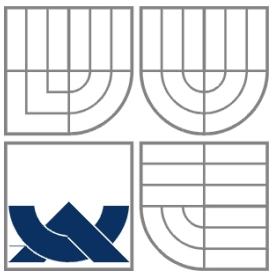
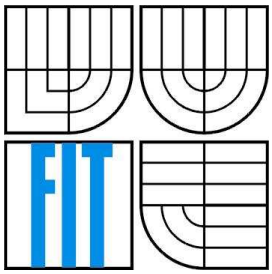


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

SLEDOVÁNÍ A ANALÝZA ZMĚN WEBOVÝCH STRÁNEK

MONITORING AND ANALYSIS OF WEB PAGE CHANGES

DIPLOMOVÁ PRÁCE
DIPLOMA THESIS

AUTOR PRÁCE
AUTHOR

BC. DAVID BLAHETA

VEDOUCÍ PRÁCE
SUPERVISOR

MGR. MAREK RYCHLÝ, Ph.D.

BRNO 2010

Abstrakt

Obsahem diplomové práce je řešení problematiky sledování a analýzy změn na webových stránkách. Práce nabízí přehled nástrojů a aplikací, které se dnes pro sledování používají a zabývá se jejich funkčností a implementací. Nejdůležitější část práce je věnována tvorbě informačního systému pro sledování změn. Přináší teoretický úvod, návrh a realizaci aplikace, která má za cíl sledování změn na základě uživatelsky definovaných filtrů. Aplikace tak podává uživateli informace jen o těch částech webu, které ho opravdu zajímají. Výsledky jsou poté prezentovány ve formě RSS kanálů. V práci je popsán kompletní návrh a implementace včetně možných budoucích rozšíření.

Klíčová slova

Sledování změn, detekce změn, analýza změn, web, webové stránky, RSS, XML, HTML, informační systém

Abstract

The content of this thesis is the issue of monitoring and analysis of web page changes. Work offers an overview of tools and applications that are used today to monitoring and deals with their functionality and implementation. The most important part is devoted to creating an information system for detecting and monitoring changes. Provides a theoretical introduction, design and implementation of an application, that aims to detect changes based on user defined filters. So application gives users only the information, which really interested him. Results are presented in the form of RSS channels. The paper describes a complete design and implementation, including possible future expansion.

Keywords

Changes monitoring, change detection, analysis of changes, web, web pages, RSS, XML, HTML, information system

Citace

David Blaheta: Sledování a analýza změn webových stránek, diplomová práce, Brno, FIT VUT v Brně, 2010

Sledování a analýza změn webových stránek

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením Mgr. Marka Rychlého Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……

David Blaheta

26. 5. 2010

Poděkování

Na tomto místě bych rád poděkoval vedoucímu své diplomové práce, Mgr. Marku Rychlému Ph.D., za odborné vedení a cenné rady, které mi během vypracování celé práce poskytoval.

Obsah

1 Úvod

Internet je v dnešní době určitě jedním z největších a nejobsáhlejších zdrojů informací. S jeho prudkým rozvojem a dynamičností však souvisí i některá úskalí. Webové stránky na internetu bývají velmi často aktualizovány, a díky jejich velkému množství a často ne příliš přehledné struktuře, je velmi složité všechny novinky sledovat a zaznamenávat. Proces sledování nových informací, které se na webu vyskytují, je tak pro uživatele časově velmi náročný a nepohodlný.

Problém se sledováním těchto změn na webu se proto snaží již nějakou dobu řešit aplikace, které jsou dostupné jak offline, tak online přímo na webu. Principem takového sledování je poskytnout uživateli přehled o tom, co se na vybrané webové stránce změnilo a poté ho informovat cestou, která je pro něj nejpohodlnější. Uživatel tak dostane informace, které jsou již pro něj zformátovány a nemusí neustále zjišťovat novinky ručně.

Tato diplomová práce se také zabývá aplikací, která uživateli umožní sledovat a detekovat změny na zvoleném webu. Na rozdíl od jiných, již hotových řešení, je větší důraz kladen na filtrovaní informací, které si uživatel zvolí. Výsledná aplikace by tedy neměla přinášet informace o všech změnách na konkrétním webu, ale pouze o změnách ve vybraných částech. Tím je uživateli nabídnut ještě větší komfort, protože data jsou filtrována a analyzována a on uvidí pouze to, co ho opravdu zajímá a není tak zahlcován množstvím zbytečných informací, které internet nabízí. Tento přístup není v online dostupných aplikací příliš častý a na českém internetu se prakticky nevyskytuje aplikace, která by se zabývala tímto přístupem ke sledování.

V diplomové práci jsem se snažil popsat, jak fungují aplikace, které jsou již dnes hotové a přinášejí relevantní výsledky. Dalším problémem, kterým jsem se zabýval, je popis filtrace a šablon, které mají sloužit k prezentaci jednotlivých informací uživateli. Tato část práce je velmi důležitá, protože tvoří základ celého výsledného systému a předcházela jí důkladná analýza a zpracování velkého množství informací. Dále je v práci uveden návrh informačního systému, který umožňuje sledování změn a jeho implementace.

Výsledky, které aplikace podává, byly systematicky testovány a jsou tedy ověřeny na mnoha internetových stránkách, od blogů až po zpravodajské servery.

2 Existující nástroje pro detekci změn

V této kapitole budou popsány nejznámější nástroje, které nabízejí detekci změn na webových stránkách. Takových nástrojů je celá řada a dají se rozdělit na dva tábory. Jsou to aplikace, které jsou dostupné jako desktopové aplikace a nabízejí většinou velké množství funkcí. Jejich použití je ale omezeno jen na přístroj, na kterém jsou nainstalovány. Další skupinou, která je pro tento projekt podstatnější, jsou aplikace, které jsou dostupné online na webu.

Online aplikace jsou postaveny na architektuře klient - server, kde uživatel pomocí klientské části zadává údaje o sledovaných stránkách a server potom tento požadavek zpracovává a vyhodnocuje. Tento druh aplikací nabízí mnoho výhod, díky kterým si získává stále větší oblibu u uživatelů. Největší výhodou je samozřejmě to, že uživatel může webové stránky sledovat z jakéhokoli místa, kde je internet. Zároveň sledování probíhá neustále a není proto nutné mít zapnutu klientskou stanici. Dalším důvodem, který ocení především tvůrci webu, je skutečnost, že aplikace pro sledování je velmi často uvolněna i pro jiné weby a může tak být použita prakticky kdekoli.

2.1 ChangeDetect

ChangeDetect [6, 7] je online aplikace, která se nachází na url http://www.changedetect.com. Tato aplikace patří mezi nejznámější online aplikace pro detekci změn na webu. Jedná se o klasickou klient – server aplikaci. Po zaregistrování si uživatel může vytvořit nový monitor, který se stará o sledování zadané webové stránky. Každý monitor je možné specifikovat, je možné zadat interval mezi detekcemi, slova, která chceme sledovat, nebo fráze, které se na stránkách vyskytují.

Nevýhodou tohoto systému je fakt, že tato služba není poskytována plně zdarma. Verze, za kterou není třeba platit, je omezena počtem sledovaných stránek. Pokud by uživatel chtěl sledovat více stránek, musí platit měsíční poplatky. Systém je jinak zpracován velmi kvalitně a umožňuje například zasílání změn i na instant messagingové programy jako třeba ICQ. Standardně jsou však jednotlivá upozornění zasílána na email.

V běžném nastavení je na email odeslána jen notifikace, že se požadovaná stránka změnila. Po změně nastavení v systému uživatel může dostávat také emaily, kdy jsou v emailu změny na stránce barevně odděleny.

2.2 ChangeDetection.com

Služba ChangeDetection.com [8] se nachází na stejnojmenné url. Tato aplikace je jednodušší než ChangeDetect, ale je nabízena zcela zdarma. Umožňuje přes webové rozhraní zadávat stránky, které

chceme monitorovat a poté upravovat kritéria pro monitorování. Stránky je možné sledovat denně, týdně, měsíčně, nebo pokud se změní něco na RSS kanálu stránky.

Jednotlivá upozornění mohou být dále zasílána v případě, že se změnila velikost sledované stránky, nebo pokud byl nějaký text přidán či smazán. Poslední možností je zaslání upozornění, pokud bylo přidáno nebo naopak smazáno nějaké slovní spojení.

Změny, které byly na webu provedeny, jsou zobrazovány přímo na stránkách provozovatele. Uživatel je pouze pomocí emailu informován, že k nějakým změnám došlo. Výhodou je možnost zobrazení sledované stránky v její skutečné podobě, kde jsou vyznačeny všechny změny (viz obrázek 1).

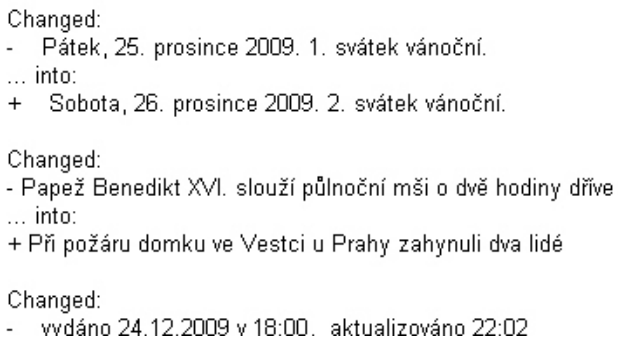
Obrázek 1: zobrazení změn na stránce

ChangeDetection.com zároveň nabízí službu pro webmastery jiných stránek. Je možné umístit vstupní pole, kde uživatel vyplní jen svůj mail, na který bude poté upozorňován na změny, které byly provedeny. I tato služba je nabízena zdarma.

2.3 FollowThatPage

FollowThatPage [6, 9] je dalším online řešením pro detekci změn na webových stránkách. Tato aplikace se nachází na adrese http://www.followthatpage.com . Stejně jako předchozí zástupce i tato aplikace je nabízena zdarma. Umožňuje sledovat jak celou stránku, tak její jednotlivé části. Tyto části jsou zadávány pomocí počátečního a koncového klíčového slova. Zvolená sekce potom může být buď ignorována, nebo mohou být sledovány údaje jen z této části. Další možností je na základě klíčového slova odfiltrovat část webové stránky, která nemá být sledována, tento filtr bere jednotlivé řádky zdrojového kódu a vypouští ty, na kterých se vyskytuje zadané slovo. Tímto způsobem mohou být odfiltrovány například komentáře z nejrůznějších blogů.

Aplikace umožňuje sledovat až 100 stránek v jednodenním intervalu a maximálně jednu stránku v hodinovém intervalu. Jednotlivé zprávy o sledování jsou zasílány na email v textové podobě, kde je na každém řádku uvedeno, jaká informace se změnila, včetně její změny. (obrázek 2).



Obrázek 2: zobrazení změn v emailové zprávě followthatpage.com

2.4 Whatch that page

Aplikace WhatThatPage [6, 10] je k nalezení na adrese http://www.watchthatpage.com. Jejím tvůrcem je společnost ATS Consulting AS pocházející z Norska. Aplikace je šířena jako freeware a umožňuje sledování neomezeného počtu webových stránek.

Sledování jednotlivých stránek může probíhat denně nebo týdně, s tím, že je možnost zadat si hodinu a den, kdy chceme být o změnách informováni. Další možností v nastavení sledování je zadávání klíčových slov pro porovnávání změn. V případě sledování na základě klíčových slov je uživatel upozorňován, pokud se na dané stránce jím zadané slovo objevilo. Aplikace nabízí také možnost procházet jednotlivé odstavce stránky a uživateli nabídne jen ten, ve kterém se klíčové slovo vyskytuje. Je zde samozřejmě také možnost vynechat odstavce, kde se klíčové slovo nachází a tak odfiltrovat nechtěné údaje.

Podobně jako jiné online služby, tak i WhatchThatPage nabízí formulář pro webmastery jiných webů, kteří pomocí tohoto formuláře mohou nechat uživatele sledovat změny na svých stránkách.

V emailu, který služba zasílá, jsou vypsány všechny změny na stránce. Tento způsob je pro uživatele poměrně nepřehledný, protože vidí jen to, co se změnilo, ale již není vidět předchozí text.

2.5 WOKO.cz

Woko.cz [12] je českým zástupcem pro sledování změn na webu. Za touto službou stojí společnost Internet Info, která má ve svém portfoliu mnoho dalších webů a aplikací. Woko.cz se nachází na stejnojmenné adrese a poskytuje denní kontrolu webových stránek.

Změny, které se na webu vyskytnou, jsou posílány na zaregistrovaný email. Jednotlivé změny jsou detekovány na základě kontrolního součtu pro danou stránku. Výhodou tohoto systému je to, že dokáže kontrolovat také stránky v rámcích a to i vnořených. Systém si také poradí s jednoduchým přesměrováním pomocí HTTP protokolu.

Oproti předchozím zástupcům je to aplikace jednodušší, ale nenabízí některé pokročilé funkce, jako například kontrolu změn vybraných částí nebo na základě klíčových slov. Nabízí však formulář, který mohou tvůrci webu umístit na svou stránku a nechat tak uživatele prostřednictvím Woko.cz kontrolovat web.

3 Způsoby sledování změn na webu

Jednotlivé způsoby, jak sledovat změny na webových stránkách, jsou jednou z klíčových vlastností, proč se uživatel rozhodne používat právě určitou sledovací službu.

3.1 Sledování na základě klíčových slov

Velmi často se uživatel zajímá jen o konkrétní slovo nebo slovní spojení. Zajímá ho například, jestli se neobjevují nějaké novinky o jeho firmě nebo jestli se nezměnil ceník v jeho oblíbeném internetovém obchodě. Tomuto problému nejlépe odpovídá detekce na základě klíčových slov.

Tento způsob sledování se provádí tak, že jsou procházeny jednotlivé části textu, v lepším případě bloky textu, a po řádcích se kontroluje výskyt zadaného slova. Důležité je v tomto případě dobré zobrazení sledovaných informací v celkovém kontextu. Jako dobré východisko se ukazuje procházení jednotlivých odstavců textu zvlášť a jejich jednotlivá kontrola. Na konci této práce detekční program nabídne uživateli celý odstavec se zvýrazněnými klíčovými slovy.

Pomocí klíčových slov si uživatel může také specifikovat, které části webu mají být pro sledování ignorovány. Tuto možnost nabízejí již funkční webové aplikace, ale tento způsob není součástí mé aplikace.

3.2 Sledování bloků textu

Sledování celých bloků textu je způsob, který se hodí pro všechny druhy webů, ale bohužel ho podporuje jen málo současných programů (především těch provozovaných online). Využití tohoto způsobu sledování je vhodné, pokud chceme sledovat úřední desku obce, novinky ve zpravodajství, měnové kurzy, nebo například nové vědecké články.

V praxi se vyskytují, pro uživatele přívětivé, dvě metody, jak sledovat celý blok textu. Jeden ze způsobů používá systém FollowThatPage, který byl uveden v předchozí kapitole. V principu jde o to, že uživatel zadá počáteční klíčové slovo a konečné klíčové slovo a text mezi nimi je potom sledován [9]. Tato metoda je poměrně jednoduchá, ale může vést k chybám, protože html dokument je často velmi špatně strukturován, a proto mohou být údaje ve zdrojovém HTML dokumentu umístěny jinak, než je tomu při zobrazení, které vidí na stránce uživatel a podle kterého slova zadával.

Lepší možností, která je i pro uživatele nejpřehlednější a nejjasnější, je označení bloku textu, který se má sledovat, a aplikace potom sleduje pouze označený blok. Tato funkce je ovšem dostupná jen v desktopových aplikacích. Má diplomová práce se, mimo jiné, zabývá také tím, jak tuto funkci použít i pro online aplikace.

3.3 Jiné způsoby sledovaní změn

Mimo výše uvedené existuje samozřejmě mnoho jiných způsobů sledování webových stránek. Jsou to především způsoby sledující jen určité položky na webu. Uživatel může například sledovat jen odkazy na stránce, pro webové fotogalerie je zase vhodné sledovat jen změnu obrázků, případně videí.

Tyto způsoby vycházejí z procházení jednotlivých HTML tagů a kontroly jejich atributů. Dnes tyto služby poskytují především desktopové aplikace. U aplikací, které jsou dostupné online, tyto funkce většinou nenajdeme, protože nabízejí jen kontrolu textových informací.

4 Testování webových stránek na změny obsahu

Tato kapitola se zabývá postupem, který je použit pro sledování změn obsahu webových stránek. Sledování probíhá v několika krocích, které se musejí provést, aby byly uživateli nabídnuty relevantní informace o sledované části.

Cílem výsledné aplikace je podat uživateli relevantní údaje o určité části webu, která je specifikována. V aplikaci je použito několik frekvencí pro sledování. Stránky je možno sledovat hodinově, denně a týdně. Velký důraz je potom kladen na popis filtru, který udává, jaká část webu bude sledována.

4.1 Příprava dokumentu

Aby mohl být dokument testován na změny, je nutné, aby byl správně formátován. Správné formátování znamená, že dokument je buď "valid" nebo alespoň "well-formed" XML dokument. Pro naše potřeby není nutné, aby dokument obsahoval značky, které nejsou párové. Tyto značky totiž nikdy nenesou textový obsah a tak jsou pro sledování změn irelevantní.

4.1.1 Well-formed XML dokument

Well-formed jsou označovány XML dokumenty, které splňují základní pravidla formátování XML dokumentu. Pokud je dokument alespoň well-formed, je možno s ním pracovat pomocí XML parserů, které nabízí PHP a jeho knihovna DOMDocument.

Podle [16] jsou základní pravidla, aby dokument splňoval well-formed především tato. Dokument musí obsahovat XML deklaraci. To znamená, že první řádek dokumentu musí obsahovat informace o verzi XML, která je uzavřena mezi značky <? a ?>. V deklaraci se dále doporučuje uvést kódování dokumentu, ale toto již není povinné. Ukázka deklarace XML dokumentu s kódováním ISO je vidět níže.

```
<?xml version=`1.0` encoding=`ISO-8859-2`?>
```

Další důležité pravidlo se týká samotných elementů. Dokument musí obsahovat kořenový element a všechny další elementy, které obsahují data, musejí mít otevírací a uzavírací tag. I prázdné

elementy, které data neobsahují musejí být uzavřeny(`<br />`). Elementy mohou být zapisovány jen malými písmeny a jejich atributy musí být uzavřeny do uvozovek [14].

4.1.2 Validní XML dokument

Lepším případem "well-formed" XML dokumentů jsou dokumenty, které se označují jako "valid". Tyto zachovávají správnou XML syntax, ale navíc přidávají definici XML značek. Tato definice, která vnáší do XML jistý řád, je v definičním souboru DTD. Toto je vhodné, když na vývoji pracuje více lidí současně a pomocí parseru se dá jednoduše ověřit, zda-li XML dokument neporušuje zavedená pravidla. [14, 15]

V práci s XHTML není třeba, aby dokumenty splňovaly validní formu, protože význam jednotlivých značek není z hlediska sledování významný.

4.1.3 Úprava dokumentu

Příprava dokumentu probíhá postupem podle [4]. Je především potřeba dokument upravit tak, aby všechny párové značky měly otevírací a uzavírací. V tomto průchodu se dále zformátuje, aby všechny značky byly poskládány ve správném pořadí. Není totiž vhodné, aby se v dokumentu vyskytovala například tato posloupnost značek:

`<a>text1<b></a>text2<a></b>text3</a>`.

Po všech úpravách se dokument stává správně formátovaným a můžeme na něj využít XML parsery a pokročilé funkce, které XML nabízí. Jedná se především po použití jazyka XPATH.

4.2 Stromová struktura dokumentu

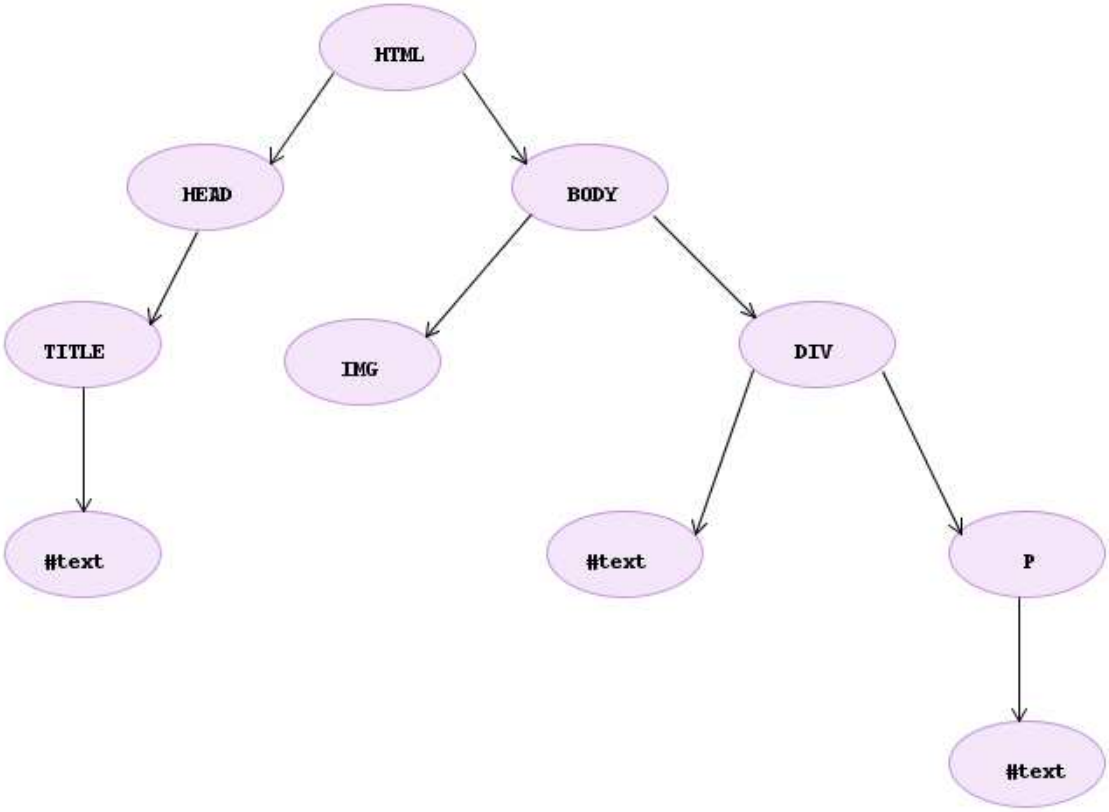
Základem pro testování změn je vytvoření stromové struktury celého dokumentu. Tato stromová struktura obsahuje v jednotlivých uzlech všechny HTML značky, které se v dokumentu nacházejí. Kořenovým uzlem je element HTML, kterým musí začínat každá webová stránka. Na tento uzel se napojují další, textové informace jsou vždy uloženy v listech.

Díky této struktuře je potom snadné se v dokumentu orientovat, je jasné, v jaké pozici se nachází sledovaná část webu. Inspirací pro vytvoření této struktury je jazyk XPATH, který je rovněž založený na podobném principu. Znamená to převést stávající HTML dokument na dokument XML. XML jazyk je založen na stromovém principu, a tak je poměrně snadné se v něm orientovat. Jakým způsobem je vytvořen strom z daného HTML dokumentu ukazuje obrázek 3.

Stromová struktura je zároveň vhodná proto, že je možné jednoznačně určit a uložit, jaký podstrom bude sledován a zaměřit se při sledování na konkrétní část webové stránky, kterou uživatel sleduje. [14]

Při validním nebo alespoň "well-formed" XML dokumentu je navíc možné specifikovat podstrom sledovaný pomocí jazyka XPATH, kterým jsme schopni jednoznačně takový podstrom určit.

Další výhodou stromové struktury je možnost použití nejrůznějších algoritmů pro průchod stromem a tak dosáhnout co největší efektivity, při vyhledávání a další prací s dokumentem.



Obrázek 3: Stromová struktura webu

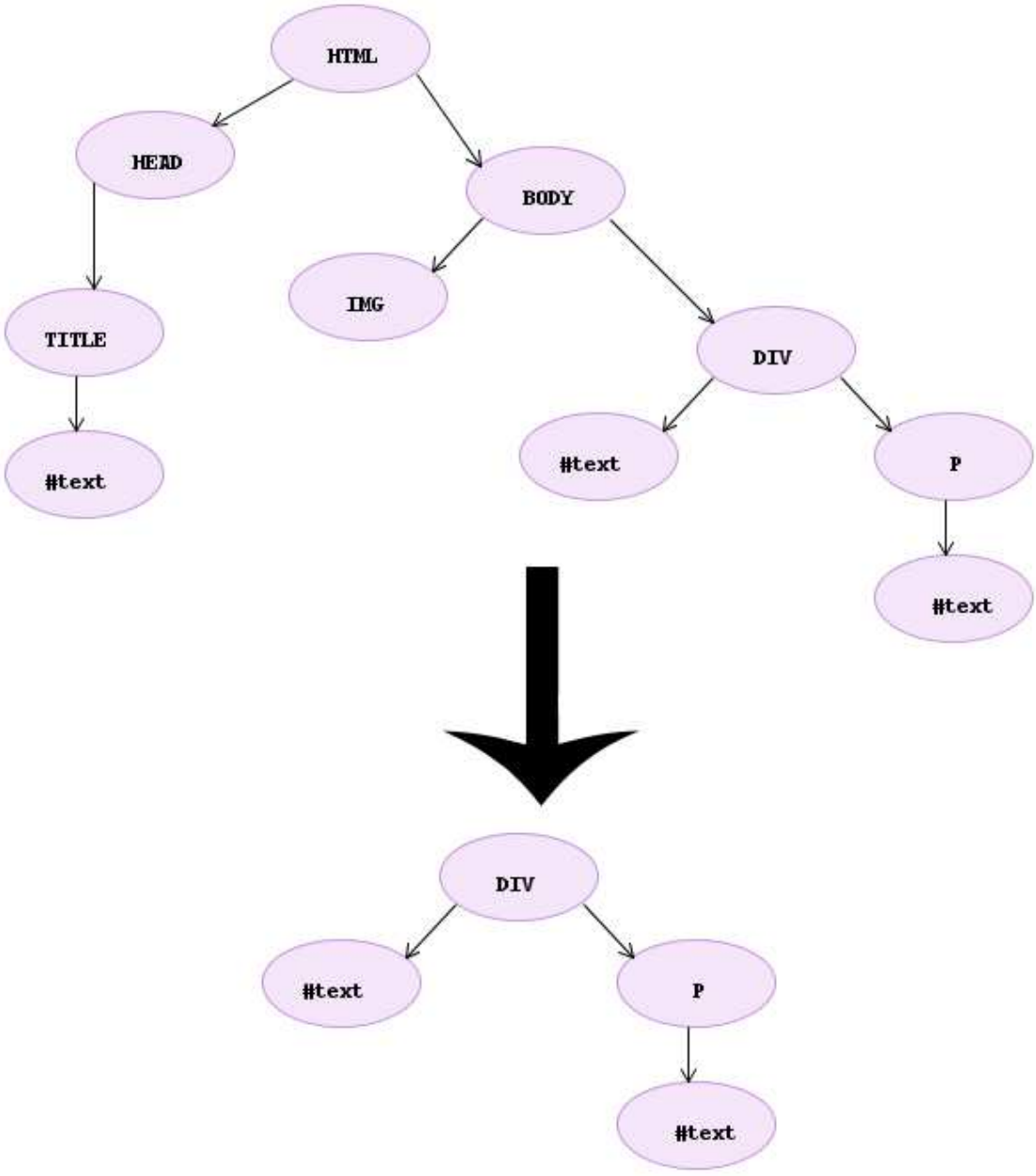
4.3 Vyhledání změn

Pro vyhledávání změn budou použity dva upravené HTML dokumenty. Starší dokument, který je uložen v databázi systému a nová verze, která je stažena přímo z webu.

V novém HTML dokumentu je nalezen podstrom, který se má sledovat, a tento je poté porovnán s referenčním podstromem, který je uložen v databázi. Nalezení podstromu v databázi je realizováno pomocí XPATH výrazu, který jednoznačně definuje daný podstrom. Pokud není uzel v nové struktuře nalezen, pokusí se program nalézt uzel pomocí alternativní cesty. Pro jeho vyhledání používá ID HTML prvku, který je ve struktuře nejblíže sledovanému uzlu. Pokud není uzel vyhledán ani pomocí alternativní cesty, je vypsána chyba, že se změnila struktura webu.

Z obou nalezených podstromů se vytáhnou textové informace, které jsou typicky uloženy v listech stromu. Tyto textové informace jsou potom srovnány pomocí knihovny pro porovnání dvou

textových dokumentů. Podstrom se změněnou částí je potom uložen do databáze a může být dále zpracováván pro prezentaci uživateli.



Obrázek 4: Strom a podstrom dokumentu

5 Popis filtrů

Jazyk pro popis filtrů umožňuje uživateli definovat, která část webové stránky ho zajímá, a kterou chce sledovat. Jazyk by měl být pokud možno co nejjednodušší a měl by být pro uživatele snadno pochopitelný, aniž by musel znát podrobnosti, například zdrojový kódu webu.

Pro svou práci jsem vybral dva způsoby, které budou popsány v této kapitole. Prvním způsobem je filtr založený na označeném textu na stránce. Druhým způsobem je jednoduchý jazyk, který má za úkol co nejpřesněji popsat tu část webu, která bude sledována.

5.1 Filtr na základě označeného textu

Inspirace pro vytvoření tohoto filtru vychází z desktopových programů, které se starají o detekci změn na webových stránkách [4]. Cílem tohoto přístupu je nabídnout uživateli co možná nejjednodušší a přitom nejpřehlednější způsob, jak definovat oblast webu, která ho zajímá.

Principem tohoto způsobu je načíst a zpracovat požadovanou stránku, která je potom uživateli předvedena ve své reálné podobě. Uživatel je pomocí myši schopen označit část textu na stránce, která ho zajímá a tuto část vybrat. Po zvolení části textu se už nemusí o nic starat.

V pozadí tohoto procesu je uživatelský javascript, který dokáže vyextrahovat zdrojový kód výběru na stránce. Tento výběr je poté porovnán s referenčním vzorkem webu a do databáze jsou uloženy informace, která část webu, respektive který podstrom ve stromové struktuře, se bude sledovat. Tento přístup je v online aplikacích poměrně nový, ale vzhledem k tomu, že nabízí pro uživatele nejlepší komfort, rozhodl jsem se ho do aplikace zakomponovat.

Jediný problém tohoto přístupu spočívá v tom, že HTML stránka, ze které uživatel vybírá, již prošla úpravou HTML dokumentu tak, jak byl popsán v kapitole 4.1. Takový dokument je tedy již umístěn na disku a tak nemusí fungovat například odkazy, nebo se nemusí správně zobrazovat obrázky, které jsou zadány pomocí relativní cesty. Častokrát jsou také odstraněny styly dokumentu a tak jeho vzhled nemusí zcela odpovídat části, která je na webu. Důležité jsou však části webu, které nesou textové informace, a ty jsou přesně zachovávány. Přehled o sledované části tak zůstává zachována poměrně dobře. Každopádně je tento způsob nejspolehlivější a uživatelsky nejpřívětivější.

Na obrázku 5 je ukázka vizuálního filtru, kdy si uživatel vybral nadpis a text článku. V tomto případě, kdy si vybral dva různé elementy, je pro sledování označen ten, který je pro oba dva rodičovský. Tento fakt může někdy vést k nepřesným výsledkům, protože nadřazený element může obsahovat ještě mnoho dalších prvků. U správně strukturovaných stránek však většinou vede k očekávanému výsledku. I tak by měl uživatel používat vizuální filtr, především pro jasně oddělená data a v případě, že si není jistý, použít filtr textový, který mu v tomto nabízí větší kontrolu. Pro

správné zaznačení uzlu není potřeba označit všechen text v uzlu, ale stačí třeba jen jedno písmeno. Stačí vytvořit jakýkoli výběr.

Obrázek 5: Ukázka vizuálního filtru

5.2 Jazyk pro popis textového filtru

Textový filtr je druhou možností pro popis sledované části webu. Pomocí textového popisu tak uživatel definuje, které věci ho na webu zajímají a které chce sledovat.

Základem pro definování filtru je výběr uzlů, které chceme sledovat a klíčové slovo, které tento uzel jednoznačně specifikuje v celé struktuře dokumentu.

Prostředek, který popis umožňuje, je imaginární jazyk, který vychází z kombinace jednoduchého *popisu umístění* sledované části a *slov pro identifikaci* této části.

5.2.1 Popis umístění

Popis umístění nabízí výčet jednotlivých částí stránky, jako jsou *nadpisy, odrážky, výčet, odstavce, zvýraznění, odkaz, tabulka a celá stránka*. Tento výběr má uživatel ve svém grafickém prostředí aplikace a může tak lépe specifikovat část webu, kde se nachází sledovaná informace.

V případě nadpisů jsou sledovány všechny standardní HTML značky pro nadpisy, jedná se tedy o značky `<h1>, <h2>, <h3>, <h4>` . Sledovaná část je potom hledána jen mezi těmito značkami.

Další části, kde mohou být změny detekovány, jsou odrážky a výčet zde se standardně jedná o obsah mezi HTML značkami `<ul>`, `<ol>`, `<dl>`.

Pokud požadujeme sledovat informaci, která je obsažena v běžném textu. Většinou se použije jako definice umístění odstavec. V tom případě jsou detekovány změny mezi značkami `<p>`.

Pod pojmem zvýraznění se myslí text, který je napsán buď tučně (značka `<b>, <strong>`), kurzívou (značka `<i>, <em>`), nebo podtrhnut (značka `<u> <ins>`), případně přeškrtnutý text (`<del>`).

Pokud chceme sledovat a detekovat změny v odkazech na stránce, je možno zvolit možnost pro sledování odkazů, kdy se sleduje text mezi značkami `<a>`.

Text se dá analogicky sledovat také v tabulkách mezi HTML elementy `<table>`.

Pro umístění, které není uživatel schopen identifikovat, je určena možnost celá stránka, kdy dochází k sledovaní a detekování změn jen na základě identifikujících slov dané části. To znamená, že popis umístění se zcela vynechává a sledovaný text se hledá v celé webové stránce. Tato možnost však může vést ke špatnému odhadnutí sledované části webu.

Jednotlivé umístění je možno řetězit tak, že se přidávají další sledované uzly. Pomocí této funkce je možno specifikovat například nadpis článku a text článku.

Princip tohoto jazyka je uživateli skrytý. Uživatel komunikuje přes webové rozhraní, kdy mu bude umožněno pomocí jednoduchého průvodce vytvořit požadovaný filtr. Jak vypadá sestavení textového filtru ve výsledné aplikaci je vidět na obrázku 5.

Obrázek 6: Vytváření textového filtru

5.2.2 Slova pro identifikaci

Když uživatel definuje umístění sledované části na stránce, je nutné ještě přidat slovo nebo slova, které tuto část identifikují. Filtr tak bude úplný a tím bude možno poměrně jasně definovat, kde na stránce se požadovaná část nachází.

Slovo pro identifikaci by mělo být zvoleno vhodně tak, aby bylo co možná nejjednoznačnější a pokud možno jedinečné. To například znamená slovo z tabulky, které jiné tabulky neobsahují. Pokud by tedy uživatel chtěl sledovat tabulku s hodnocením nějakého filmu, zadá jako umístění tabulku a klíčové slovo hodnocení.

U tabulek a odrážek je vybrána jen konkrétní odrážka, případně jen konkrétní řádek tabulky. Pokud by chtěl uživatel sledovat všechny odrážky nebo řádky tabulky, je potřeba toto definovat zvlášť. Tento způsob je zvolen především proto, že mnoho webů využívá stále tzv. tabulkový layout, kterým je celá stránka designována a tak by filtr nemusel vracet relevantní výsledky.

S vybraným uzlem je pracováno jako s podstromem, kdy vybraný uzel je kořenovým pro tento podstrom. Proto nevadí, když uživatel vybere nadpis, který obsahuje vnořený odkaz. Sleduje se pak text obsažený v tomto odkazu.

5.2.3 Ukázka filtru

Tato ukázka obsahuje filtr, který definuje sledování úřední desky obce. Úřední deska obce je vytvořena pomocí odrážek, kdy seznam obsahuje stále stejný počet položek a zatím, co je nová položka na desku přidána, je ta nejstarší zase odebrána.

Předpokládejme následující úřední desku:

- 22. 12. Zasedání zastupitelstva
- 10. 11. Změna úředních hodin
- 21. 10. Zasedaní zastupitelstva
- 14. 10. Zahájení provozu nové čističky
- 22. 9. Zasedání zastupitelstva

Ideálním způsobem jak uživatel může definovat, že chce sledovat právě tuto úřední desku, je pro umístění zvolit volbu odrážek a jako slovo pro identifikaci například slovo *zastupitelstva*. Pseudo jazyk filtru, který takovou událost popisuje, by potom mohl vypadat následovně.

```
Filtr(blok) = Place(odrazky) AND Word('zastupitelstva')
```

Jazykem filtru však uživatel filtr nezapisuje, pouze využívá webového rozhraní, které se o toto postará. Tento pseudo jazyk je převeden na XPATH výraz a dále se už pracuje jen s tímto výrazem.

Pokud je uživatel zběhlý v jazyku XPATH, může uzel identifikovat také pomocí XPATH výrazu. Tento způsob je nejjednoznačnější, ale zase klade jistou náročnost pro uživatele, protože musí otevřít zdrojový kód dokumentu a sestavit sám XPATH výraz.

6 Popis šablon pro informování

Celá aplikace je velmi závislá na tom, jak se uživateli prezentují výsledky sledování změn. Prezentace těchto výsledků je důležitá, protože program by neplnil svou funkci, kdyby výsledky nebyly hned jasně patrné. Pro informování uživatele je možno zvolit dva způsoby. Prvním způsobem je klasické informovaní na zadaný email, kdy uživatel dostává v textové podobě, co se změnilo a na co. Druhým způsobem je modernější informování pomocí RSS (viz kapitola 8.1.5) kanálů, které odpovídají jednotlivým sledovaným stránkám.

6.1 Informování pomocí emailu

Informování pomocí emailu je jedním z nejčastějších způsobů pro zasílání změn. Tento způsob komunikace s uživatelem je poměrně jednoduchý, ale má i jisté zápory. Mezi záporné vlastnosti jistě patří fakt, že uživatel je tak velmi často „bombardován“ emailovými zprávami, což může být do jisté míry nepříjemné. Přesto je tento způsob velmi populární, především kvůli své jednoduchosti.

V mé aplikaci nebude použito informování pomocí emailu, ale pouze modernější informování pomocí RSS. Tento způsob totiž disponuje lepšími možnostmi pro formátování textu a zároveň je mezi uživateli oblíben, protože nejsou jejich emailové schránky zahlcovány emaily se změnami. Je však možno tento způsob informování jednoduše doplnit, protože každý uživatel registrovaný v systému musí zadat kontaktní email. Je tedy možno o tomto způsobu informování uvažovat při rozšíření aplikace.

6.2 Informování pomocí RSS

Informování pomocí RSS je velmi pohodlné, protože je jasně strukturované a jednotlivé změny mohou být od sebe dobře a jasně odlišeny a přitom je vidět vývoj, jak se daná stránka měnila. Pro každou sledovanou stránku bude vytvořen samostatný RSS kanál. Stejně tak je možno odebírat RSS jen jednoho filtru a tak si rozdělit sledovanou stránku na několik RSS zdrojů.

RSS kanál (`<channel>`) obsahuje popis sledované stránky a jednotlivé položky změny, které se na webu provedly. Pro každý kanál tak jednotlivé položky přibývají podle toho, co se na stránce změnilo. Tento systém je vhodný především pro sledování kratšího textu, kde není zapotřebí uvádět v RSS dlouhé informace o změnách. Pokud by se jednalo o sledování celé stránky, nebo velké části textu, je tento text zkrácen a uživateli je prezentována jen ukázka. Celý text si potom může prohlédnout po přihlášení do aplikace.

Každá položka, v RSS kanálu pod značkou `<item>`, obsahuje čas, kdy byla stránka kontrolována a nový text, který se změnil. Nový text je zvýrazněn tak, aby měl uživatel jasný přehled o změnách jen z RSS zdroje. Pokud se jedná o detekci změn v bloku textu, je v RSS zobrazen celý tento blok. Toto vše je doplněno adresou na změněnou stránku.

Uživatel, který se rozhodne změny kontrolovat pomocí RSS kanálu, tak dostává jasný přehled o tom, co se na stránkách změnilo a kdy se tak stalo. Není zahlcován emaily ve své schránce a pomocí RSS čtečky tak může jednotlivé změny sledovat. V tabulce níže je uvedena struktura RSS kanálu, který obsahuje informace o změnách. Příklad RSS je sestaven podle [11].

```
<?xml version="1.0" encoding="utf-8"?>

<rss version="2.0" xmlns:atom="http://www.w3.org/2005/Atom">
   <channel>
     <title>Titulek kanálu</title>
     <link>http://www.domena.cz</link>
     <copyright>David Blaheta 2010</copyright>
     <pubDate>
          2010-05-10 20:40:53
     </pubDate>
     <lastBuildDate>
          2010-05-10 20:40:53
     </lastBuildDate>
     <description>
          Popis kanálu
     </description>
     <item>
       <title>
          Titulek položky
       </title>
       <description>
          Popisek položky
       </description>
       <pubDate>
          Datum publikace
       </pubDate>
    </item>
</channel>
<rss>
```

7 Návrh systému

V této kapitole bude nastíněn základní návrh celého systému s rozborem jednotlivých částí. Aplikace bude komponovaná jako klasická klient – server aplikace.

7.1 Klient – server aplikace

Pro tvorbu online informačního systému pro sledování a analýzu změn na webu je model klient – server jednoznačně nejvhodnější.

Klientská část aplikace obsahuje především rozhraní pro komunikaci s uživatelem. Je to prostředí klasické webové stránky, kde uživatel po zaregistrování a přihlášení může zadávat informace o sledovaných stránkách, přidávat je, rušit, editovat a používat další funkce. Jedná se o tzv. tenkého klienta, který slouží pouze ke komunikaci se serverem, o všechny funkce aplikace se tak stará serverová část.

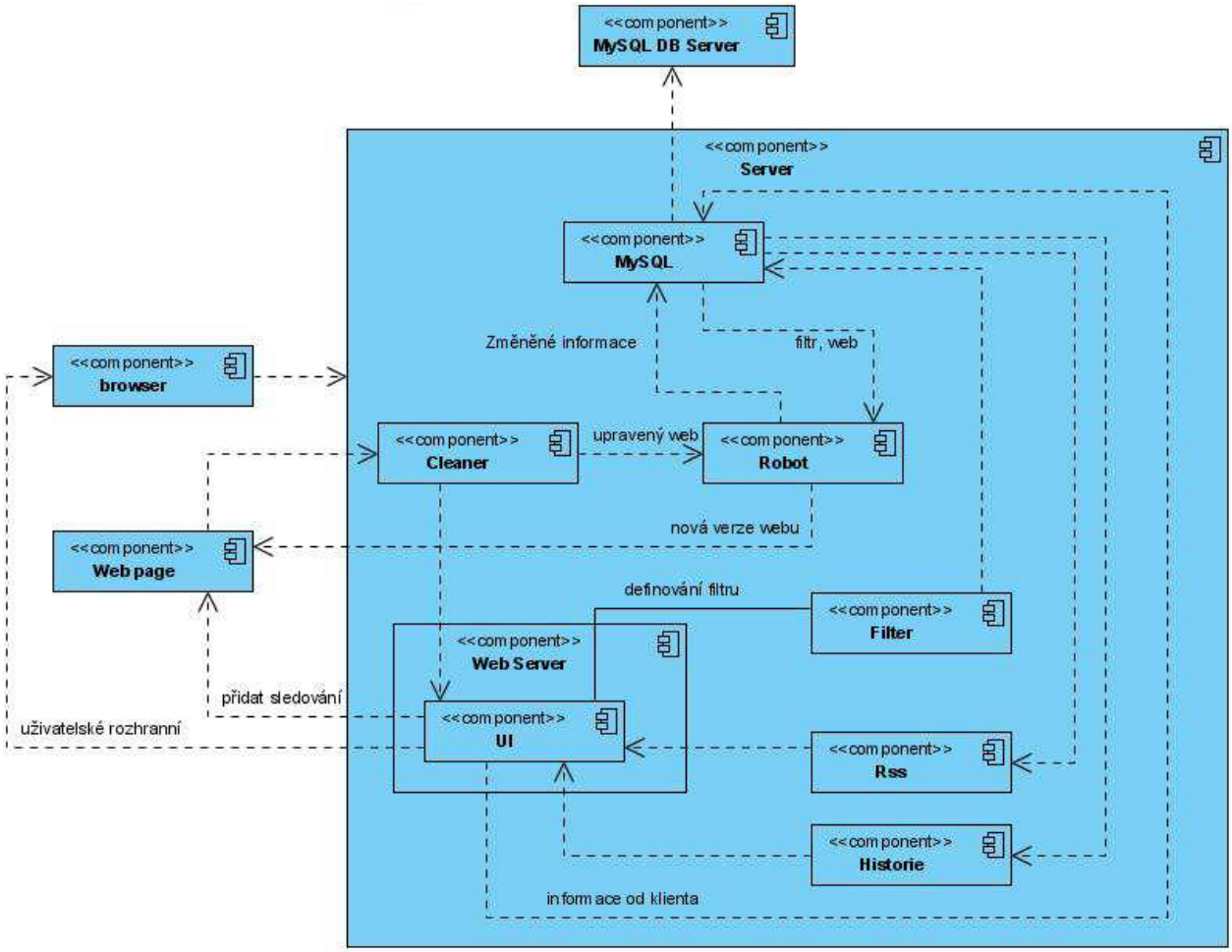
Serverová část aplikace potom obsahuje jednotlivé funkce, které se starají o samotné sledování a detekci změn. V této části jsou veškeré funkční části, které jsou potřebné pro správný chod celé aplikace. Serverová část je uživateli zcela skryta a nepřichází s ní do kontaktu.

Aplikace se skládá z klientské části, což jsou jednotlivé HTML stránky uživatelského rozhrání generovány podle pokynů uživatele.

Další důležitou součástí je webový server, který se stará o interpretaci PHP skriptů. Tyto skripty generují prvky uživatelského rozhraní. Skripty přistupují do databáze a z ní získávají data, které jsou prezentovány uživateli.

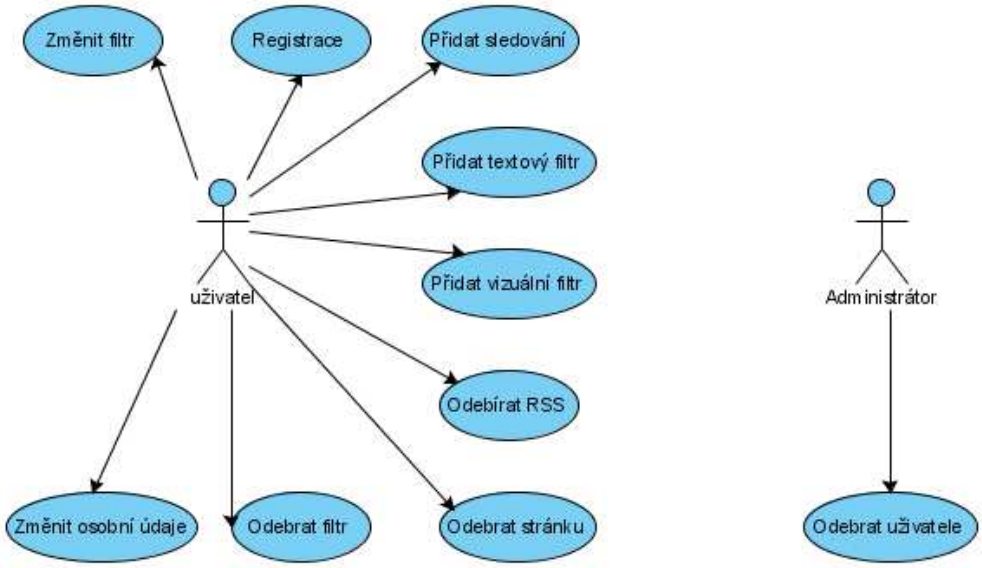
Hlavní funkční část se odehrává na serveru a opět se skládá z několika součástí. K databázi se přistupuje přes rozhraní a umožňuje veškerou práci s databází. Na obrázku je tato komponenta označena jako *MySQL*. Jedná se o několik PHP skriptů, které přistupují k databázi a starají se veškerou práci s daty v databázi, a také data do databáze ukládají. Komponenta *Cleaner* je opět složena z PHP skriptu a upravuje staženou webovou stránku tak, aby odpovídala validní formě. Komponenta *Filtr* implementuje vytváření uživatelských filtrů. V této komponentě se vytvářejí jak textové tak vizuální filtry, které jsou potom ukládány do databáze. K této komponentě také přeneseně přistupuje uživatel přes uživatelské rozhraní, kde filtry definuje. Komponenta *Rss* se stará o formátování dat pro RSS prezentaci výsledků. Komponenta *Historie* přistupuje k databázi, kde se ukládají již navštívené stránky a vytváří se tak archiv jednotlivých sledování. Přistupuje do uživatelského rozhraní, kde je poté generováno zobrazení v celém kontextu stránky. Jednou z nejdůležitějších částí na serveru je komponenta *Robot*. Tato komponenta spolupracuje s několika skripty a stará se o stahování stránek podle filtrů, porovnávání filtrovaných částí a ukládání výsledků do databáze.

Poslední částí architektury je databázový server, který uchovává veškerá potřebná data.



Obrázek 7: Architektura aplikace

Funkčnost a rozsah celé aplikace je dále ukázán na *use case* diagramu, který ukazuje základní možnosti, které výsledná aplikace nabízí. Systém je rozdělen na uživatelskou a administrátorskou část. Uživatel může po registraci provádět všechny úkony týkající se sledování webových stránek. Administrátor potom může spravovat jednotlivé uživatelské účty a přes tyto účty přistupovat do systému. Má tak veškerá práva nad systémem.



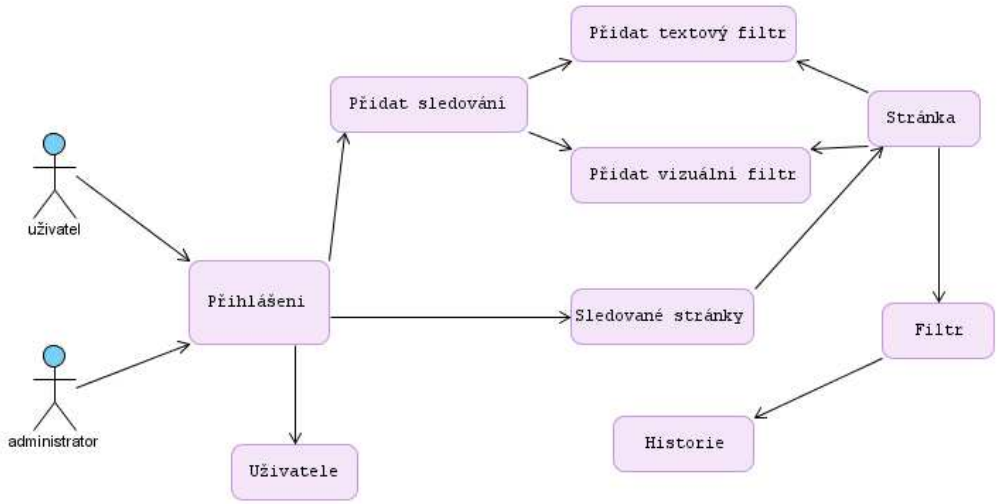
Obrázek 8: Use Case diagram

7.2 Klientská část

7.2.1 Uživatelské rozhraní

Uživatelské rozhraní je klasická webová aplikace, která je napsána v jazyce XHTML. Hlavním cílem rozhraní je jednoduchost a přehlednost pro uživatele, který by měl na první pohled jasně pochopit, které části webu jsou pro něj důležité.

Jak vypadá struktura klientské části webu je vidět na diagramu, který ukazuje posloupnost obrazovek na Obrázek 9.



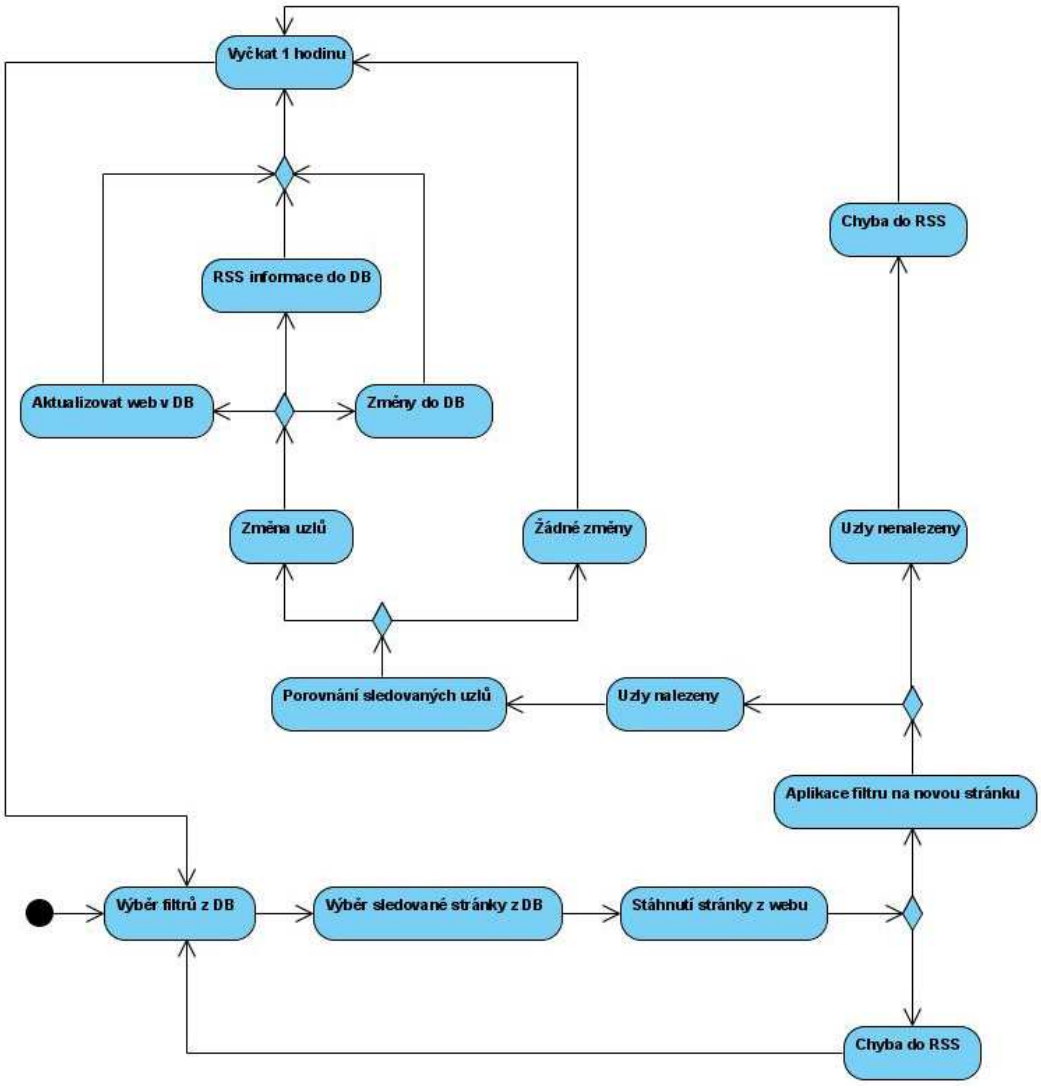
Obrázek 9: Posloupnost obrazovek

7.3 Serverová část

7.3.1 Činnost serveru

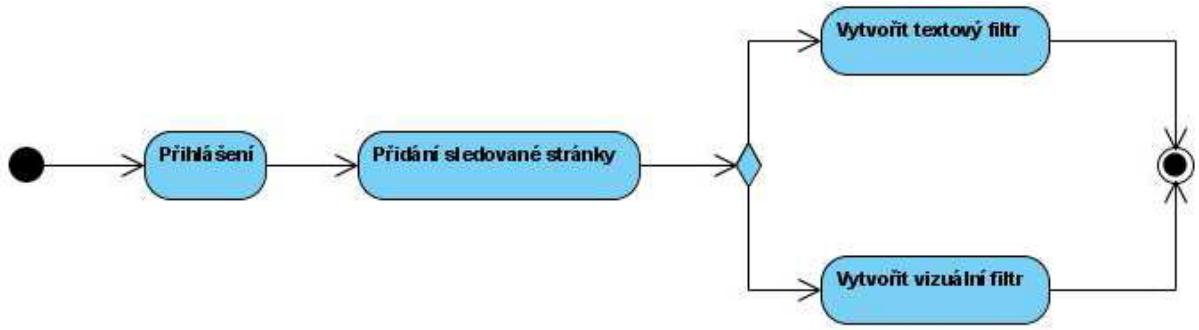
Serverová část aplikace je složena z několika modulů, které spolupracují a ve výsledku předávají konečné informace klientovi.

Činnost serveru se skládá z hlavních dvou částí. První část je nezávislá na uživateli a pracuje zcela samostatně. Základem této části je robot, který se stará o návštěvy webu a jejich porovnávání s referenčními vzorky. Data stažená robotem jsou zformátována do XHTML podoby. Na základě informací z filtru je potom vybrán uzel v HTML stromu, který se bude sledovat a tento uzel je porovnán s referenčními daty. Pokud byly nalezeny změny, jsou uloženy do databáze tak, aby je mohla klientská část aplikace nabídnout uživateli. Lepší přehled o činnosti této části aplikace dává Obrázek 10.



Obrázek 10: architektura serveru

Druhá část serveru je již závislá na zásahu uživatele z venčí. Tato část se stará o veškeré vykreslení prvků grafického rozhranní a také komunikaci s uživatelem. Uživatel se musí do systému přihlásit a potom může přidávat stránky pro sledování a filtry, které definují sledované části. Diagram aktivit pro tuto část se nachází na Obrázek 11Obrázek 11.

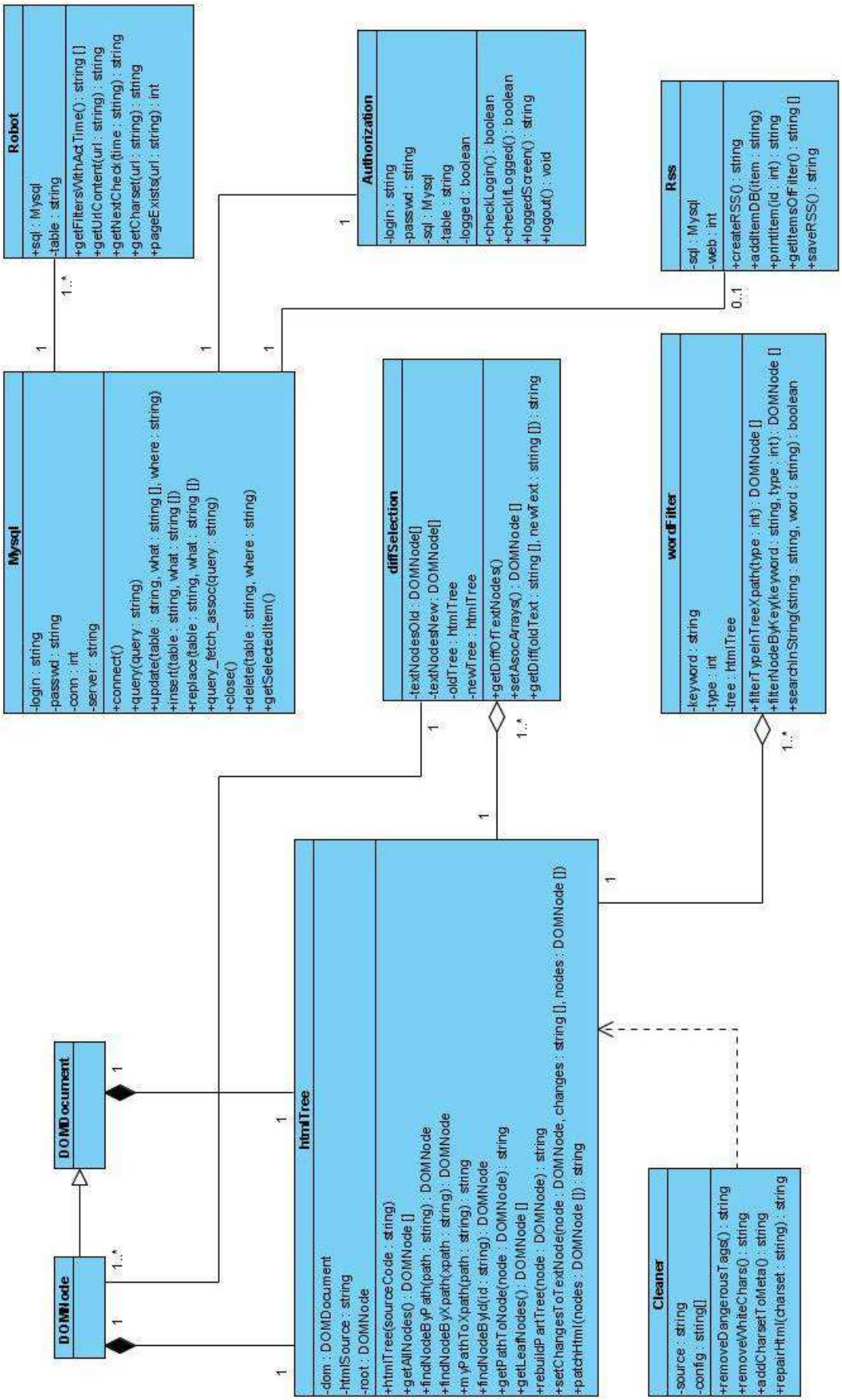


Obrázek 11: činnost serveru pro zadávání sledování

7.3.2 Diagram tříd

Pro implementaci je zvolen objektově orientovaný přístup, který umožňuje lepší orientaci v kódu, ale hlavně nabízí dobré možnosti, jak rozdělit aplikaci na jednotlivé funkční části. Pro každou základní část aplikace je tak vytvořena třída, která obsahuje všechny potřebné metody.

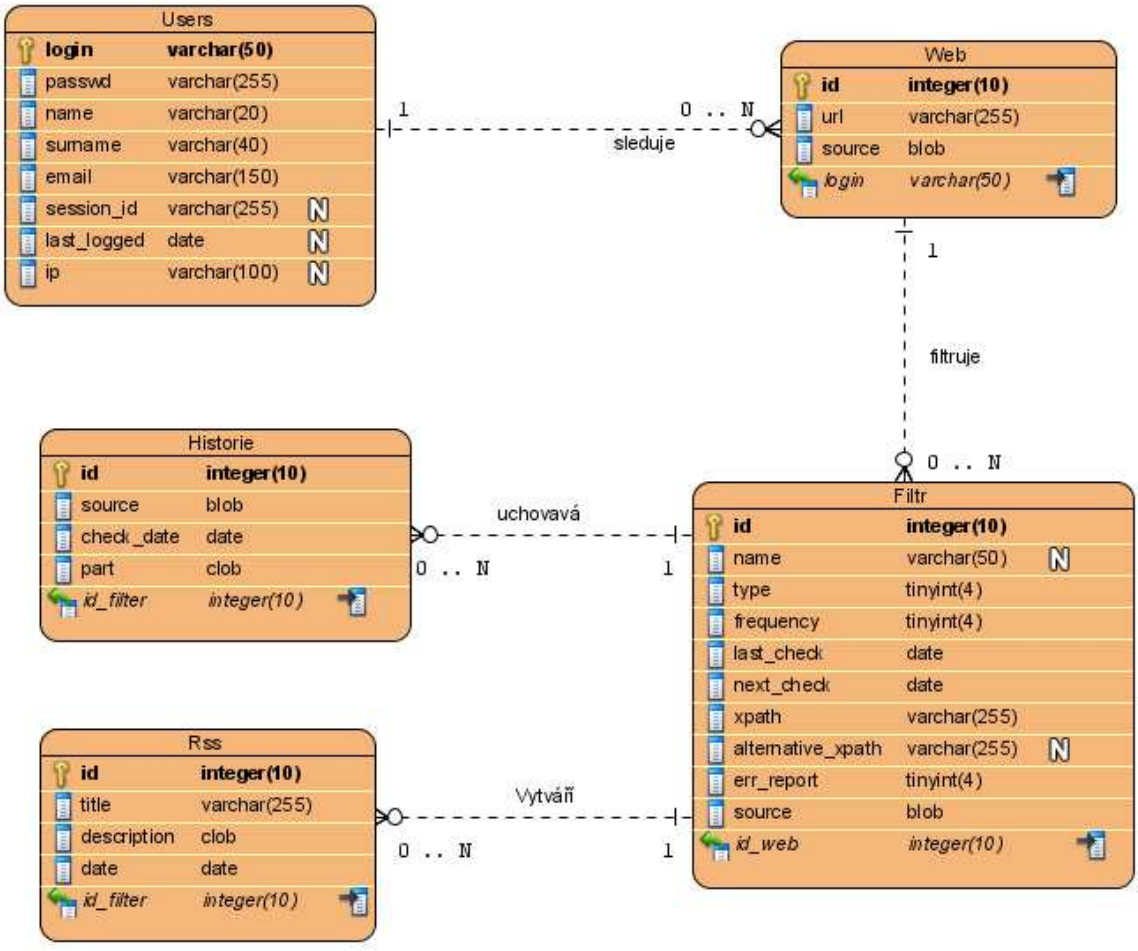
Základem pro práci s HTML dokumentem je třída htmlTree, která se stará o poskládání HTML dokumentu do stromu a nabízí důležité metody pro práci se stromem. Pro úpravu HTML dokumentu do validní podoby, která je důležitá pro sestavení stromu, se stará třída cleaner. Pro textový filtr slouží třída wordFilter, která úzce spolupracuje se třídou diffSelection, která se stará o najití a zobrazení změn v HTML dokumentu.



Obrázek 12: Diagram tříd

7.3.3 Návrh databáze

Databáze je základním prvkem pro ukládání dat v celém systému. Obsahuje jak údaje o jednotlivých uživatelích, tak o jednotlivých stránkách a filtrech. Stránky jsou ukládány jako textové pole do tabulky webu a zároveň se každá nová verze ukládá k danému filtru. Filtr je také ukládán do databáze a jednomu webu může být přiřazen libovolný počet filtrů. V tabulce filtr se ukládají především informace o umístění sledovaného místa, které je specifikováno XPATH výrazem a alternativním XPATH výrazem. Dále je ve filtru uvedena poslední návštěva stránky a naplánován čas další návštěvy. Každý filtr obsahuje zdrojový kód stránky, který se aktualizuje pokaždé, když se filtrovaná část webu změnila. Veškeré změny stránek jsou archivovány v tabulce historie. V historii se ukládá nejen celý zdrojový kód stránky, ale také jen změněná část. Změněné části, které se zobrazují v RSS kanálu daného filtru nebo webu, jsou uloženy v tabulce RSS. Z ní může být jednoduše sestaven kanál pro daný filtr a pro daný web.



Obrázek 13: ER diagram databáze

8 Realizace

8.1 Použité technologie

8.1.1 PHP

PHP je skriptovací programovací jazyk. Je to jazyk interpretovaný, což znamená, že se vyhodnocuje až za běhu a nepřevádí se do podoby binární spustitelné aplikace. PHP běží na straně serveru a k prezentaci ve webových aplikacích potřebuje webový server. Syntaxe jazyka kombinuje hned několik programovacích jazyků (Perl, C, Pascal a Java). PHP je nezávislý na platformě, skripty fungují bez úprav na mnoha různých operačních systémech.

Velkou výhodou PHP a také jeden z hlavních důvodů jeho značné oblíbenosti, je jeho velký seznam všech možných funkcí. PHP obsahuje velké množství funkcí pro práci se soubory, grafikou, přístup k velkému počtu databází (např. MySQL, PostgreSQL aj.), zpracování textových dat, hojně jsou také zastoupeny funkce pro práci s různými protokoly, které se na internetu vyskytují. Ať už jsou to protoky pro odesílání elektronické pošty (*SMTP*, *IMAP*, *POP3*), protokol http a další. Funkcí usnadňujících práci vývojářům je v PHP celá řada. Ty jsou buď standardně obsaženy už v jádru PHP, ale mnoho z nich je volně dostupná jako jakési rozšíření ve formě dynamických knihoven (*extension*).

S verzí pět se také výrazně zlepšil postoj k objektově orientovanému programování a PHP se v tomto směru trochu přiblížilo jazyku Java.

Syntaxe jazyka PHP je velmi podobná jazyku C, ale na rozdíl od něj nabízí programátorovi výrazně větší svobodu při psaní programů.

Nespornou výhodou je možnost začlenit výrazy jazyka přímo do HTML kódu aplikace a tak měnit jednotlivé části webu. Web se tak stává dynamický.

Mezi typické vlastnosti jazyka patří to, že je dynamicky typový. To znamená, že datový typ proměnné se určí v době přiřazení hodnoty.

Celé jádro mé aplikace je napsáno v PHP a tak bylo možno využít všech výhod tohoto jazyka a to od použití nejrůznějších knihoven až po přenositelnost na různé platformy. [22]

8.1.2 MySQL

MySQL je databáze pro mnoho platforem. Komunikaci zajišťuje dotazovací jazyk SQL. SQL je standardizovaný jazyk, vytvořený pro práci s daty v relačních databázích. Jazyk SQL je v této databázi používán ještě s několika specifickými rozšířeními. Základ vychází ze standardního jazyka SQL.

Hlavní výhody MySQL je její značná rychlost. Už od počátku se tato databáze snažila orientovat na rychlost zpracování dat. Proto je použitelná i pro vývoj složitějších systémů. Oblibu si získala také díky jednoduché přenositelnosti přes různé platformy. Díky tomu byla její podpora integrována do jazyka PHP a společně se serverem *Apache* tvoří základ většiny dnešních webových aplikací.

Syntaxe jazyka pro komunikaci vychází ze standardního dotazovacího jazyka SQL. [22]

8.1.3 JavaScript

Další nástroj, který byl použit v mé práci, je jazyk JavaScript. Tímto jazykem je vytvořen vizuální filtr a další pomocné prvky uživatelského rozhranní.

Jazyk JavaScript je stejně jako PHP skriptovacím jazykem a z toho plynou jisté podobné rysy. Zásadně se tyto jazyky ale liší tím, kde se zpracovávají. Zatímco PHP se zpracovává na straně serveru, JavaScript se zpracovává na straně klienta. Interpretuje ho přímo webový prohlížeč. Je zřejmé, že jazyk JavaScript je odvozen od jazyka Java firmy *Sun*, ale pravda je, že tyto jazyky mají jen podobnou syntaxi. Jinak se od sebe výrazně odlišují.

JavaScript není jazyk příliš univerzální. Je použitelný pouze pro vývoj webových aplikací. Zapisuje se přímo do HTML dokumentu nebo může být uložen v externím souboru. Je spustitelný pouze přes prohlížeč. Je to jazyk objektový. Využívá objektů prohlížeče a zabudovaných objektů (např. objekt `window`). Funguje pouze v prohlížeči, z čehož pramení některé nevýhody. Uživatel může spouštění JavaScriptu zakázat a poté některé aplikace nefungují správně. Různé prohlížeče JavaScript různě interpretují, což může vést k chybám.

I přes tyto nevýhody má ale JavaScript své místo v internetových prezentacích. Díky své jednoduchosti a pohodlnosti zápisu je velmi často využíván k vytvoření jednoduchých aplikací, které ale mohou uživateli stránek značně pomoci. [22]

8.1.4 XHTML

Podle [22] *HyperText Markup Language* (dále jen HTML) vznikl roku 1990 na půdě výzkumného centra CERN. Za jeho vznikem stáli především **Tim Berners-Lee** a **Robert Caillau**. Byl původně vytvořen pro popis dokumentů, v té době převážně vědeckých. A byl navržen tak, aby byl jednodušší než TeX, postscrip a jiné jazyky, které se v té době hojně využívali. Ve stejné době a na stejném místě také vznikl protokol, který umožňoval jeho přenos v síti a to HTTP. V roce 1993, po vzniku prohlížeče *Mosaic*, což byl první prohlížeč HTML s grafickým uživatelským rozhranním, začala popularita HTML rychle růst. Od té doby se jazyk HTML stále vyvíjí a nabízí stále nové věci, čímž se přizpůsoboval potřebám dnešního moderního publikování na internetu.

Jazyk HTML je značkovací jazyk, to znamená, že jeho zdrojový kód obsahuje zároveň jak vlastní text, tak instrukce pro jeho zpracování. Ty se zpracovávají pomocí značek (*tags*). Zdrojový text je ASCII soubor, takže je snadno editovatelný i v jednoduchých textových editorech.

HTML, jak je známe dnes, je HTML verze 2.0. Je charakterizován množinou značek, které mají definované určité atributy. Mezi tyto značky je umístěn text, který určuje význam daného dokumentu. Značky jsou většinou párové, kdy rozlišuje počáteční a konečnou značku. Ta začíná znakem lomítko. Značky jsou uzavřeny mezi úhlové závorky. Jednotlivé značky se do sebe dají vnořovat a tím vzniká struktura HTML dokumentu. Příklad dokumentu zapsaného pomocí jazyka HTML by vypadal takto.

```
<html>
     <body>
          <center><h1>Nadpis</h1></center>
     </body>
</html>
```

Značky HTML dokumentu můžeme rozlišit na tři základní skupiny.

1. **Strukturální značky:** rozvrhují strukturu dokumentu (např. odstavce, nadpisy)
2. **Popisné značky:** popisují povahu elementu (např. `<title>`)
3. **Stylistické značky:** utvářejí vzhled dokumentu

Pro zobrazení HTML dokumentu slouží prohlížeč. Ten je vybaven *parserem,* který dokument rozloží na jednotlivé elementy. Každý prohlížeč obsahuje tabulku značek, které podporuje. Tuto tabulku můžeme omezit zapsáním typu HTML dokumentu na začátku zdrojového kódu. Každé značce je poté přiřazen styl. Ten definuje, jak bude text mezi dvěmi značkami vypadat.

Vývoj jazyka HTML je již nyní zastaven. Byl ukončen verzí 4.01 a dalším vývojem pro tvorbu prezentací je jazyk XHTML.

Jazyk XHTML je nástupce jazyka HTML verze 4.01. Jedná se také o značkovací jazyk a vychází ze stejných standardů. XHTML konkrétně ve verzi 1.0 byl použit pro tvorbu mé bakalářské práce.

Byl vyvinut tak, aby vyhovoval podmínkám tvorby XML dokumentů a přitom, aby zůstala zachována kompatibilita s jazykem HTML. Existuje ve třech verzích s*trict*, t*ransitional* a *frameset*. Z faktu, že se jedná vlastně o jazyk XML, vyplývají některé zákonitosti, které je třeba dodržovat. Jedná se hlavně o uzavírání HTML tagů. V XHTML musí být každý tag uzavřen, včetně nepárových, které se musí ukončovat zpětným lomítkem. Dalším typickým rysem je to, že atributy tagů musí být psány do uvozovek a všechny tagy musí být psány malými písmeny.

8.1.5 RSS

RSS je zkratka, která znamená Rich Site Summary. Tato technologie se v dnešní době, kdy se na webu vyskytuje velké množství informací, stále rozšiřuje a je používána na stále více webových stránkách.

Původní funkce RSS byla komunikace mezi jednotlivými weby, které si tak mohli předávat informace o aktuálně zveřejněných článcích. S nárůstem informací se ale RSS začalo prosazovat i mezi obyčejné uživatele. Dnes už je jen málo lidí, kteří žádný RSS kanál nesledují.

Hlavní motivací pro vznik RSS bylo usnadnění prohlížení především zpravodajských webů. Výhodou je totiž především to, že aniž by se uživatel musel někde registrovat, je jednoduchou formou informován o novinkách. To vše probíhá, aniž by uživatel musel stránky navštívit. Dnes je RSS standardem u všech zpravodajských webů, kde bývá většinou množství různých kanálů pro každou rubriku.

8.1.5.1 Historie

Podle [1] byla první verze formátu RSS představena v roce 1999 společností Netscape pro použití na portálu My.Netscape.com. Tato verze nesla označení 0.9 a nabízela jen velmi málo možností. Dnes už se s verzí 0.9 nesetkáme.

Později v roce 1999 byla vydána nová specifikace RSS, která nesla označení 0.91 a tato verze se stala první masivně používanou. Můžeme se s ní setkat i dnes, je podporována prakticky všemi RSS čtečkami. Úspěch této verze RSS pro jednoduché zobrazování novinek byl veliký a tak se tato technologie začala z portálu firmy Netscape rozšiřovat dále po webu. Později se vyvíjeli další verze, které vždy vycházeli z verze 0.91 a přinášeli jen drobné změny.

Větší změna nastala v roce 2002, kdy byla předvedena verze 2.0. Od této doby je RSS hojně využíváno po celém webu. Důvodem je především to, že tento formát již vyhovuje moderním potřebám tvůrců webu a určitě také ustálená specifikace.

8.1.5.2 Technologie

RSS je založeno na jazyku XML, což znamená, že je velmi univerzální. Výhodou XML je také podobnost k jazyku HTML a jednoduchá zpracovatelnost v programovacích jazycích. RSS má předepsanou XML strukturu, která je dána specifikací a autor by se od ní neměl odchýlit.

Každý RSS kanál se tak skládá z několika po sobě jdoucích XML elementů, které označují například autora článku, název, datum publikace a další důležité informace. Tyto jednotlivé XML dokumenty jsou poté zpracovávány pomocí RSS čteček, které XML převádějí do uživatelsky přívětivé podoby. Ukázka zdrojového kódu RSS kanálu podle [2, 3, 11] je uvedena níže.

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">

<channel>
<title>RSS Priklad</title>
<description>popis prikladu</description>
<link>http://www.domenaprikladu.cz</link>
<lastBuildDate>Mon, 28 Aug 2009 11:12:55 -0400 </lastBuildDate>
<pubDate>Tue, 29 Aug 2009 09:00:00 -0400</pubDate>

<item>
<title>Polozka prikladu</title>
<description>telo clanku</description>
<link> http://www.domenaprikladu.cz </link>
<guid isPermaLink="false"> 1102345</guid>
<pubDate>Tue, 29 Aug 2009 09:00:00 -0400</pubDate>
</item>

</channel>
</rss>
```

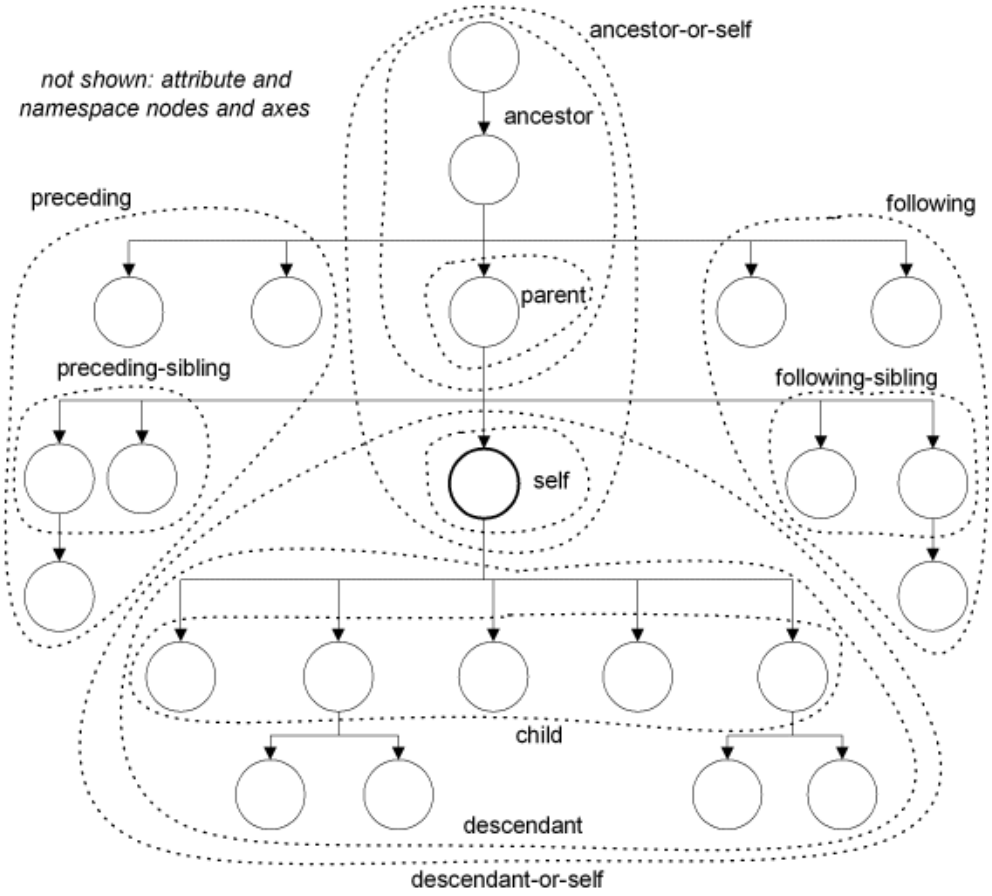
8.1.6 XPATH

XPATH je jazyk, který slouží k adresování částí HTML dokumentů. Pomocí jazyka XPATH jsme schopni vyjádřit relativní cestu od jednoho uzlu ke druhému. Možnost XPATH jsou však mnohem větší.

XPATH se využívá při práci s XML velmi hojně, vlastně všude, kde je třeba vyhledávat ve struktuře XML dokumentu určité data. Chceme-li vybrat určitou množinu dat, odpovídajících zadaným podmínkám, obvykle nám postačuje jediný XPATH výraz a XML parser se o prohledání dokumentu a vyhodnocení podmínek postará sám. Takovéto vyhledávání dat je výborně využitelné také v XSL transformacích [17].

Jazyk XPATH je standardem vydaným organizací W3C. Základní součástí jazyka je *path expression*, „výraz popisující cestu“. Taková cesta se zapisuje jako posloupnost přechodů mezi jednotlivými sadami uzlů oddělených lomítky. Nejjednodušší zápis se skládá jen z názvů elementů oddělených lomítky. Možno je využívat i složitější konstrukce, uzly vyhledávat podle atributů nebo například podle os s prvky (předchůdci, následovníci).

XPATH je založen na procházení stromové struktury XML dokumentu. Na Obrázek 14 je uvedena stromová struktura s vyznačením jednotlivých části podstromu. Pomocí těchto částí se dá také popsat cesta k danému uzlu.



Obrázek 14: Strom dokumentu, převzato z [20]

8.2 Použité standardní knihovny

V mé práci byly použité některé standardní knihovny, které nabízí jazyk PHP. Pro vytváření HTML stromu byla využita knihovna DOM. Pro stahování zdrojových kódů byl použit balíček CURL a pro převod HTML dokumentů na well-formed XHTML dokumenty byla použita knihovna TIDY.

8.2.1 DOM

DOM je označení pro *document object model*. Jedná se o standardní API, které nabízí PHP od verze 5. Je to rozhraní pro práci s dokumenty XML definované konsorciem W3C. Rozhraní definuje způsob, jakým se dokument XML mapuje na hierarchii objektů v paměti [18].

Každé části dokumentu, jako je element, atribut, textový uzel a podobně, odpovídá v paměti jeden objekt. Pomocí metod a vlastností dostupných na každém objektu můžeme zjišťovat druh uzlu, jaký ve stromu dokumentu XML zastupují, jejich název, obsah, seznam objektů reprezentujících dětské uzly, objekt zastupující rodiče uzlu atd.

Základem knihovny je třída DOMDocument. Tato třída popisuje XML dokument jako celek a stará se o načtení, práci a ukládání dokumentu. Veškerá práce s HTML dokumenty používá s tuto třídu. DOMDocument pracuje s kódováním UTF-8, a proto jsou veškerá data, se kterými se v aplikaci pracuje, převáděna na toto kódování.

Rozhraní je vytvořeno hierarchickým způsobem a tak všechny načtené elementy jsou potomky třídy DOMNode. Tato třída jich nabízí velký počet pro práci s uzly. Obsahuje například informace o jménech uzlů, jejich dětí a rodičů. Pomocí této třídy tak může být sestavena stromová struktura.

Velkou výhodou rozhraní DOM je také interpret jazyka XPATH, který je v aplikaci využíván pro hledání sledovaných uzlů.

8.2.2 CURL

Podle [21] je CURL knihovna, která je vytvořena pro práci pro komunikaci se servery pomocí různých protokolů. V současné době jsou pro komunikaci podporovány protokoly https, ftp, gopher, telnet, dict, file, a ldap protokoly.

Rozšíření CURL je v PHP dostupné od verze 4 a je založeno na knihovně *libcurl* vytvořené Danielem Stenbergem.

V aplikaci je CURL použito pro stahování zdrojových kódu ze zadaných adres, ale rozšíření CURL nabízí daleko více funkcí.

8.2.3 TIDY

Knihovna TIDY je potřebná pro převod HTML dokumentů na validní XHTML dokumenty. S touto knihovnou se setkáváme v celé řadě programovacích jazyků. Je to utilita, která nám umožňuje opravit nevalidní dokumenty [19].

Knihovna se vyskytuje v PHP jako volitelná součást od PHP verze 4. Od té doby prošla jistým vývojem a v PHP verze 5 nabízí také objektový přístup. Knihovna obsahuje také mnoho funkcí pro práci s HTML stromem, které ovšem v aplikaci nebyly použity. Velmi důležité je správné nastavení knihovny, které se provádí pomocí mnoha přednastavených konstant.

V mé aplikace je TIDY použito pro úpravu HTML dokumentů, tak aby odpovídaly XHTML 1.0 strict. Takovéto dokumenty jsou dále zpracovávány.

8.3 Implementace jednotlivých částí

8.3.1 Robot

Robot je část aplikace, která pracuje bez jakéhokoli zásahu uživatele a využívá další komponenty aplikace. Je to část, která běží na straně serveru a stará se o navštěvování webů a jejich porovnávání s referenčními weby z databáze.

Robot je implementován jako PHP skript, který pro svou funkčnost využívá především metody z třídy Robot. Robot prochází přes všechny definované filtry, které mají zrovna naplánováno sledovaní, stahuje zdrojové kódy sledovaných webů. O stahování těchto informací se stará metoda *getUrlContent ()*. V této metodě je využito jedno ze standardních rozšíření jazyka PHP CURL. Výhodou použití CURL je jeho značná robustnost a malá náchylnost k chybám. Toto má velké využití také pro vzdálenou manipulaci s webovými stránkami a umožňuje například automatický přístup na zabezpečené weby. Hlavní motivací, proč jsem zvolil toto řešení, je fakt, že CURL umožňuje následovat přesměrování a tak se uživatel nemusí starat, zdali bude stránka přesměrována na jinou adresu.

Po stáhnutí zdrojového kódu je tento kód upraven na validní XHTML dokument a poté převeden na stromovou strukturu. O úpravu dokumentu se stará třída Cleaner, která je popsána dále. Stejně tak je dále popsána třída *htmlTree*, která se stará o sestavení stromu HTML dokumentu.

Dalším krokem je připojení k databázi a stáhnutí referenčního zdrojového kódu - cesty ke sledovanému uzlu. Pomocí ní jsou vybrány v novém a starém dokumentu sledované uzly, jež jsou poté porovnány. Nejprve je provedena kontrola změny informace v uzlech. Je provedena na základě MD5 hashe textových informací z daných uzlů. Pokud se tyto hashe nerovnají, dochází k porovnání

dokumentů a formátovaní změněných informací. O porovnávaní a formátování se stará třída *diffSelection*, která je také popsána dále.

Pokud na stránkách došlo ke změnám, jsou tyto změny uloženy do databáze a zároveň je aktualizován zdrojový kód sledované stránky. Po skončení práce s filtrem je ještě do databáze uložen čas dalšího navštívení. Poté je skript uspán na jednu hodinu, po které se celá procedura opakuje.

8.3.2 Příprava html dokumentu

Příprava HTML dokumentu je velmi důležitá. Jen správně formátovaný dokument může být načten do stromové struktury, která je základem pro další práci.

Pro vytvoření validního dokumentu je vytvořena třída *Cleaner*. Nejprve jsou z dokumentu pomocí metody *removeDangerousTags()* odstraněny některé nebezpečné části kódu, které by mohly nadále vadit. Metoda má jako své dva parametry začínající a ukončovací tag, popřípadě část textu a všechno mezi těmito značkami je poté vynecháno. Mezi nebezpečné části, které musí být vynechány, patří komentáře, CDATA a skripty. Komentáře jsou rozpoznány tak, že začínají posloupností znaků `<!--` a končí znaky `-->`. Podobně se rozpoznávají také CDATA, kdy začínající posloupnost je `<!` a konečná je `>`. Skripty jsou vypouštěny především z toho důvodu, že se často nacházejí v těle webu a při sledovaní například celé stránky by mohla vést změna ve zdrojovém kódu skriptu k označení stránky za změněnou. Skripty jsou detekovány pomocí startovacího tagu `<script>` a koncového `</script>`. Po odstranění všech nebezpečných částí je možno dokument dále zpracovat.

Nejdůležitější části přípravy HTML dokumentu je jeho převedení na formu valid. O tento převod se stará PHP rozšíření TIDY, které umožňuje z nevalidních dokumentů udělat validní. V mém případě jsem zvolil validní dokument XHTML 1.0 strict. O převod na validní dokument se stará funkce *tidy_repair_string()*. Jako parametry jsou této funkci předány zdrojový kód HTML stránky a konfigurační pole, které udává, jakým způsobem budou data upravena. V konfiguraci je nastaveno, že se bude upravovat celý web od kořenového uzlu, a že výstup bude formátován v kódování UTF-8. Typ dokumentu je nastaven na formu *strict*. Ze zdrojového kódu je jsou dále odstraněny tagy TBODY, které by vadily při sestavování vizuálního filtru pomocí JavaScriptu. Převedení je realizováno v metodě *repairHtml()*.

Po převedení je dokument ještě dále upravován. Především je nutné jej převést do správného kódování. Zjištění kódování je implementováno ve třídě Robot a stará se o něj metoda *getCharset()*. Kódování stránky je zjištěno buď z meta tagu na stránce, nebo z http hlavičky, kterou posílá server. Problém nastává tehdy, když se tyto dva údaje liší. Jako referenční se v takovém případě bere ta informace, kterou posílá server. Když je kódování zjištěno, je dokument převeden pomocí funkce *iconv()* na UTF-8 a je změněna informace v meta tagu, pokud jej zdrojový kód obsahuje.

Poslední potřebnou úpravou dokumentu je vynechání prázdných znaků mezi jednotlivými tagy. Metoda *removeWhiteChars()* smaže všechny bílé znaky mezi tagy, které by se jinak mohly ukazovat jako textové informace.

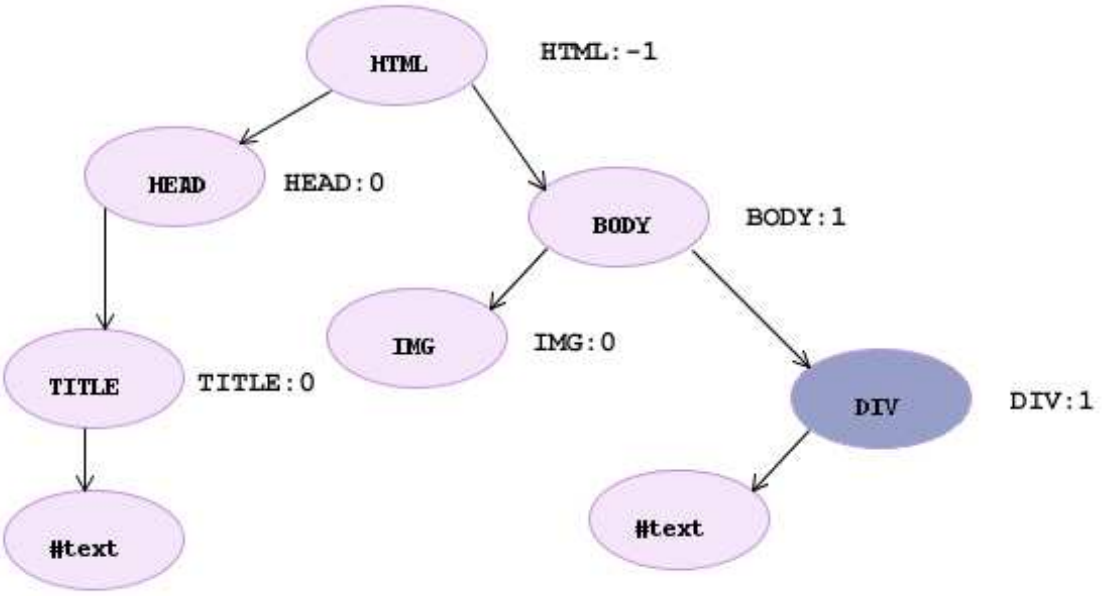
8.3.3 Strom html dokumentu

Vytvoření stromu HTML dokumentu a práce s tímto stromem je základním prvkem celé aplikace. Všechny potřebné funkce pro manipulaci s daty a uzly ve stromu jsou implementovány ve třídě *htmlTree*.

Základem pro stromovou strukturu dokumentu je použití standardní PHP knihovny DOMDocument. Tato knihovna obsahuje třídy, které popisují uzly dokumentu, jednotlivé data, ale obsahuje také vyhodnocovač jazyka XPATH. Pro vytvoření objektu htmlTree je potřeba zadat zdrojový kód, ze kterého má být strom vytvořen. Zdrojový kód je pomocí DOMDocument knihovny a metody *loadHTML()* načten a je tak vytvořen objekt typu DOMDocument.

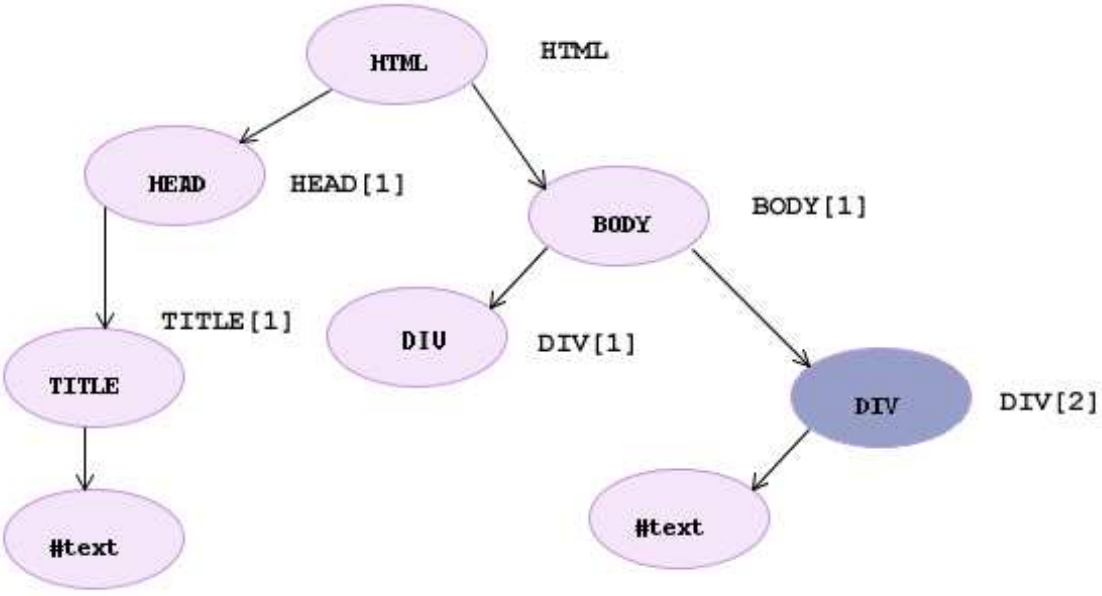
Mezi velmi důležité metody pro práci se stromem je určitě metoda *getAllNodes()*. Tato metoda vrací všechny uzly, které obsahuje daný zdrojový HTML kód. Pro získání všech uzlů, je projita celá struktura, kdy za pomoci zásobníků jsou expandovány všechny uzly od kořenového a postupně ukládány do výsledného pole.

Třída dále obsahuje metody, které vrací cestu, popřípadě XPATH výraz, který přesně popisuje daný uzel. Pro tyto účely jsou implementovány metody *getPathToNode()* a *getXpathToNode()*. Cesta k uzlu je jednoznačná cesta, která daný uzel popisuje ve struktuře dokumentu. Tato cesta je používána pro vytvoření vizuálního filtru. Cesta má následující formát *uzel_rodič:uzel_číslo/uzel_dítě:dítě_číslo*. Kořenový uzel HTML má jako uzel číslo uvedeno -1, což značí to, že nemá žádného předka. V ostatních případech číslo udává, kolikátým potomkem rodičovského uzlu uzel je. Uzly jsou číslovány od 0. Příklad cesty k danému uzlu je znázorněn na Obrázek 15. Zvýrazněný uzel by měl cestu HTML:-1/BODY:1/DIV:1.



Obrázek 15: Cesta k uzlu

XPATH cesta k uzlu je vlastně ekvivalentem předchozího případu, jen s tím rozdílem, že odpovídá jazyku XPATH a může tak být vyhodnocována překladačem, který obsahuje knihovna DOMDocument. Struktura XPATH cesty je velmi podobná s předchozí cestou, stejný případ je pomocí XPATH zapsán tímto způsobem *uzel_rodič[uzel_číslo]/uzel_dítě[dítě_číslo]*. Rozdíl je hlavně v tom, že v XPATH výrazu se číslo uzlu bere podle toho, kolikátý uzel stejného typu to je. Lépe je toto patrné na Obrázek 16. Cesta k zvýrazněnému uzlu by tak byla HTML/BODY[1]/DIV[2].



Obrázek 16: XPATH cesta k uzlu

Třída samozřejmě nabízí metody pro převod mé cesty na XPATH výraz a naopak. Dále metody pro získání všech listových uzlů, nebo jen textových uzlů. Tyto metody jsou realizovány pomocí rekurzivních algoritmů a mnoha dalších pomocných metod.

Další významnou funkcí této třídy je práce na zpětném sestavení HTML dokumentu z daných uzlů. O zpětné sestavení, které se používá při prezentaci výsledků se stará metoda *rebuildPartTree()*. Jako parametr stačí této metodě předat uzel typu DOMNode, zněj je potom sestavena část HTML dokumentu. Uzel se použije jako kořenový a je tak sestaven podstrom, který má všechny vlastnosti jako ten v původním dokumentu. Metoda vrací zdrojový kód části dokumentu. Pro prezentaci změn dále slouží metoda *setChangesToTextNode()*. Která všechny změny uloží do textových uzlů dokumentu na správné místo.

V této třídě je také implementováno patchování HTML dokumentu novými částmi. Toto slouží pro zobrazení změn v celém kontextu stránky. Kdy jsou jednotlivé změny zvýrazněny zeleným rámečkem. O patchování se stará metoda *patchHtml()*. Jako parametry slouží zdrojový kód HTML a pole se všemi uzly, které byly změněny. Tyto uzly jsou pak upraveny právě tak, aby obsahovaly změněné informace a byly obaleny zeleným rámečkem, který usnadňuje uživateli orientaci ve změněné HTML stránce.

8.3.4 Filtr

Realizace filtru probíhá dvěma oddělenými způsoby podle toho, zda se jedná o filtr textový, nebo o filtr vizuální. Textový filtr je realizován pomocí jazyka PHP a vizuální filtr pomocí JavaScriptu.

8.3.4.1 Textový filtr

Textový filtr je realizován pomocí třídy *wordFilter*. Tato třída obsahuje metody pro veškerou práci s textovým filtrem. Pro jeho vytvoření je potřeba zadat typ sledovaných uzlů, klíčové slovo a objekt stromu dokumentu *treeHtml*.

Jako typ se udává číselný údaj 0 - 8, který značí konkrétní HTML tagy, které se budou prohledávat. Aplikace rozlišuje následující tagy. Všechny HTML tagy označuje číslo 0. Jedná se o všechny tagy v těle HTML dokumentu, tedy to co se nachází mezi prvky `<body>`. Další možností je vybrání všech nadpisů. Vybrány jsou všechny elementy `<h1><h2><h3>` a `<h4>`. Odrážky jsou reprezentovány tagy `<ul>` a `<dl>`. Číslované seznamy označuje text mezi značkami `<ol>`. Jako odstavec je text nacházející se v elementech `<p>`. Zvýraznění označuje části, které jsou například psány tučně nebo kurzívou. Jedná se konkrétně o tyto tagy `<b>`, `<i>`, `<strong>`, `<em>`, `<u>`, `<s>`, `<del>`, `<ins>`. Odkazy jsou hledány v elementech `<a>`. Tabulky mezi tagy `<table>` a nakonec je položka text, která se snaží pokrýt zbytek možností, kde všude by se mohli nacházet textové informace. Jedná se tak o tagy `<p>`, `<span>`, `<code>`, `<pre>`, `<div>`.

Pro vybrání jednotlivých tagů podle typu potom slouží metoda *filterTypeInTree()*. Tato metoda na základě XPATH výrazu vybere zvolené uzly z HTML stromu. Například XPATH výraz pro odrážky vypadá následovně `//ul | //dl`. Podobným stylem jsou dále konstruovány i další XPATH výrazy pro výběr typů.

Všechny vyfiltrované uzly se potom předávají ke zpracování metodě *filterNodeByKey()*. Zde jsou projity všechny vybrané uzly a jejich textové informace, ve kterých se pomocí regulárního výrazu hledá zadané klíčové slovo. Je možno nastavit, zda se bude hledat přesně zadané slovo, nebo i jeho část v jiném slově. Tato metoda vrací pole všech uzlů, které vyhovují zadané specifikaci.

8.3.4.2 Vizuální filtr

Na rozdíl od předchozích částí je vizuální filtr tvořen v klientské části aplikace. Vizuální filtr je vytvořen pomocí jazyku JavaScript.

V uživatelské části pro tvorbu filtrů je zobrazen vnořený rám, ve kterém je zobrazena sledovaná stránka. Vzhled stránky nemusí přesně odpovídat vzhledu na webu, protože po odstranění některých nebezpečných částí nemusí například fungovat styly stránky.

Uživatel má tedy možnost označit na stránce myší část textu, která ho zajímá a odeslat tuto volbu. Pro zjištění jakou část webu si uživatel vybral využita JavaScript funkce *getSourceSelection()*. Tato funkce zjistí, kde je vybraná část dokumentu a zároveň zjistí HTML tag, který je označen. Poté se postupuje od tohoto tagu směrem ke kořenovému uzlu dokumentu a sestavuje se cesta, která byla popsána v kapitole 8.2.3. Zvláštní pozor je třeba věnovat tabulkám, kde JavaScriptový DOM sám přidává tag TBODY. Tento tag je tedy odstraněn, aby výsledná cesta byla platná pro daný dokument. Pokud je označen text, který tak obsahuje dva různé uzly, je vrácen ten, který je těmto nadřazený. Pro označení celé tabulky tak stačí označit dva její řádky. Do cesty se neuvádějí textové uzly, které jsou přeskakovány. Po vytvoření cesty je tato informace přenesena do skrytého vstupního pole a odeslána na server pro další zpracování. Na serveru je cesta převedena na XPATH výraz a pomocí tohoto výrazu jsou vybrány všechny uzly, které se mají sledovat.

8.3.5 Zpracování změn

Zpracování změn je prováděno ve třídě *diffSelection*. Objekt této třídy je vytvořen za pomocí čtyř parametrů. Je nutné zadat staré sledované uzly, nové sledované uzly, starý strom dokumentu a nový strom dokumentu.

Pro vyhledávání změněných uzlů se vytvoří dvě asociativní pole se starými a novými uzly. Asociativní pole vypadá následovně *Pole(Nodes)[XPATH]*. Každý uzel je tak možno v poli najít pomocí XPATH cesty k uzlu.

Základní metodou pro vyhledání změn je *getDiffOfTextNodes()*. V této metodě jsou projité všechny vybrané uzly z nového dokumentu a v asociativním poli starých uzlů se podle cesty hledají

odpovídající staré uzly. Vyhledané dva se porovnají a změny se uloží do pole změn. Pokud žádné změny nejsou, potom se do pole uloží prázdný řetězec.

Porovnání uzlů probíhá v metodě *getDiff()*. Důležité je v této metodě použití knihovny *Text_Diff*, což je standardní balíček z PHP PEAR repozitáře. Tato knihovna umožňuje inteligentní porovnání dvou textových informací a nabízí také několik funkcí, jak výsledné porovnání zformátovat. V mé aplikaci je použito formátování, které se využívá například v emailové aplikaci *Horde*. Textové informace, které přibyly, jsou tak uzavřeny mezi HTML značky `<ins>` a informace, které byly odstraněny, jsou uzavřeny mezi značky `<del>`. Toto formátování dává uživateli jasný přehled o tom, jaká část se změnila a jaké nové informace přibyly.

8.3.6 Prezentace změn

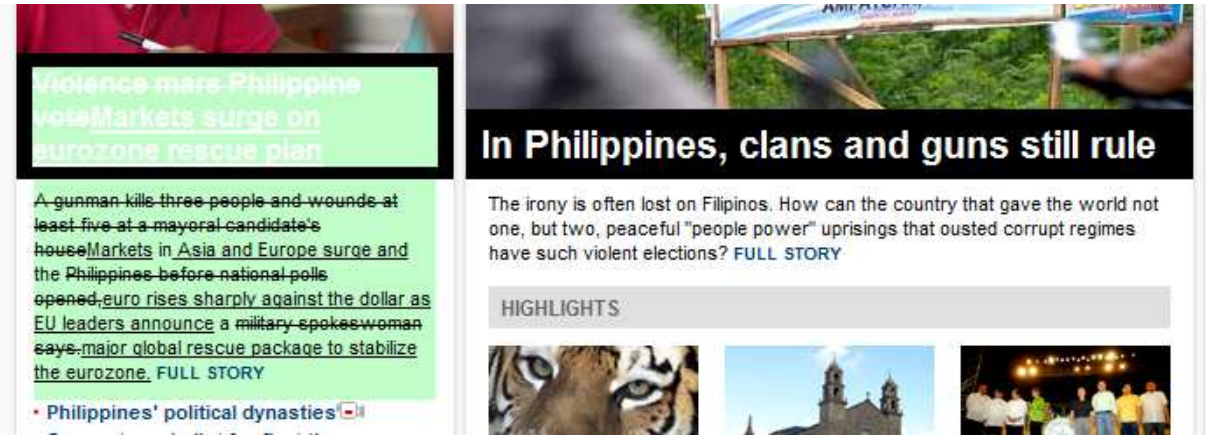
Změny jsou prezentovány buď jako zobrazení celého kontextu nebo jako RSS kanál. Pro vytváření RSS kanálu slouží třída *Rss*. V této třídě jsou z databáze vybrány jednotlivé informace, které jsou zformátovány do XML podoby RSS kanálu. Jednotlivé položky z databáze jsou vypsány v pořadí, kdy nejnovější změna se nachází nejvýše.

Struktura RSS kanálu je uvedena v kapitole 6.2. RSS kanál existuje jak pro celou danou stránku, kdy obsahuje všechny změny ze všech filtrů, které patří dané stránce, ale také RSS kanál pro jednotlivé filtry.

Obrázek 17: zobrazení změn v RSS čtečce

Pro prezentaci změn se dále používá historie změn. Historie změn je uložena v databázi a obsahuje vždy zdrojový kód stránky, a část změněného textu. Každý uzel změněné části je uzavřen mezi tagy *node*, kdy jako atribut je zadána XPATH cesta k danému uzlu. Podle tohoto XPATH výrazu je pak uzel detekován pro zobrazení v celém kontextu stránky.

Zobrazení v celém kontextu stránky je realizováno pomocí metody *patchHtml()* ze třídy *htmlTree*. Jako parametry jsou této metodě zadány uzly, které se budou nahrazovat a změněná část textu z historie. Jak vypadá zobrazení v celém kontextu stránky je vidět Obrázek 18.



Obrázek 18: Dokument s aplikovaným patchem

8.3.7 Autentizace a autorizace uživatelů

Každý uživatel se nejprve musí do systému přihlásit, poté může využívat veškeré možnosti, které aplikace nabízí. Pro autorizaci uživatelů byla implementována třída *Authorization*. Autorizace je prováděna pomocí session proměnných, které nesou unikátní ID pro dané připojení a login uživatele.

ID je uloženo do databáze a pro každou zabezpečenou stránku se kontroluje, zdali je ID platné pro dané připojení. Login je potom jednoznačné jméno uživatele, které slouží pro jeho identifikaci v systému. Hesla jsou v databázi šifrovány pomocí MD5 a tak je zajištěna jejich bezpečnost.

Do session proměnné se také ukládá čas navštívení stránky, podle kterého je potom detekována délka nečinnosti. Po hodině nečinnosti je uživatel automaticky odhlášen ze systému.

V případě vstupu na zabezpečenou stránku bez přihlášení je uživatel vyzván, aby se přihlásil a stránka není zobrazena. Odhlášení potom probíhá tím způsobem, že se smažou všechny session proměnné, které se vážou k přihlašování.

Obrázek 19: přihlašovací obrazovka

8.3.8 Práce s databází

Veškerou práci s databází obstarává třída *Mysql*. V této třídě jsou implementovány funkce, které usnadňují práci s MySQL databází. Pomocí této třídy je možno vkládat prvky do databáze, aktualizovat části databáze, mazat prvky z databáze a provádění všech SQL dotazů.

Metody jsou ošetřeny proti útoku typu SQL injection. Případné chyby jsou vypisovány na standardní výstup a obsahují vždy identifikaci chyby podle *mysql_error()*.

K připojení k databázi slouží metoda *connect()*. Jako parametry jsou zadány adresa serveru, uživatelské jméno a heslo.

9 Výsledky testů

Vytvořená aplikace byla testována na mnoha webových stránkách. Velká část testů byla provedena na stránkách zpravodajských serverů, českých i zahraničních. Většina testů proběhla podle předpokladů a uživatelé byli o změnách správně informováni.

Jako největší problém se při testování ukázala změna struktury stránek, která je pro zpravodajské weby poměrně typická. V některých případech byla tato změna struktury detekována a sledovaný uzel byl vyhledán pomocí alternativní cesty. Někdy ale změna měla za následek sledování špatných informací. Tento problém se však prakticky nedá ovlivnit, protože na velkou změnu struktury webu se nedá univerzálně reagovat. Pokud je změna struktury dlouhodobá, má uživatel jedinou možnost, a to vytvořit filtr znovu a sledovat stránku s novým filtrem. Ve většině případů však byla změna jen dočasná, například pokud byla vložena nějaká aktuální zpráva nebo obrázek. Po vrácení do původního stavu potom sledování pracovalo opět bez problémů.

Další problém, který byl při testování odhalen se týká kódování stránek s diakritikou. Týká se to tedy webů, které používají některá diakritická znaménka. Pro správnou práci aplikace jsou všechny stránky převedeny do kódování UTF-8. Informace o kódování jsou stahovány podle META tagu a informace ze serveru, která má v případě kolize přednost. Při testování však bylo zjištěno, že některé stránky obsahují znaky, které nejsou pro dané kódování přípustné. Převodní PHP funkce *iconv()* si pak s takovými stránkami není schopna poradit. V tomto případě jsou špatné znaky ignorovány, což ale vede k dalším problémům. Na tyto problémové stránky je proto uživatel upozorněn a jejich sledování se nedoporučuje.

Výsledky testů ukázali také, že aplikace má poměrně velké požadavky na velikost databáze. Kvůli archivaci stránek pro historii sledování velikost databáze roste. Tento problém by se dal vyřešit definováním maximálního počtu archivovaných stránek.

Aplikace je optimalizována pro sledování částí webů a není tedy určena pro sledování celých stránek. Při testování bylo zjištěno, že funkce sledování celé stránky je časově poměrně náročná.

Testy aplikace probíhaly především na těchto serverech s různou frekvencí sledování: idnes.cz, novinky.cz, seznam.cz, cnn.com, bbc.co.uk, root.cz, fit.vutbr.cz, centrum.cz, pocasi.centrum.cz/pocasi-v-cr.phtml, blesk.cz, csfd.cz.

10 Zhodnocení práce

Cílem této diplomové práce bylo vytvořit aplikaci pro sledování změn na webových stránkách.

V první části práce jsem se zabýval především teoretickým úvodem do problematiky. Ve druhé potom samotnou realizací výsledné aplikace.

Bylo zjištěno, že na internetu existuje hned několik aplikací, které se zabývají touto problematikou, ale všechny aplikace jsou si velmi podobné. Jejich výhodou je, že uživateli nabízejí možnosti sledování, ale bohužel mnohokrát tyto možnosti nejsou příliš pokročilé. Většina dnes existujících aplikací nabízí sledování celých stránek, což je pro mnoho webů velmi nevhodné. Uživatele totiž zajímají většinou jen jisté konkrétní informace z dané stránky. Dále existují aplikace desktopové, které jsou zaměřeny trochu jiným způsobem, ale v dnešní době je potřeba být informován kdekoliv na světě a tak se tyto aplikace dostávají do pozadí zájmu. Bylo ale dobré prozkoumat i tyto aplikace, protože některé jejich přístupy byly převedeny také do výsledného projektu.

Mnou vytvořená aplikace nemá být konkurencí pro již existující, ale naopak by měla přinášet nový styl sledování a detekce změn.

V dalších kapitolách práce se zabývám možnostmi sledování webových stránek. Tyto možnosti jsou především dvě - sledování celých bloků textu a sledování klíčových slov. V mé aplikaci je použito sledování bloků textu, protože toto sledování je jistě univerzálnější a lépe použitelné pro většinu případů.

Čtvrtá kapitola se věnuje samotnému testování změn obsahu. Je zde popsáno, jaké operace se musejí provést při testování. Nejprve je potřeba dokument zformátovat do použitelné podoby pro parsery, poté vytvořit strom dokumentu. Stromová struktura dokumentu má potom velké výhody pro orientaci a vyhledávání sledovaných uzlů. Tento přístup se dnes také používá ve většině jak komerčních tak vědeckých aplikací.

Z hlediska toho, že tato práce pojednává o vývoji aplikace, která má za úkol zaměřit se na sledování konkrétních částí webu, je velmi důležitá pátá kapitola. V této části jsou rozebrány filtry, které jednoznačně popisují část dokumentu pro sledování. V aplikaci jsou použity dva základní filtry. Filtr textový, který umožňuje specifikovat uzel podle umístění klíčového slova, a filtr vizuální. Vizuální filtr se v jiných online aplikacích nevyskytuje. Ten je právě inspirován filtry, které se používají u aplikací desktopových. Pro uživatele je jistě velmi pohodlný a nabízí poměrně velkou spolehlivost v tom směru, že budeme sledovat to, co opravdu chceme.

Samozřejmě je podstatné získané informace správně prezentovat. Prezentace změn by měla být rychlá a pro uživatele vždy přístupná. Ve výsledné aplikaci bylo zvoleno informování pomocí RSS, které je rychlé, moderní a neobtěžuje uživatele tak, jako informování pomocí emailu. Prezentace výsledků je potom zajištěna přímo v klientské části aplikace.

V druhé části diplomové práce je potom probrána samotná realizace výsledného informačního systému. Systém byl navrhnut jako klient-server aplikace, kdy jako hlavní programovací jazyk byl zvolen jazyk PHP. Další použité nástroje při vytváření jsou uvedeny výše. Pro vývoj byl zvolen objektově orientovaný přístup, který mi umožnil vytvářet jednotlivé části celkem na sobě nezávislé a na konci byly pospojovány do jednoho pracujícího celku.

10.1 Další vývoj

Výsledná aplikace je již dnes funkční a může být použita pro sledování prakticky jakéhokoli webu. Určitě je ale pořád hodně možností jak aplikaci rozšířit.

Hlavní potřeba rozšíření spočívá v robotovi, který navštěvuje jednotlivé weby a provádí sledování změn. Pro vytváření tohoto robota byl použit také jazyk PHP, což se negativně projevuje v jeho rychlosti. PHP jako interpretovaný jazyk je o mnoho pomalejší než jazyky kompilované, do kterých by bylo vhodné robota přepsat. Pro účely této práce je rychlost dostačující, ale s přibývajícími weby časová náročnost poměrně roste.

Dalším možným rozšířením aplikace je možnost informování pomocí emailu. Pro toto rozšíření je v aplikaci všechno připraveno. Každý uživatel musí totiž vyplnit emailovou adresu a tak je možné zasílat na tuto adresu informace o změnách.

Rozšířit by se aplikace mohla také o sledování na základě klíčových slov. Pro některé případy může být toto sledování vhodné a někteří uživatelé by ho mohli ocenit. V nynější aplikaci toto není implementováno, protože při sledování na základě klíčových slov je potřeba procházet vždy celou stránku, což je velmi náročná procedura.

Určitou pozornost by si při dalším vývoji zasloužil také vizuální filtr. Bylo by dobré aplikaci upravit tak, aby uživateli nabídla náhled stránky, který je přesně shodný se stránkou, která je na webu. Toho je ovšem poměrně složité docílit, protože málokterá stránka na internetu je stránkou validní, a tak dobře formátována, že by nemusela být upravena. Tento fakt má ovšem zlepšující se tendenci a tak je možné do budoucna uvažovat i o tomto rozšíření aplikace.

11 Závěr

.

Cílem diplomové práce bylo vytvořit víceuživatelský systém pro sledování a detekci změn na webových stránkách. Vytvořený systém umožní sledování na základě definovaných filtrů a výsledné změny potom uživateli prezentuje ve formě RSS kanálu.

Všechny tyto cíle byly v práci splněny a byla implementována také některá rozšíření, která uživateli zpříjemňují orientaci ve změnách. Může procházet historií změn, zobrazovat si je v kontextu celého webu a tak si vytvářet soukromý archiv jeho oblíbených stránek. Podařila se tak vytvořit aplikace, jejíž obdoba je na českém internetu poměrně vzácná a například definování filtru pomocí označení textu je v online aplikacích tohoto typu novinkou.

Testy ukázaly, že aplikace je funkční, uživatelsky přátelská a připravena k použití. Samozřejmě je možné ji nadále zlepšovat a rozšiřovat. Především v případě, kdy by měla být nasazena pro velké množství uživatelů, pak by bylo pravděpodobně třeba zvýšit efektivitu některých částí.

Vytvořený systém je volně přístupný na internetu a každý uživatel může aplikaci používat prakticky bez omezení. Díky zvoleným implementačním prostředkům je aplikace přenositelná a může být provozována na operačních systém, kde jsou spustitelné všechny její nástroje.

Literatura

[1] KRČMÁŘ, Petr. *Root.cz* [online]. 2006, 14. 9. 2006 [cit. 2010-12-04]. Vše podstatné o RSS. Dostupné z WWW: <http://www.root.cz/clanky/vse-podstatne-o-rss/>. ISSN 1212-8309.

[2] *RSS 2.0 specification.* Technical report, RSS Advisory Board, 2007. Dostupný z WWW: <http://www.rssboard.org/rss-specification>

[3] RSS 2.0 Specification [online]. 2009 [cit. 2009-12-04]. Dostupný z WWW: <http://cyber.law.harvard.edu/rss/rss.html#sampleFiles>

[4] IMAD KHOURY, RAMI M. EL-MAWAS, OUSSAMA EL-RAWAS, ELIAS F. MOUNAYAR, HASSAN ARTAIL, *An Efficient Web Page Change Detection System Based on an Optimized Hungarian Algorithm*, IEEE transactions on knowledge and data engineering, 19, 5, stránky 599-613, Květen, 2007.

[5] ANOOP SANKA. *A Dataflow Approach to Efficient Change Detection of HTML/XML Documents in WebVIGIL*. Diplomová práce, The University of Texas at Arlington, 2003. Dostupný z WWW: <http://itlab.uta.edu/itlabweb/students/sharma/theses/Anoop.pdf>

[6] KAREN BLAKEMAN. *Monitoring Web Page Changes*. RBA Information Services, 2009. Dostupný z WWW: <http://www.rba.co.uk/sources/monitor.htm>

[7] changedetect [online]. 2009 [cit. 2009-12-01]. Dostupný z WWW: <http://changedetect.com>

[8] changedetection [online]. 2009 [cit. 2009-12-01]. Dostupný z WWW: <https://www.changedetection.com/>

[9] FollowThatPage FAQ [online]. 2009 [cit. 2009-12-01]. Dostupný z WWW: <http://www.followthatpage.com/faq>

[10] WatchThatPage [online]. 2009 [cit. 2009-12-01]. Dostupný z WWW: <http://www.watchthatpage.com/index.jsp>

[11] RSS examples [online]. 2009 [cit. 2009-12-01]. Dostupný z WWW: <http://www.rss-tools.com/rss-example.htm>

[12] Woko.cz TECHNICKÉ INFORMACE [online]. 2009 [cit. 2009-12-01]. Dostupný z WWW: <http://www.woko.cz/akce.phtml?ukaz=technicke>

[13] ARTAIL, Hassan; ABI-AAD, Michel . An enhanced Web page change detection approach based on limiting similarity computations to elementsof same type. *Journal of Intelligent Information Systems* [online]. 12. 10. 2007, 32, 1, [cit. 2010-05-05]. Dostupný z WWW: <http://www.springerlink.com/content/7x30516728584g75/fulltext.pdf>. ISSN 1573-7675.

[14] XML Validation [online]. 2009 [cit. 2009-1-5].
Dostupný z WWW: <http://www.w3schools.com/Xml/xml_dtd.asp>

[15] Well formed XML documents [online]. 2009 [cit. 2009-1-5].
Dostupný z WWW:
<http://www.javacommerce.com/displaypage.jsp?name=wellform.sql&id=18238>

[16] HAVEL, Jakub . *Zive.cz* [online]. 2002, 13. 12. 2002 [cit. 2010-05-11].
XML pro web aneb od teorie k praxi, 2.díl. Dostupné z WWW:
<http://www.zive.cz/clanky/xml-pro-web-aneb-od-teorie-k-praxi-2dil/sc-3-a-109709>.

[17] BŘÍZA, Petr . *Interval.cz* [online]. 2004, 9. 4. 2004 [cit. 2010-05-11].
Základy jazyka XPath. Dostupné z WWW:
<http://interval.cz/clanky/zaklady-jazyka-xpath/>.

[18] KOSEK, Jiří. *Zdroják.cz* [online]. 2009, 26. 10. 2009 [cit. 2010-05-11]. DOM – načteme to do paměti. Dostupné z WWW:
<http://zdrojak.root.cz/clanky/dom-nacteme-to-do-pameti/>.

[19] TIDY - Introduction [online]. 2010 [cit. 2010-5-10]
Dostupný z WWW:
<http://www.php.net/manual/en/intro.tidy.php>.

[20] HERNANDEZ, George. *XPath* [online]. 2007, 11.7.2007 [cit. 2010-05-17]. XPath.
Dostupné z WWW:
<http://www.georgehernandez.com/h/xComputers/XML/XSL/XPath.asp>.

[21] ACHOUR, Mehdi , et al. *PHP Manual : Client URL Library* [online]. 1997, 2010-05-15 [cit. 2010-05-17].
Dostupné z WWW: <http://php.net/manual/en/book.curl.php>.

[22] BLAHETA, David, Internetový portál výzkumné skupiny, bakalářská práce, Brno, FIT VUT v Brně, 2007

Seznam použitých zkratek

HTML - HyperText Markup Language

XHTML - eXtensible HyperText Markup Language

RSS - Rich Site Summary

XPATH - XML Path Language

DOM - Document Object Model

PHP - Hypertext Preprocessor

XML - Extensible Markup Language

URL - Uniform Resource Locator

SQL - Structured Query Language

UTF - UCS Transformation Format

SMTP - Simple Mail Transfer Protocol

IMAP - Internet Message Access Protocol

POP3 - Post Office Protocol version 3

Seznam příloh

Příloha 1

Uživatelský manuál

1 Seznámení s aplikací

Aplikace slouží jako prostředek pro sledování změn na webových stránkách. Hlavním účelem je přinést uživateli co nejlepší informace ze sledované stránky a zobrazit aktuální změny.

Aplikace je především určena pro sledování částí stránek, není tedy příliš vhodná pro sledování celé stránky, i když i toto umožňuje.

Aplikace je standardní webová služba, která je spouštěna v prohlížeči, a tak může být použita na všech systémech, kde je možno spustit jeden ze standardních prohlížečů s grafickým režimem. Testována byla na prohlížeči Firefox verze 3 a Internet Explorer verze 8.

2 Hlavní část

Hlavní část aplikace je společná pro uživatelskou i administrátorskou část. Jedná se o obrazovku s jednoduchým menu, kde je možné provést registraci, popřípadě se přihlásit jako běžný uživatel, nebo jako administrátor.

Obrázek 1: Hlavní menu

1. Odkaz vede zpět na hlavní stránku aplikace.
2. Přes tento odkaz se dostanete do sekce přihlašování do aplikace.
3. Registrace nového uživatele je prováděna přes tuto položku menu.
4. Kontakt na správce aplikace

2.1 Registrace a přihlášení

Registrace uživatelů je prováděna přes klasické formulářové pole. Po registraci je umožněno přihlásit se do systému a začít se sledováním stránek.

Obrázek 2: Registrace

1. Položky registrace - všechny musí být vyplněny
    - **Login:** uživatelské jméno, které musí být jedinečné pro každého uživatele. Je možno zadat jméno o délce až 20 znaků.
    - **Heslo:** Uživatelské heslo o maximální délce 50 znaků. Heslo je do databáze ukládáno v zašifrované podobě.
    - **Jméno uživatele:** Normální jméno uživatele až 20 znaků
    - **Příjmení:** příjmení o délce až 40 znaků
    - **Email:** platný email. Po rozšíření aplikace o tento email mohou být odesílány změny stránky.
2. Tlačítko potvrzující registraci, po které je možno se do systému přihlásit.

Obrázek 3: Přihlášení do systému

1. Login a heslo, které bylo zadáno při registraci do systému.
2. Tlačítko, po kterém následuje přihlášení do systému.

3 Uživatelská část

V uživatelské části je možno přidávat sledování a pracovat se sledovanými stránkami. Po přihlášení se zobrazí okno, které nabízí přístup k hlavním částem aplikace.

Obrázek 4: Hlavní menu

1. Vrací na úvodní stránku.
2. Umožňuje přidávat web pro sledování.
3. Umožňuje prohlížet již sledované stránky.
4. Umožňuje upravovat profil uživatele.
5. Nápověda.
6. Kontakt na správce aplikace v případě problémů.
7. Odhlášení z aplikace.

3.1 Přidat sledování

Přidání sledování je základním úkonem pro práci s aplikací. Do systému je možné přidat jakoukoli webovou stránku. Do políčka URL stačí jen zadat adresu stránky, zvolit filtr a tím je možné začít stránku sledovat. Důležité je zadávání adresy. Je potřeba dát pozor, jestli má stránka adresu s www nebo bez něj. Špatné zadání adresy může způsobit chyby. Chybějící *http://* je automaticky doplněno.

Obrázek 5: Přidat sledování

1. Adresu sledované stránky je nutné zadat existující adresu ve správném formátu. Pokud byla stránka zadána špatně, je na to uživatel upozorněn.

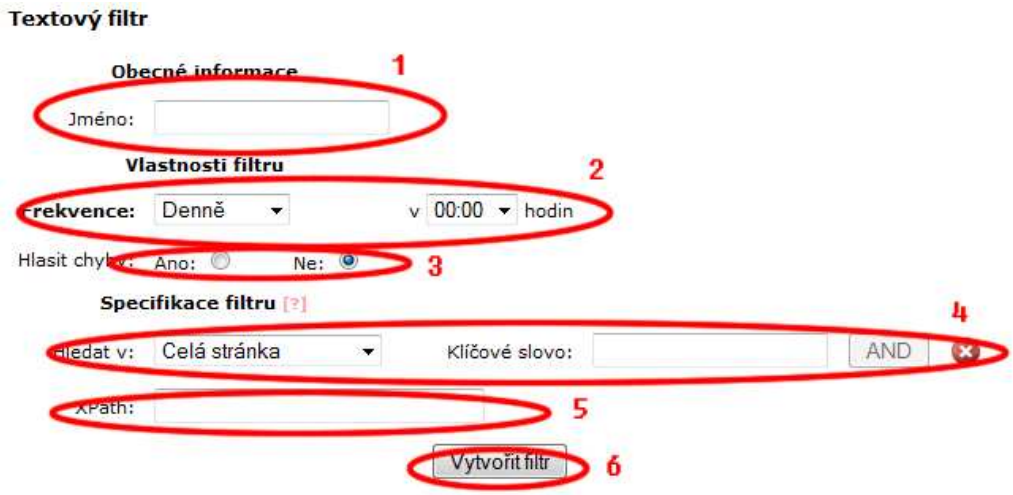
2. Volba filtru umožňuje zvolit, buď textový filtr nebo vizuální. Vizuální filtr je pro uživatele přehlednější a rychlejší, ale nenabízí tolik funkcí jako textový, kde uživatel má vše ve své moci.
3. Po potvrzení je stránka přidána do sledování a uživatel přechází na vytváření filtru.
4. Kontextová nápověda.

3.2 Filtry stránky

Jelikož aplikace slouží především pro sledování částí stránky, jsou důležité uživatelské filtry, které nám sledovanou část definují. Na výběr je textový a vizuální filtr.

3.2.1 Textový filtr

Textový filtr umožňuje zadat informace o sledování v textové podobě. Uživatel má díky němu kontrolu filtru zcela ve svých rukou a může tak specifikovat i více částí webu nezávisle na sobě.



Obrázek 6: Textový filtr

1. Jméno filtru, které není povinné
2. Frekvence sledování je důležitá položka, určuje, kdy začne sledování a jak často se bude stránka navštěvovat. Začít sledování může v jakoukoli denní hodinu, pro frekvenci je na výběr denní, hodinová a týdenní.
3. Hlášení chyb umožňuje potom v RSS hlásit chyby uživateli.
4. Specifikace, kde na stránce má být nalezena sledovaná část, možné jsou následující volby:
    - **Celá stránka:** hledá se v elementu body na celé stránce webu
    - **Nadpisy:** hledá se v elementech `<h1>, <h2>, <h3>, <h4>`
    - **Odrážky:** všechno mezi značkami `<ul>, <dl>`

- **Číslované seznamy:** všechno mezi značkami `<ol>`
- **Odstavce:** všechno mezi značkami `<p>`
- **Zvýraznění:** všechno mezi značkami `<b>, <i>, <strong>, <em>, <u>, <s>, <del>, <ins>`
- **Odkazy:** všechno mezi značkami `<a>`
- **Tabulky:** všechno mezi značkami `<table>`
- **Text:** všechno mezi značkami `<p><span><code><pre><div>`

Klíčové slovo potom lépe specifikuje vybranou část webu. Klíčové slovo musí být zadáno přesně. Pokud není klíčové slovo zadáno, může být sledována jen vybraná část webu, tedy všechny, které vyhovují (například všechny nadpisy). Pomocí tlačítka AND potom mohou být přidávány další sledované části stránky. Ikona křížku slouží k odstranění sledovaných částí.

5. XPATH výraz, který specifikuje danou část. Pokud je XPATH zadán, je zbytek volby umístění ignorován.
6. Vytvoří filtr a přejde na seznam sledovaných stránek

3.2.2 Vizuální filtr

Vizuální filtr nabízí grafickou tvorbu filtru. Uživatel vidí celou stránku a pomocí myši nebo klávesnice může označit sledovanou část. Jinak je tvorba stejná jako u textového filtru.

Obrázek 7: Náhled stránky

1. Náhled celé stránky, ve které je možno označovat. Po kliknutí na otazník se zobrazí kontextová nápověda.
2. Část označená uživatelem, která bude sledována.
3. Filtr se vytvoří po kliknutí na " Vytvořit filtr z výběru ".

3.3 Sledované stránky

V této části má uživatel k dispozici veškerou správu nad sledovanými stránkami.

Obrázek 8: Sledované stránky - položka

1. Adresa stránky, kdy po rozkliknutí je možno vidět detail stránky.
2. Smazání stránky. Se stránkou jsou smazány všechny filtry, RSS i historie. Stránka už tak není sledována.
3. RSS kanál stránky obsahující všechny filtry stránky.

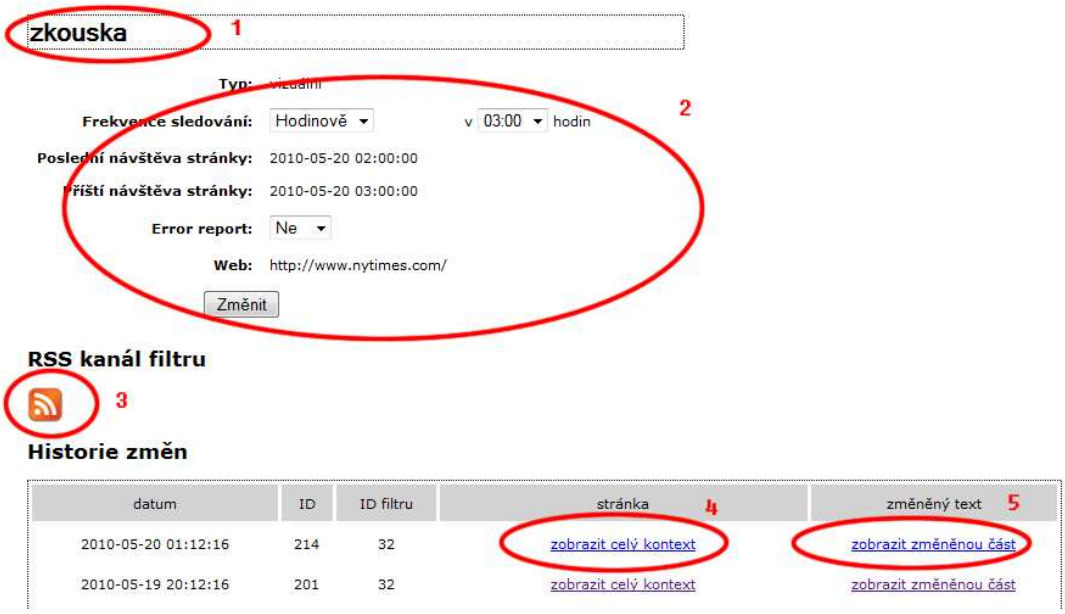
3.3.1 Detail stránky

Umožňuje přejít na další vlastnosti stránky a dává přehled o celé stránce.

Obrázek 9: Detail stránky

1. Adresa stránky.
2. Všechny filtry stránky. Po kliknutí na název je možno přejít na detail filtru. Pokud filtr nemá jméno, je zobrazeno jeho ID. Ikona křížku slouží ke smazání filtru. RSS a historie filtru jsou tak smazány a tento filtr se nepoužívá ke sledování.
3. Možnost přidat další filtr k dané stránce.

3.3.2 Detail filtru

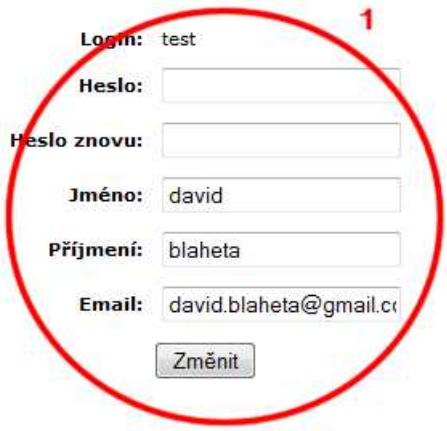


Obrázek 10: Detail filtru

1. Jméno filtru, které může být editováno.
2. Další údaje, které je možno editovat. Po stisknutí tlačítka jsou údaje změněny.
3. RSS kanál jen daného filtru.
4. Umožňuje zobrazit sledovanou část v celém kontextu stránky. Zobrazí se celá stránka, kde je sledovaná část označena zeleně.
5. Umožňuje zobrazit jen změněnou část, která je formátována stejně jako na stránce. Jsou vyznačeny změny a nové informace.

3.4 Můj profil

V této části si uživatel může změnit své osobní údaje. Jediný údaj, který nemůže být změněn je uživatelské jméno.



Obrázek 11: Profil

1. Údaje které je možno změnit jsou ve formulářových polích. Po stisknutí tlačítka jsou údaje změněny a uživatel do aplikace přistupuje již s novými údaji.

3.5 Ostatní funkce

Další funkce, které nabízí uživatelský režim už nejsou ničím složité. Je možno zobrazit nápovědu, která je podobná tomuto manuálu a umožňuje uživateli jednoduše pochopit všechny funkce aplikace. Pod položkou kontakt se skrývá údaj pro kontaktování správce systému v případě jakýchkoli chyb v aplikaci, nebo problémů s aplikací.

Pro odhlášení z aplikace uživatel jednoduše klikne na položku menu odhlásit.

4 Administrátorský režim

Administrátor se do aplikace přihlašuje stejně jako normální uživatel, ale na rozdíl od něj má možnost spravovat jednotlivé uživatele. Administrátorské rozhranní aplikace je velmi jednoduché a ovládá se stejně jako uživatelská část. Systém má vždy jednoho administrátora, jehož heslo a jméno je přiděleno přímo v databázi aplikace.

4.1 Správa uživatelů

Po přechodu do této části se zobrazí seznam všech uživatelů, kteří jsou zaregistrováni do systému. Po kliknutí na jejich uživatelské jméno je možno přejít na detail uživatele.

4.1.1 Detail uživatele

V detailu uživatele jsou vidět všechny údaje uživatele a je zde možnost ho smazat. Po smazaní jsou ze systému odstraněny také všechny sledované stránky uživatele, všechny filtry, všechny RSS informace a veškerá historie jeho stránek.