kontext-stop -->",konec+1);
16    }
17    return str;
18 }
```

Pro server Novinky.cz byla vytvořena speciální funkce, která dokáže ze zdrojového textu vyextrahovat pouze zprávu. Tato zpráva je potřeba jěště ošetřit od HTML tagů pomocí následující funkce.

A.1.3 Funkce odstraňující HTML tagy

```
1 private static String HTMLTagy(String strana,String start,String stop){
2           String str = new String();
3           int zacatek=strana.indexOf(start);
4           int konec=strana.indexOf(stop);
5           try {
6           if (zacatek != -1) {
7                 str = strana.substring(0, zacatek);
8                 zacatek = strana.indexOf(start,zacatek + start.length());
9                 while (zacatek != -1) {
10                    str = str + strana.substring(konec + stop.length(),
                          zacatek);
11                    zacatek = strana.indexOf(start,zacatek + start.length
                          ());
12                    konec = strana.indexOf(stop,konec + stop.length());
13                }
14                str = str + strana.substring(konec + stop.length(),
                      strana.length());
15          }
16          else {
17              str = strana;
18          }
19          } catch (Exception e) {
20                Main.log.error(e);
21           }
22          return str;
23         }
```

Tato funkce se volá z nadřazené funkce, která obsahuje jednotlivé řetězce (tagy), které jsou nutné odstranit pro dosažení čistého textu.

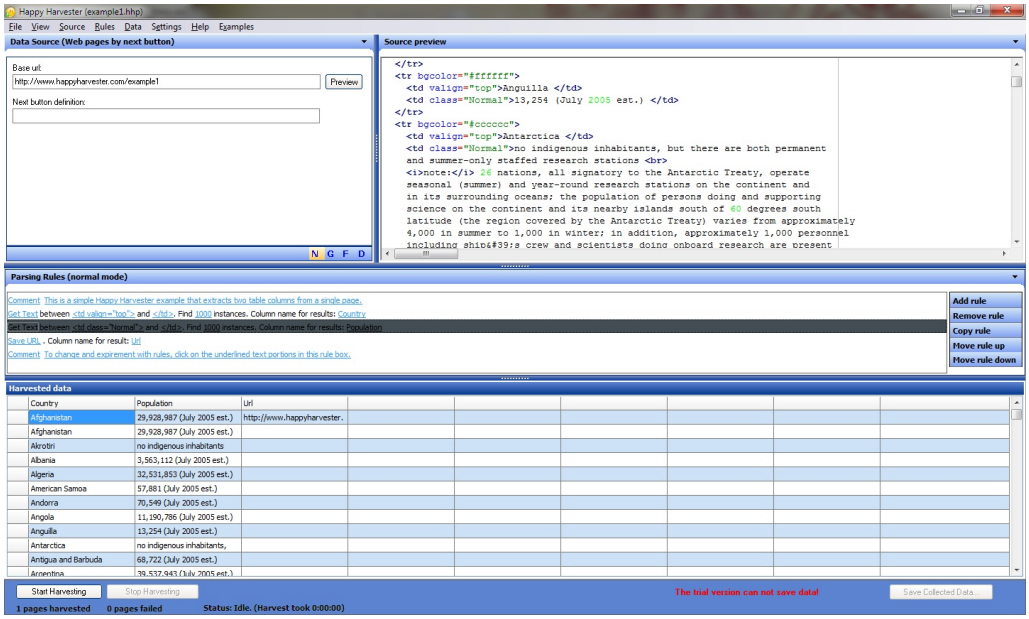
A.1.4 Ukázka RSS feedu

Nejrozšířenější verze RSS 2.0 může mít tuto podobu:

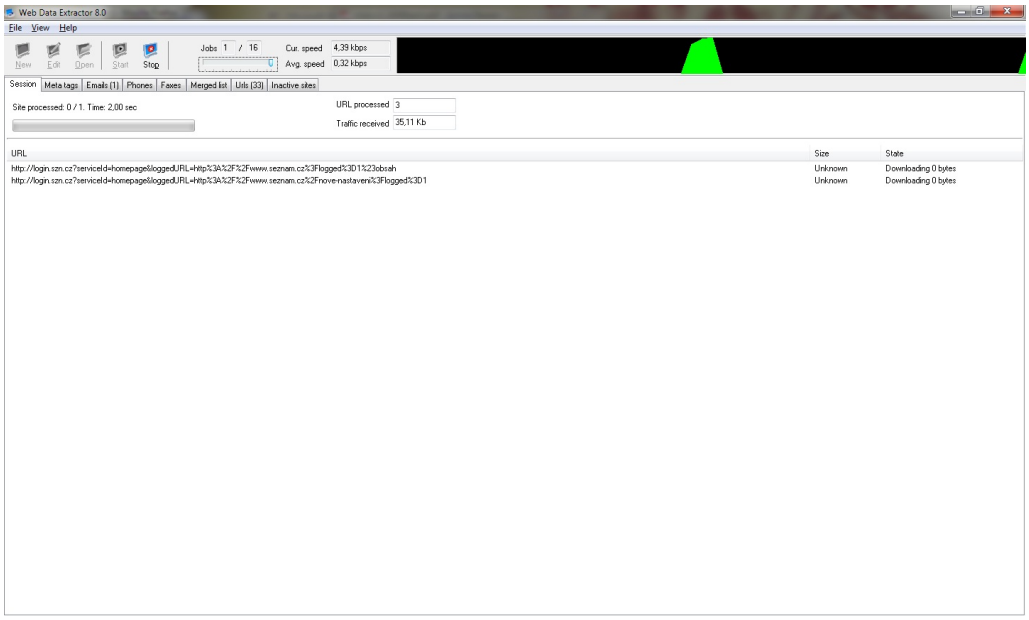
```
1 <?xml version='1.0' encoding='UTF-8'?>
2 <?xml-stylesheet type='text/xsl' href='http://novinky.cz.feedsportal.
    com/xsl/eng/rss.xsl'?>
```

```
3 <rss xmlns:itunes="http://www.itunes.com/dtds/podcast-1.0.dtd" xmlns:dc
    ="http://purl.org/dc/elements/1.1/" xmlns:taxo="http://purl.org/rss
    /1.0/modules/taxonomy/" xmlns:rdf="http://www.w3.org/1999/02/22-rdf
    -syntax-ns#" xmlns:szn="http://www.seznam.cz" version="2.0">
4 <channel><title>Novinky.cz - Hlavní stránka</title>
5 <link>http://www.novinky.cz/</link>
6 <description>Novinky.cz - zpravodajský server</description>
7 <language>cs</language>
8 <pubDate>Sat, 28 Apr 2012 12:05:55 GMT</pubDate>
9 <lastBuildDate>Sat, 28 Apr 2012 12:05:55 GMT</lastBuildDate>
10 <ttl>2</ttl>
11 <image><title>Novinky.cz - Hlavní stránka</title>
12 <url>http://www.novinky.cz/static/images/logo.gif</url>
13 <link>http://www.novinky.cz/</link></image>
14 <item><title>Americké pobřeží ohrožují obří krevety větší než lidské
    chodidlo</title>
15 <link>http://novinky.cz.feedsportal.com/c/33064/f/534746/s/1ed3142c/l/0
    L0Snovinky0Bcz0Ckoktej/0
    Eamericke0Epobrezi0Eohrozuji0Eobri0Ekrevety0Bhtml/story01.htm</link
    >
16 <description>&lt;img src="http://media.novinky.cz/435/124350-nextstory1
    -ubf2g.jpg"/&gt; Asijské tygří krevety berou americké pobřeží
    útokem. Nejde přitom jen o jihovýchodní pobřeží, ale také o Mexický
     záliv, kde se krevety velmi často chytají. Američtí biologové se
    obávají toho, že asijské protějšky značně sníží čísla svých menších
     protějšků, které s oblibou pojídají. Asijské tygří krevety se
    přitom letos v oblasti vyskytují až desetkrát častěji než loni.</
    description>
17 <pubDate>Sat, 28 Apr 2012 11:04:01 GMT</pubDate>
18 <guid isPermaLink="false">266117</guid>
19 <szn:upDate>20120428T14:04:01+0000</szn:upDate>
20 <szn:image>20120428T14:04:01+0000</szn:image></item>
21 .....
```

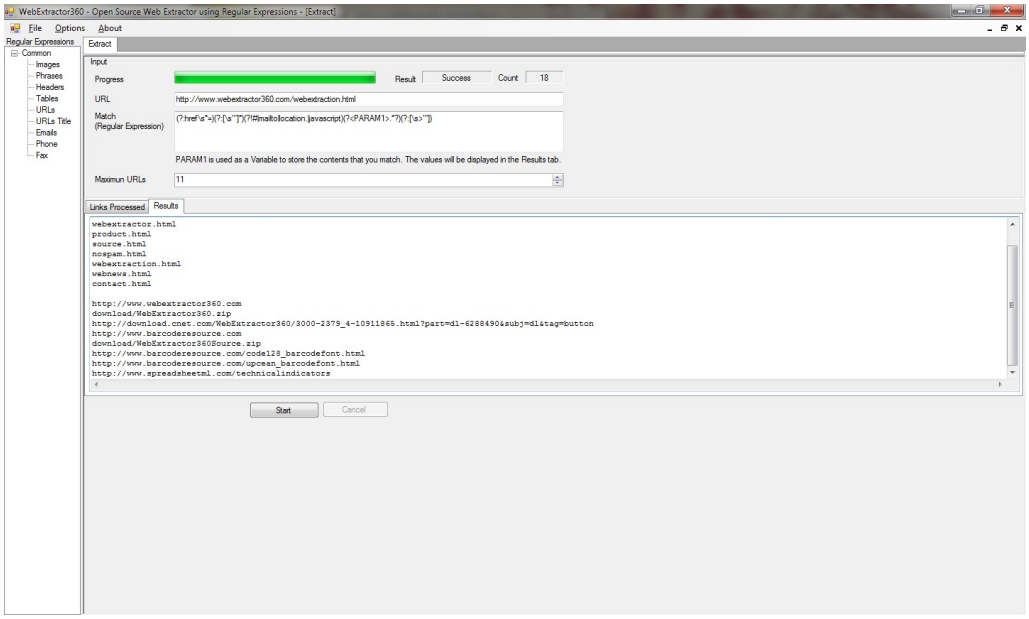
A.2 Grafické rozhraní programů



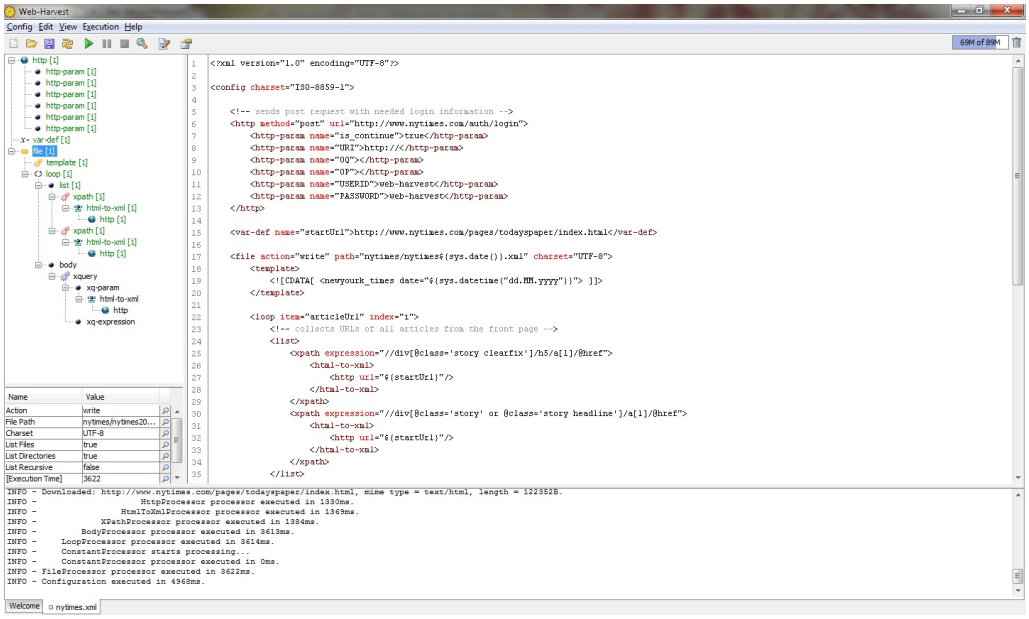
Obr. A.1: Vzhled programu Happy Harvester 2



Obr. A.2: Vzhled programu WebData Extractor



Obr. A.3: Vzhled programu WebExtractor 360



Obr. A.4: Vzhled programu Web Harvest 2.0

B DRUHÁ PŘÍLOHA

B.1 Seznam přiloženého CD

Bakalarska_prace		
	TextMining.pdf	
Miner		
	Miner.jar	
	log4j.properties	
	lib	
		log4j-1.2.16.jar, twitter4j-async-2.2.5.jar, twitter4j-core-2.2.5.jar, twitter4j-media-support-2.2.5.jar, twitter4j-stream-2.2.5.jar
Miner-zdroj.zip		
	Zdrojové kódy programu a knihoven	