VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

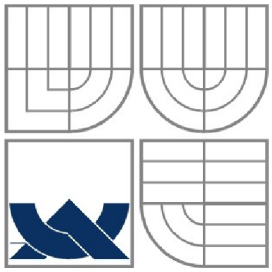
WEBOVÝ PROHLÍŽEČ AUDIO/VIDEO ZÁZNAMŮ PŘEDNÁŠEK: PŘIDÁNÍ FUNKCIONALITY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

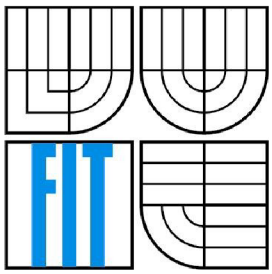
AUTOR PRÁCE
AUTHOR

ANTONÍN KALVODA

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

WEBOVÝ PROHLÍŽEČ AUDIO/VIDEO ZÁZNAMŮ PŘEDNÁŠEK: PŘIDÁNÍ FUNKCIONALITY

WEB BASED AUDIO/VIDEO LECTURE BROWSER: ADDING OF FEATURE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

ANTONÍN KALVODA

VEDOUCÍ PRÁCE
SUPERVISOR

ING. IGOR SZÖKE

BRNO 2010

Abstrakt

Práce se zabývá přidáním funkcionality k již vytvořenému webovému prohlížeči záznamů přednášek, která byla vypracována Ing. Josefem Žižkou jako diplomová práce. První část je úprava pomocného programu pro extrahování slidů tak, aby byla možná co nejvyšší míra automatizace při přidávání nových videozáznamů do systému. Druhá část se zabývá vytvářením pluginů do JW Playeru, který přehrává videozáznamy přednášek.

Abstract

Thesis deal with adding of features to already created Web-based lecture browser, which was elaborated by Ing. Josef Žižka as a dissertation. The first part is adjustment of utility for slides detection and extraction with the claim on the highest possible automation to add new videos into the system. The second part deals with creating plugins for JW player which plays video lectures.

Klíčová slova

JW Player pluginy, Flash, ActionScript, Převod PDF, BashScript, zrychlené přehrávání, JW Player komentáře

Keywords

JW Player plugins, Flash, ActionScript, PDF conversion, BashScript, Fast forward playing, JW Player comments

Citace

Kalvoda Antonín: Webový prohlížeč audio/video záznamů přednášek: přidání funkcionality, bakalářská práce, Brno, FIT VUT v Brně, 2010

Webový prohlížeč audio/video záznamů přednášek: přidání funkcionality

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Igora Szöke. Uvedl jsem všechny literární prameny a publikace, z nichž jsem čerpal.

.....
Antonín Kalvoda
19. 5. 2010

Poděkování

Především bych chtěl poděkovat Ing. Igoru Szöke, který vedl a konzultoval tuto bakalářskou práci. Taktéž se podělil o mnoho nápadů na vylepšení a v případě výskytu jakéhokoliv problému se aktivně podílel na vyhledání způsobu jeho vyřešení. Poděkování patří i autorovi webového prohlížeče přednášek Ing. Josefu Žižkovi, jenž poskytl tento systém a v případě nepochopení zdrojových kódů podával vysvětlení. Děkuji také tvůrci zde využitého systému pro správu komentářů Radimu Glajcovi a Bc. Jakubu Janovičovi za poskytnutí návrhu a realizaci databáze pro webový přehrávač záznamů přednášek. Nutno je také poděkovat škole za zapůjčení licence na vývojové prostředí Adobe Flash. Na závěr musím vyjádřit dík i Ing. Vítězslavu Beranovi za poskytnutí aplikace na extrahování slajdů z videa.

Obsah

1 Úvod

V předchozích letech byla vyvíjena internetová aplikace, která měla za úkol studentům Vysokého učení technického fakulty informačních technologií zjednodušit přístup k jinak pracně vyhledávaným informacím a tím *zefektivnit jejich sebevzdělávání*. Webový prohlížeč videozáznamů přednášek je systém obsahující záznamy kurzů vyučovaných na VUT FIT. Je speciální tím, že zobrazuje transkripty neboli přepisy řeči na text. Aplikace je nadále vyvíjena několika bakalářskými i diplomovými pracemi, jejichž náplní je například přetváření systému uložení dat do databáze, se kterým je třeba počítat, jelikož na tuto verzi se bude průběžně přecházet. Cílem bakalářské práce je přidání prospěšné funkcionality tomuto webovému prohlížeči přednášek.

Druhá kapitola obsahuje seznámení se všemi využitými technikami i technologiemi, kterých bylo třeba k nastudování zdrojových kódů webového prohlížeče záznamů přednášek či vytvoření praktické části této bakalářské práce.

Pro správce přidávání přednášek je náročné zadávání záznamů do systému a tudíž je kladen důraz na vysokou úroveň automatizace spojenou s úsporou času a s eliminací možnosti chyb při zadávání. Pomocný program pro extrahování slajdů tento požadavek nesplňuje. Řešení tohoto problému je náplní třetí kapitoly.

Náplní kapitoly čtvrté je přidání možnosti rychlejšího přehrávání. To by uživatelům systému umožnilo lépe nakládat se studijním časem, především u kurzů, které jsou přednášeny příliš pomalým mluvčím.

Kapitola pátá se zaobírá přidáním komentářů a časových poznámek k právě přehrávané přednášce. Tyto komentáře a poznámky by mohly pomoci studentům vybrat si z přednášky to nejdůležitější pro jejich potřeby studia.

Závěr v šesté kapitole obsahuje zhodnocení dosažených výsledků, napojení na jiné práce a možnosti budoucího vývoje rozšíření, jež byly bakalářskou prací vytvářeny.

2 Použité technologie

V této kapitole jsou popsány technologie využívané pro vývoj jednotlivých částí práce.

2.1 HTML (HyperText Markup Language)

HTML bylo vyvinuto Timem Berners-Lee a Robertem Caillau *na základě jazyka SGML*, který byl pro jejich práci příliš složitý. Při vytváření informačního systému CERN v r. 1989 promysleli nápad na vytvoření standardu jednoduššího. V roce 1990 navrhli tento systém HTML a zároveň pro něj vytvořili prohlížeč. První oficiální verze byla vypuštěna v r. 1991.

Tento *jazyk je charakterizován* především konečnou *množinou značek neboli tagů* a jejich atributů. Značky určují sémantiku obsaženého prvku a jsou uzavírány do úhlových závorek, z nichž koncová značka obsahuje lomítko. Příklad takovéto konstrukce je:

* Text *

Ovšem u některých elementů lze vynechat koncovou značku.

Celý programový kód je uzavřen mezi značky <html> a </html>, jenž dávají jasně najevo, že se jedná právě o zápis jazyka HTML. Dále je dokument strukturován na hlavičku „head“ a tělo „body“. V hlavičce jsou umístěny metadata, tedy informace o dokumentu. Naopak v těle je především vlastní obsah dokumentu.

Rozvržení dokumentu je definováno buď tzv.” *framesety*” považovanými za zastaralé. Dále se používá tabulkový layout, kde se rozmístění v dokumentu vytváří pomocí *tabulky*. Takto navržený dokument však není z programátorského pohledu příliš správný, jelikož každý element se má používat na co byl předurčen. Nejpoužívanější metoda, jak navrhnout html stránku, je pomocí CSS. *CSS* je *stylový předpis* a může být buď u každého elementu přímo v html kódu, nebo se tyto styly vloží do jednoho souboru, jak je doporučeno.

Grafická stránka tohoto dokumentu může být definována některými atributy elementů, ale v dnešní době se preferují stejně jak u rozvržení dokumentu CSS předpisy[1].

2.2 CSS (Cascading Style Sheets)

CSS je jazyk pro popis formátování a designového návrhu stránky, ať už se jedná o HTML, XHTML či XML. Byl navržen konsorciem W3C kolem roku 1997. Nápad na jeho zhotovení vznikl díky myšlence definovat vzhled mimo obsah stránky a tím dodat programátoru větší přehled o svých zdrojových kódech. CSS lze zapisovat buď přímo do HTML kódu (tzv. inline styl) nebo do zvláštního souboru, který je následně do stránky přilinkován. Styly se definují dle následujícího zápisu:

Selektor {vlastnost: nastaveni;vlastnstost2:nastaveni;... }

Selektor určuje, pro jaké elementy se má tento blok provést, přičemž může obsahovat třídy, identifikátory, ale také tagy. Dále jsou nastavovány vlastnosti odpovídající vybranému elementu.

Přestože má CSS spoustu omezení, je nejpoužívanější metodou pro definici layoutu[2].

2.3 PHP (Hypertext Preprocessor)

Za vznikem tohoto skriptovacího jazyka stál v roce 1994 autor Rasmus Lerdorf. Ten jej nejdříve zapsal pomocí jazyka Perl. Jelož však nebyl příliš výkonný, byl přepsán do jazyka C.

PHP generuje dynamicky html kód na straně serveru, jenž je následně zaslán klientovi. PHP není závislé na platformě a syntaxe je inspirována mnoha jazyky. Velmi často se využívá ve spojení s XML nebo některou databází, která slouží jako datový sklad. Velká výhoda tkví v tom, že je podporován prakticky všemi poskytovateli webového prostoru. Vyvíjení tohoto jazyka pokračuje i nadále, což potvrzují počty rok od roku narůstajících frameworků[3].

2.4 Uložení dat a informací

2.4.1 XML (eXtensible Markup Language)

XML vznikl taktéž jako HTML z jazyka SGML, ale jeho využití se trochu liší. Tento standard se využívá v oblastech *výměny dat mezi aplikacemi nebo prezentaci dat*, v němž neřeší vzhled. Protože se jedná o formát otevřený a neváže se na žádnou platformu, začal být hojně využíván. Grafická reprezentace se dodává do těchto dokumentů pomocí CSS stylů nebo tzv. XSL (eXtensible Stylesheet Language), ta umožní dokument různě transformovat a upravovat.

Tento jazyk neobsahuje žádné předdefinované značky, takže definice zůstává na programátorovi. Značky lze definovat pomocí DTD, ale také XML schémat. Vytvořený XML dokument lze poté ověřit na validitu pomocí *parseru*. XML dokument musí splňovat základní pravidla jenž jsou[4]:

- Obsahuje jeden kořenový (root) element.
- Neprázdné elementy musí být ohraničeny startovací a ukončovací značkou. Prázdné elementy mohou být označeny tagem „prázdný element“.
- Všechny hodnoty atributů musí být uzavřeny v uvozovkách – jednoduchých (') nebo dvojitých ("), ale jednoduchá uvozovka musí být uzavřena jednoduchou a dvojitá dvojitou. Opačný pár uvozovek může být použit uvnitř hodnot.
- Elementy mohou být vnořeny, ale nemohou se překrývat; to znamená, že každý (ne kořenový) element musí být celý obsažen v jiném elementu.

2.4.2 SQL (Structured Query Language)

První myšlenka je datována na 70. léta 20. století firmou *IBM*, která provozovala výzkum relačních databází a kladla si za úkol vytvořit sadu příkazů pro práci s ní.

SQL je dotazovací jazyk *pro práci s relačními databázemi*. Je navržen tak, aby se co nejvíce blížil stavbě vět v přirozeném jazyce. Pro práci s ním je potřeba jen několik málo příkazů logicky se řadících za sebe.

Data jsou uložena v tabulkách, nad nimiž se provádí různé dotazy a různé transakce měnící obsah tabulek.

Mnoho firem si vytvořilo své jazyky, které se od konkurenčních lehce liší zálohováním, transakcemi a jiné. Mezi ně patří např.: PostgreSQL,MS SQL, MySQL, Oracle.

MySQL

Je jedna z nejpoužívanějších SQL databázích vůbec. Je to nejspíše z důvodu velké podpory poskytovatelů webového prostoru a jazyku PHP. Tato multiplatformní databáze byla vyvinuta švédskou firmou MySQL AB. Patří mezi nejrychlejší databáze vůbec, ovšem za cenu jednodušších zálohovacích metod. Proto není příliš nasazována na komerční aplikace, ale pro naše účely plně vystačuje[5].

2.5 Javascript

Pochází z dílny Brendana Eicha, někdejšího vývojáře firmy Netscape. Tento objektově orientovaný jazyk pro WWW stránky, jenž je zapsán nebo přilinkován do html kódu a následně spouštěn na webovém klientovi, nevytěžuje stranu serveru. Je standardizovaným jazykem, který umožní rozpohybovat a dynamicky upravovat obsah stránek na straně klienta. Tento jazyk je nezávislý na platformě, ale v některých webových prohlížečích se jeho implementace liší.

Jelikož je JS jazyk pracující na straně klienta, vznikají pro něj některá omezení. Nesmí pracovat se soubory na disku, ale může využívat cookies, což může sloužit pro ukládání nastavení stránky. JS skriptování lze úplně zakázat, ovšem některé stránky nezaručují správnost zobrazení.

Jazyk byl původně vyvíjen pod názvem LiveScript, následně však byl přejmenován na JavaScript, aby se svezl na slávě produktu Java od Sun Microsystems. Později byl JS standartizován u firmy ECMA a dostal název ECMAScript[6].

2.6 AJAX (Asynchronous JavaScript and XML)

Ajax je označení technologie asynchronního načítání stránek. To znamená, že není potřeba obnovovat celou stránku znovu, a tak jednoznačně omezit objem přenášených dat. Tato metoda umožňuje vyšší

interaktivitu webových stránek. Dokáže spolupracovat s HTML a XHTML stránkami a pro své fungování potřebuje klientský javascript. V praxi jde o využití objektu „XMLHttpRequest“, který si vyměňuje se serverem informace v různé podobě (HTML, XML, JSON, plain-text,..). Pochází z dílny Microsoftu, který jej představil v roce 2000 na svém programu pro správu pošty.

Mezi nevýhodu této techniky patří rozhodně prodleva mezi jednotlivými AJAXovými komunikacemi.

2.7 OpenCV

Multiplatformní knihovna pro C/C++, ale i dalších jazyků, jenž obsahuje velké množství algoritmů a funkcí pro vývoj programů v oblasti počítačového vidění a zpracování obrazu. Za touto knihovnou stojí firma Intel, jež ji používala jako proprietární a později ji vydala pod BSD licencí.

První alpha verze zazářila na IEEE konferenci počítačového vidění a rozpoznávání objektů v roce 2000. Následovalo několik beta verzí a v roce 2006 vyšla oficiálně první stabilní verze 1.0. Spolu s verzí 1.1 vydalo v r 2008 nakladatelství O'Reilly Media publikaci „Learning OpenCV: Computer Vision with the OpenCV Library“, která OpenCV přiblížila širší veřejnosti. V roce 2009 byla vypuštěna verze 2.0, která obsahovala nejen nové funkce, ale optimalizovala i funkce staré pro vícejádrové procesory.

2.8 Aplikace det_slajds

Jedná se o program sloužící k rozpoznávání slajdů ve videu. Porovnává slajdy obsažené ve videu na promítacím plátně se slajdy ve formě obrázků, jejichž cesty jsou zadány v xml souboru, který je součástí vstupu této aplikace. Při správném nastavení může *det_slajds* také extrahovat slajdy, což může být v případě kreslení přednášejícího na plátno velmi užitečná vlastnost. Ke svému fungování potřebuje tato utilita knihovny ffmpeg, OpenCV a libxml. Aplikace prozatím funguje pouze ve spolupráci s OS Windows. Příklad spuštění:

```
det_slajds.exe -i ISS_2008-10-03.avi -p slides/detected/ -s slides.xml

kde   -i je vstupní videosoubor se záznamem přednášky,
      -p umístění pro extrahování slajdů,
      -s xml soubor s umístěním, kde se referenční slajdy nachází
```

Takto spuštěný program zanalyzuje umístění plátna a vytvoří si konfigurační xml soubor. Následně je video procházeno po několika snímcích a porovnáváno se slajdy ve formátu PNG, zadanými ve slides.xml.

2.9 Adobe Flash

Adobe Flash (původně Macromedia Flash) je platforma používaná pro vývoj internetových aplikací ať už se jedná o animace, přehráváni multimedií či hry. Aplikace vytvořené tímto nástrojem se často nazývají Rich Internet Application (RIA), volně přeloženo bohaté internetové aplikace.

Programy oživené Flashem můžou obsahovat jak rastrové tak vektorové prvky. Tyto prvky mohou spolupracovat s dalšími předdefinovanými objekty či vlastními komponentami. Flash zastřešuje ohromnou sílu v podobě objektově orientovaného jazyku ActionScript, v němž lze vytvářet jak vizuální tak nevizuální komponenty včetně všech animací.

K rozšíření Flashe pomohla malá velikost výsledných souborů, protože se uchovávají ve vektorovém formátu[7].

2.10 ActionScript

ActionScript (dále jen AS) je skriptovací jazyk založený na ECMAScriptu. Využívá se především pro vývoj webových stránek a komponent, které jsou oživovány a rozhýbávány Flashplayerem. Původně byl vyvinut a vlastněn firmou Macromedia, jež byla později odkoupena společností Adobe. AS2 byl zpočátku vyvinut na ovládání 2D prostoru animací vytvořených v Macromedia Flash/Adobe Flash. Teprve pozdější verze umožnily vytváření webových her, internetových aplikací a možnost streamovat (přehrávat za běhu) video i zvuk.

AS je jazyk, který nabízí schopnost několika málo příkazy oživit stránky i objekty. Novější verze AS mají stejnou syntaxi jako JS, ale odlišný způsob zápis knihoven a tříd. AS se používá především pro aplikace s vysokou úrovní interakce objektů.

Příchod Flash MX 2004 znamenalo vypuštění skriptovacího jazyka AS2 pro vývoj flashových aplikací.Tento však nebyl příliš používán, protože lze díky výbornému vývojovému prostředí téměř všechny akce provést bez psaní skriptů.

Spolu s flashplayerem v9 (v roce 2006) byla vydána verze AS3, která z něj dělá objektově orientovaný jazyk. To tvoří silnou stránku Flashe hlavně při vývoji rozsáhlejších aplikací. Důležitým milníkem byla spolupráce s XML a take s technologií AJAX. Znamenalo to příchod technologie nazvané flex, která dostala i vývojové prostředí s názvem Adobe Flex (nyní Flash Builder) s možností vytvářet webové stránky s obrovskou interaktivitou a spolu s tím možnost zabezpečit Flash volání proti nabourání[8].

2.11 JW Player

JW Player je flashová aplikace často používaná na přehrávání streamovaných audio/video souborů přes internet. Tento přehrávač je vyvíjen firmou LongTail video.

Přehrávač je navržen tak, aby jeho modifikovatelnost byla jednoduchá. Skýtá možnost vytváření a přidávání dalších pluginů jako je zobrazení titulků, vytváření statistik či přidání audio popisek k videu. Přehrávač obsahuje řadu parserů pro různé typy XML souborů a spoustu další užitečných funkcí, které lze využívat při tvorbě pluginů

Implementace je založena na programovém kódu jazyka ActionScript, jenž je určen pro aplikace Adobe Flash a Adobe Flex. Firma LongTail zpřístupnila pro nekomereční účely všechny zdrojové kódy, které je možné libovolně modifikovat a používat pod volně šiřitelnou licencí. Multimediální přehrávač JW player je optimalizován pro vytváření a přidávání vlastních pluginů. Pro všechny tyto pluginy je předpřipravena šablona, kterou lze při vývoji použít:

```
package {

import flash.display.Sprite;
import com.jeroenwijering.events.*;
public class Helloworld extends Sprite implements
PluginInterface {

    /** Configuration list of the plugin. **/
    public var config:Object;
    /** Reference to the JW Player View object. **/
    private var view:AbstractView;
    /** This function is automatically called by the player
after the plugin has loaded. **/
    public function initializePlugin(vw:AbstractView):void {
        view = vw;
        view.sendEvent(ViewEvent.TRACE,"Hello World!");
    };
}
}
```

Tyto pluginy lze obohacovat o jakékoliv vizuální či nevizuální komponenty Adobe Flash či Flex. Lze také využívat všechna volání JW Playeru, která pro něj byla implementována, informování o změně stavu, času či posouvání ve videu.

2.12 Komunikace webové stránky a JW Playeru

Komunikace mezi aplikací webový přehrávač záznamů přednášek a flashovou aplikací JW Player probíhá přes jazyk JavaScript, který nejprve tento přehrávač zinicializuje a následně nastaví „posluchače“, kterými získá informace o přehrávaném videu jako je právě přehrávaný čas, stav přehrávače, apod. Dále probíhá komunikace zasíláním zpráv, díky nimž je možné JavaScriptem zastavit přehrávání, posunout přehrávání na určitý čas atd.

2.13 Mencoder

Tato volně šiřitelná konzolová aplikace sloužící jako konvertor video souborů obsahuje zároveň spoustu filtrů pro úpravu videa i audia. Ve Webovém prohlížeči záznamů přednášek hraje roli konvertoru záznamu přednášky ve formátu AVI na FLV.

2.14 GhostScript

GhostScript nebo také GS je interpret jazyka PostScript, který skýtá velké možnosti a od verze 3.33 je také souborového formátu PDF. Umí převádět výstupy na nejrůznější typy zařízení, ať už se jedná o obrazovku, tiskárnu či externí soubor.

Tento program byl vyvinut roku 1988 L. Peterem Deutschem, který jej naprogramoval v jazyce C a zpřístupnil jeho zdrojové kódy. GS se stal základem mnoha aplikací jako např. tiskové drivery, převodní filtry nebo front-end uživatelská rozhraní. Samotný program je ovládán výhradně příkazovou řádkou, ale pro jeho rozšíření mezi veřejnost vzniklo také několik grafických rozhraní[9]. Příklad spuštění:

```
gs -sDEVICE=png16m -r80x80 -o slides-%03d.png -q -dFirstPage=5
-dLastPage=7
```

3 Automatizace detekce a extrahování slajdů

Zadávání nových přednášek do systému je netriviální záležitost z důvodu nutnosti provádění části procesů v prostředí Linux a části v systému Windows. Jelikož je webový prohlížeč záznamů přednášek lokalizován na Linuxovém serveru, je pro automatizaci naprosto nezbytné přesunout veškeré akce na tento systém a tím eliminovat zásah v současnosti vyžadovaného lidského faktoru.

Aplikace nazývaná *det_slajds* sloužící *pro extrahování a rozpoznávání slajdů,* zobrazovaných při přehrávání záznamu přednášky ve vyšším rozlišení než v přehrávaném videu, není připravena na automatizaci. Program byl vytvořen v OS Windows ve vývojovém prostředí Visual Studio naprogramovaný v jazyce C/C++, je tedy nezbytné provést takové úpravy, které umožní spouštět program v prostředí Linux bez grafického rozhraní.

Ing. Josefem Žižkou bylo poznamenáno, že program pracuje s PNG soubory, které byly dosud manuálně extrahovány z formátu PDF, softwarem s grafickým rozhraním v prostředí Windows. Pro vyšší úroveň automatizace je nezbytné řešení takovým způsobem, jenž by správce záznamů nenutil k další interakci se systémem.

3.1 Cíl práce

Cílem práce je přeložení aplikace *det_slajds* v prostředí Linux a rozšíření vstupního formátu slajdů o PDF soubory.

3.2 Realizace programu *det_slajds* pro Linux

Pro kompilaci byl vybrán OS Ubuntu. Ten disponuje jednoduchou správou balíčků, což pro mě jakožto převážně ve Windows operujícího studenta znamenalo velké plus.

Především bylo nutné nainstalovat všechny potřebné knihovny, jejichž průběh je sepsán v příloze 1. Zpočátku byla zvažována možná integrace do vývojového prostředí NetBeans, avšak toto řešení bylo shledáno zbytečným, protože vývoj bude i nadále pokračovat pod systémem Windows. Nutností se stalo vytvořit Makefile, kterým budou zdrojové kódy kompilovány. Jelikož mezi zdrojovými kódy jsou soubory jazyka C tak i C++, bylo nutné použít dva překladače. Zdrojové soubory jazyka C byly přeloženy pomocí GCC a pro soubory C++ byl využit g++ překladač. Samozřejmostí bylo přilinkování knihoven OpenCV, Libxml a ffmpeg bez jejichž přítomnosti developerských verzí překlad nenastane. Po úspěšném překladu bylo třeba oříznout výstup programu tak, aby neprodukoval žádné grafické prvky. Ty by jednak byly na serverové stanici zbytečné,

ale hlavně způsobily by pád této aplikace. Dále byl program upraven tak, aby s určitým přepínačem neprodukoval jakékoliv výstupy a tím zbytečně nezahlcoval server, na němž *det_slajds* běží.

3.3 Extrahování PDF souborů

Bylo zvažováno více variant, jakými rozšířit formát slajdů zadávaných na vstupu.

- Přidání knihovny pro úpravu pdf souborů do stávajícího programu na detekci slajdů, jež by měla mimo jiné vlastnost konverze do jiných formátů.
- Vytvoření externího programu, který by používal knihovnu pro úpravu pdf souboů.
- Vytvoření BASHScriptu využívajícího některou utilitu/software Linuxu

Byla vybrána varianta poslední, jež se ukázala být jako nejlepší pro dané účely.

Přidání knihovny do aplikace *det_slajds*

Jedná se o způsob, kdy je program rozšířen o knihovnu pro úpravu pdf souborů řízenou dalším modulem programu *det_slajds*. Je nutno podotknout, že k této variantě by bylo třeba připojit i další knihovny schopné pracovat se souborovým systémem. To by znamenalo najít adekvátní alternativu i pro OS Windows a zdrojové kódy by tímto navyšovaly svoji velikost. I přesto není toto řešení špatné, ale utilita *det_slajds* není vyvíjena výhradně pro účely webového prohlížeče záznamů přednášek a tudíž přidávání nových rozšíření není v tomto případě vhodné.

Vytvoření externího programu pro extrakci PDF souborů

Znamenalo by vytvoření externího programu v jazyce C++ pracujícího s knihovnou pro úpravu a konverzi souborů typu PDF, jež by obsahovala vlastnost konverze souborů do PNG. Řešit tímto způsobem problém je možné, ale jeví se jako přístup neúměrný, když existuje v Linuxu spoustu utilit schopných činnost zvládnout stejně dobře nebo dokonce lépe.

Vytvoření BASHScriptu, který se stará o extrakci PDF souboů i obsluhu aplikace *det_slajds*

Vytvoření BASHScriptu znamená využívat velké síly BASHe, systémové příkazy a utility prostředí Linux. Tento přístup se jeví jako nejlepší, přičemž úprava skriptu je jednodušší, než změna zdrojového kódu a následná kompilace. Proto byl realizován skript s názvem *auto_det_slajds* zastřešující přípravné práce aplikace *det_slajds*.

Nejdůležitější část celého skriptu je výběr utility pro konverzi PDF do obrázkových PNG. Ten probíhal mezi:

- balíkem ImageMagick a jeho podprogramem convert
- a GhostScriptem.

Utilita convert by mohla sloužit jako řešení daného problému. Je ovšem nutné zdůraznit, že standardně nebývá v systému nainstalována a je nutné explicitně balík ImageMagick přidat. Dále nelze s aplikací specifikovat stránky, které budou vyextrahovány. Jak bylo později zjištěno, balík ImageMagick vychází z GhostScriptu majícího více a detajlnějších možností převodu. Díky tomu bylo také použito řešení s GhostScriptem nativně obsaženém ve většině distribucí Linuxu.

BASHScript neřeší jen převod formátu PDF do PNG souborů, ale bylo mu zakomponováno i načítání slajdů ve formátu PNG. Ty v případě, že neexistuje dokument PDF, stačí překopírovat do podsložky slides/ pracovního adresáře a budou zahrnuty do skriptem vygenerováného xml souboru s umístěním slajdů.

Skript musí splňovat nejvyšší míru automatizace, aby správce záznamů nemusel zadávat údaje, jež by se daly zjistit nebo vydedukovat. Proto byl skript navržen tak, aby mohl být spuštěn bez jakýchkoliv parametrů, což navodí plně automatizovaný režim. Plně automatizovaný režim vyhledá v pracovní složce videosoubor se záznamem přednášky, extrahuje pdf soubry se slajdy do PNG a následně vygeneruje XML soubor, který bude obsahovat cesty k těmto extrahovaným slajdům požadovaným aplikací *det_slajds*. Toto vše je realizováno systémovými utilitami OS Linux. Jediné, co je třeba před prvním spuštění skriptu zkontrolovat, je správné nastavení cesty k aplikaci *det_slajds* udávané proměnnou *$pathtodet*.

4 Zrychlené přehrávání záznamů přednášek

Toto rozšíření bylo navrženo z důvodu podávání informací přednášejícím s několikanásobně nižší rychlostí, než by studentům stačilo pro zdárné vstřebávání probíráné problematiky. Posluchači tak zbytečně ztrácí koncentraci i čas, který by mohli věnovat dalšímu studiu. Neposledním motivem bylo to, že každý student si jistě rád před zkouškou zopakuje látku, která se probírala v průběhu semestru, ovšem průchod všech záznamů přednášek daného kurzu není možný už z důvodu, že přednášky jsou dlouhé tři hodiny a jejich počet může dosáhnout až čísla třináct.

Řešení tohoto problému pro implementaci může být více, přičemž se vždy musí jednat o vytváření programu v jazyku ActionScript, kterým lze pĺuginy k JW Playeru implementovat.

JW Player je komponenta systému, která přehrává videozáznamy přednášek. Tento přehrávač se nachází ve stavu, kdy umí přehrávat zadanou přednášku a připojuje trasskripty, jenž jsou zaslány derivátem XML souboru známým pod zkratkou TTML.

4.1 Cíl práce

Cílem práce bylo vytvořit plugin pro změnu rychlosti přehrávání záznamů.

4.2 Plugin pro zrychlené přehrávání videa

Nejdříve bylo třeba rozhodnout, jakým způsobem implementace se vydat.

- Plugin, realizující rychlé přehrávání videa.
- Pomocný plugin, který by načítal soubory s upravenou rychlostí videa.

Plugin, realizující rychlé přehrávání videa:

Situace, kdy se na právě přehrávané video použije filtr, jenž je převzorkuje na rychlost požadovanou uživatelem. Tento případ je ideální, avšak filtr, který by toto imlementoval, není doposud ve flashi potažmo ActionScriptu implementován a jeho vývoj by se spíše ujal jako práce pro celou skupinu lidí, než jako součást bakalářské práce. Proto bylo nutné tento způsob zavrhnout.

Pomocný plugin, načítající soubory videa s upravenou rychlostí:

Plugin v případě požadované změny rychlosti provede načtení souboru obsahujícího převzorkované video na zadanou rychlost. Takto realizovaný plugin by pracoval bezproblémově v případě, když by nebylo nutné časově synchronizovat transkripty s videem, jak je tomu u webového

přehrávače záznamů přednášek. Za předpokladu, že videosoubor, jehož přehrávací rychlost je 1.0x bude mít délku záznamu 00:15:00, by při rychlosti 1.5x měl délku 1.5x menší (Tabulka 4.1).

Rychlost	1.0x	1.2x	1.5x
Délka	00:15:00	00:12:30	00:10:00

Tabulka 1: Příklad délky videa pro různé rychlosti

Zobrazení transkriptů nemůže být v takovou chvíli regulérní. Bylo tedy nutné provést taková opatření, která by tuto záležitost ošetřila.

V úvahu by přicházelo přečasování celé transkripce nacházející se na webovém rozhraní. Takové řešení by znamenalo úpravu nejen všech používaných i do budoucna použitých pluginů, ale při každém přehrávání provádět mnoho výpočtů spojených s aktuálně přehrávanou rychlostí. Náročnost přehrávání by tímto úkonem rapidně stoupla a na pomalejších počítačích by mohla způsobovat až chvilkové zamrznutí webového prohlížeče. Bylo tedy nutné provést zásah do zdrojových kódů JW Playeru.

Zdrojové kódy JW Playeru jsou umístěny na oficiálních stránkách tvůrce tohoto multimediálního přehrávače ve více verzích. Pro zachování kompatibility se systémem webový prohlížeč záznamů přednášek, byl využit poslední build JW Playeru verze 4, který není zdaleka poslední verzí tohoto populárního přehrávače.

Zásah do zdrojového kódu programu JW Player:

Jde o modifikaci zdrojového kódu JW Playeru, který zasílá upravené údaje o délce videosouboru a stejně tak i o pozici videa, v níž se přehrávač nachází. Díky tomuto zásahu přibývají vteřiny na přehrávači rychleji a tím pádem je zaručená synchronizace se zobrazovanými transkripty i všemi pluginy závislými na čase.

4.3 Vizuální komponenty pro změnu rychlosti

Nutností se stalo vytvoření způsobu, jakým přepínat rychlost přehrávání. Jako nejpřívětivější pro uživatele systému se jevila možnost pomocí ovládacího panelu JW Playeru. Bylo vytvořeno tlačítko, které rozvine nabídku s instancí komponenty reprezentující táhlo, sloužící pro změnu rychlosti. Pro ujištění správnosti vybrané rychlosti se doimplementovalo grafické upozornění s aktuální rychlostí přehrávání.

Nabídka s táhlem byla v případě skrytí ovládacího panelu stále viditelná, proto se vytvořila další událost přehrávače JW Player, která byla zasílána v případě jeho skrytí. Odchycením této události a následným skrytím táhla byla tato nechtěná vlastnost eliminována.

4.4 Vytváření videosouborů s vyšší rychlosti přehrávání

Videosoubory s rychlostí 1.0x jsou vytvářeny programem mencoder, což je výhodné, neboť tato utilita má zabudován režim převzorkování na vyšší rychlost. K přednášce, kterou je třeba konvertovat s vyšší rychlostí, stačí spustit mencoder se stejnými parametry jak tomu bylo doposud[10], a přidat navíc parametr *-speed* určující rychlost přehrávání.

```
mencoder $input_video_file -ofps 25 -o $output_video_file -of lavf
-oac mp3lame -lameopts abr:br=56 -srate 22050 -ovc lavc -lavcopts
vcodec=flv:keyint=25:vbitrate=$vbitrate:mbd=2:mv0:trell:v4mv:cbp:la
st_pred=3 -speed 1.5
```

Takto vytvořený záznam má chybu v podobě nesprávného vyladění zvuku a hlas přednášejícího nezní tak jak by měl. Tuto vlastnost lze eliminovat nasazením metody brzdění/zrychlení neboli pitch a nastavit její meze. Mencoder bohužel takovou funkcí nedisponuje, avšak existuje přídavný plugin řešící tento nedostatek. Jedná se o balík TAP-Plugins, konkrétně plugin LADSPA, jehož instalace je v příloze 1.

Nastavení správnosti mezí je důležité pro věrohodnost hlasu přednášejícího. Její výpočet je následující[11]:

```
mez A = (100-100x)/x kde x je rychlost přehrávání
mez B = -90
```

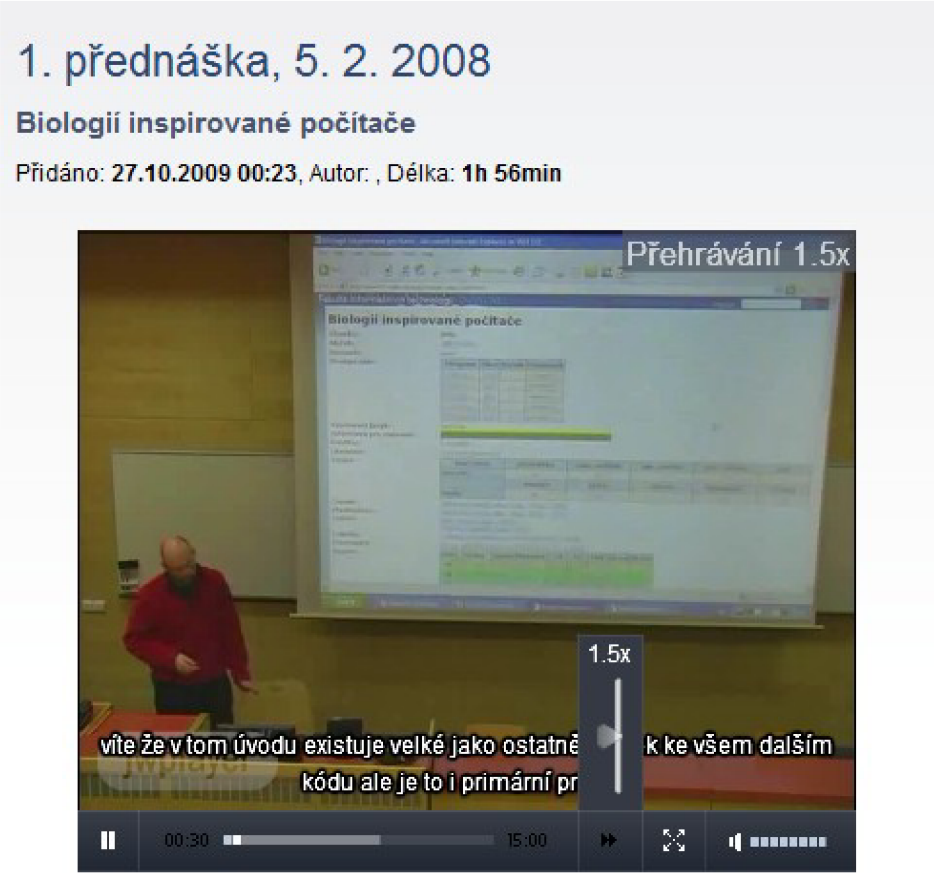
Pro rychlost přehrávání 1.5x tedy stačí připojit k mencoderu tyto parametry:

```
-af ladspa=/usr/lib/ladspa/tap_pitch.so:tap_pitch:0:-33:-90:0 -
speed 1.5
```

Takto vytvořený záznam splňuje jak zrychlené přehrávání, tak správnou úroveň hlasu přednášejícího. Jak takto vytvořený soubor propojit s pluginem fastplay je nastíněno v příloze 2.

4.5 Finální podoba JW playeru s pluginem Fastplay pro zrychlené přehrávání

Na této stránce naleznete podobu přehrávače JW Player s přidaným pluginem pro zrychlené přehrávání záznamů (Obrázek 4.1). Objekty implementované pluginem jsou aktuální rychlost přehrávání, umístěná v pravém horním rohu a tlačítko s vygenerovaným táhlem, nacházející se ve druhé třetině ovládacího panelu. To slouží pro nastavení rychlosti přehrávání.



Obrázek 4.1: Přehrávač s pluginem pro zrychlené přehrávání

5 Komentáře a časové poznámky k videu

Uživatelé serverů poskytujících přehrávání medií jako je například YouTube.com, uveřejňují své názory na shlédnutá videa prostřednictvím komentářů umístěných pod přehrávačem. Tím vyjadřují svůj souhlas či nesouhlas s účinkujícím, jeho výkonem i náplní tohoto videa. Podobnou šanci by měli mít i uživatelé webového prohlížeče záznamů přednášek, tedy studenti.

Pro tento systém by byly vhodné dva druhy komentářů k záznamu. První by byl vázán na přednášku jako celek, tzv. klasický komentář a poté časová poznámka, jež by souvisela s určitým časem přednášky.

Klasickými komentáři mohou být například vyjádření směrem k úrovni celkové obtížnosti přednášené látky nebo schopnost přednášejícího podat probírané téma. Takové komentáře bývají běžnou součástí hodnotícího systému.

Studenti využívající systém by jistě ocenili, aby se na záznamu přednášky objevilo jen to nejdůležitější. Takové úseky přednášek zatím bohužel nelze strojově vymezit, ale student shlédnutý přednášku jistě zvládne posoudit, co je důležité a co nikoliv. Tento student může svým kolegům, kteří v budoucnu záznam shlédnou, oznámit, že následujících 15 minut je naprosto nepotřebných a nezajímavých. Takovéto poznámky by měly být více viditelné. Příklad tohoto případu může být:

Student zašle komentář „Přednášející probírá své zážitky z dovolené“ v úseku 00:00:00 – 00:05:00.

Takový komentář by měl dalšího studenta vybudit k pozornosti a je na jeho uvážení, zda shlédne tento pětiminutový úsek přednášky, který nesouvisí s přehrávaným předmětem nebo jej přeskočí.

5.1 Cíl práce

Cílem této práce je implementovat komentáře a časové poznámky k videu, které bude možno zobrazovat ve stromové struktuře. Umožnit jejich přidávání, mazání či upravování a při dokročení na časovou poznámku zobrazit zvýrazněně její podobu.

5.2 Implementace klasických komentářů

Systém operace s komentáři byl implementován v jejich zjednodušené verzi, tedy bez grafické úpravy či využívání karmy. Vývoj byl zastaven, protože paralelně vznikala bakalářská práce řešící stejnou problematiku. Práce byla rozdělena na části:

- Vytvoření architektury, způsobu uložení, implementace grafického rozhraní pro oba druhy komentářů.
- Způsob zobrazování časových poznámek k záznamu přednášky přímo v přehrávači, který se stal součástí této bakalářské práce.

5.3 Zobrazení časové poznámky k videu

Zobrazování tímto způsobem je výhodné, pokud pracuje i v režimu pro celou obrazovku. Komentář je vykreslen přímo na plochu přehrávače s průhledným podkladem, aby bylo vidět přehrávané video a zároveň poznámka nezanikla. Takto vykreslený komentář nelze ve stávajícím systému zobrazit jiným způsobem, než vytvořením pluginu pro JW Player v jazyce ActionScript3, který bude tuto záležitost řešit.

JW Player má vytvořen spoustu pluginů z nichž právě „Captions“, používaný pro zobrazení transkripce, je podobný tomuto problému. Proto se stal tento plugin inspirací pro vyvíjené rozšíření zobrazující poznámky „Notes“. Byly použity podobné techniky pro získání dat, resp. časových poznámek. Tzn. využití formátu TTML (Timed Text Markup Language)[12], jenž vznikl z XML.

Plugin Notes je implementován tak, že při dosažení startovacího času zobrazí text, který náleží časové poznámce. V praxi jde o to, že pokud existují časové poznámky s překrývajícími časy, tak při dokročení na kolidující úsek je zobrazen komentář se startovací pozicí položenou dále ve videu. V případě stejného startovacího času je zobrazen komentář, který byl přidán později.

Jelikož celý systém uložení dat pracuje nad databází MySQL je třeba v tom i nadále pokračovat. Bylo tedy nutné vytvořit PHP skript zajišťující konverzi dat z databáze do formátu TTML a vytvoří tak vrstvu mezi databází a JW Playerem.

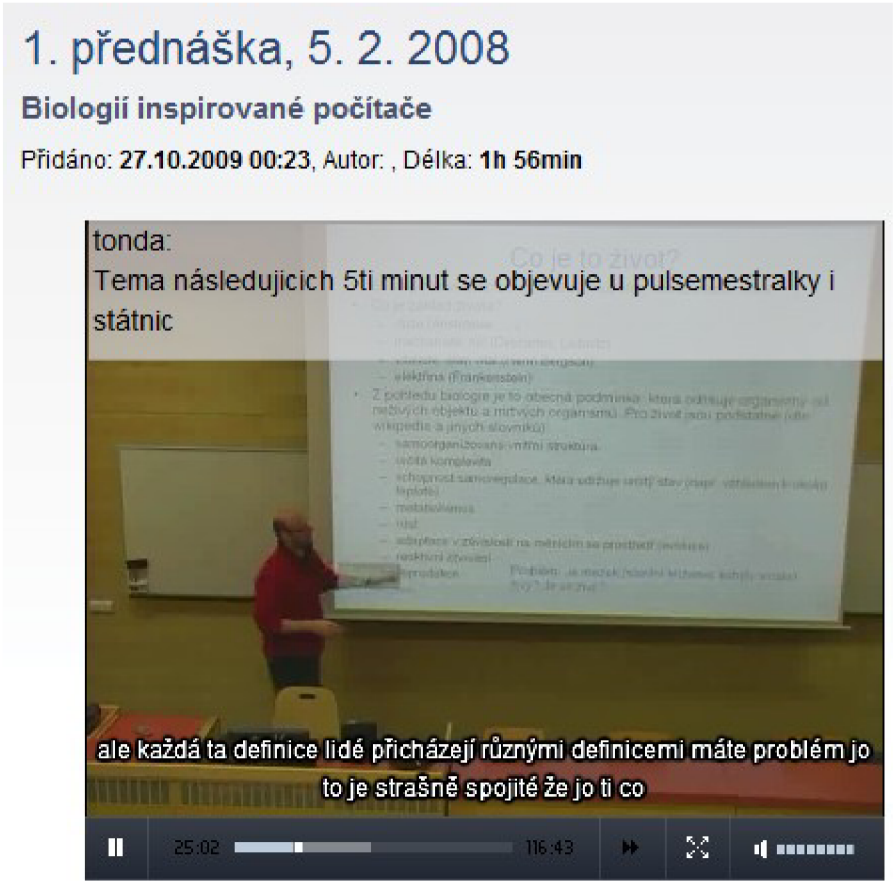
Pro zobrazení komentáře přímo ve videu byla použita komponenta MovieClip, která patří mezi stavební kameny vizuální části jazyka ActionScript . Textová část zobrazení poznámky byla položena na MovieClip, jenž je viditelný v případě dokročení na startovací pozici poznámky.

Už z návrhu pluginu Notes je zřejmé, že přidání časové poznámky neznamená její okamžité zařazení do poznámek zobrazovaných v přehrávači. Bezprostřední načtení do přehrávače by znamenalo opakovat databázový dotaz na komentáře. Pro nižší vytížení školního serveru byla tato možnost zavržena.

Pro tento plugin nebylo vytvořené grafické webové rozhraní, neboť paralelně vznikala bakalářská práce řešící přidávání klasických komentářů včetně dalších funkcionalit, což by při nasazení mnou vytvořeného GUI znamenalo jeho brzký zánik. Proto bylo použito řešení systému zobrazování komentářů vyvinuté kolegou Radimem Glajcem.

5.4 Finální podoba JW Playeru s pluginem Notes pro zobrazování časových poznámek

Na této stránce se nachází ukázka JW Playeru s pluginem zobrazující časové poznámky (Obrázek 5.1). Poznámka je zobrazena ve vrchní části přehrávače přes celou šířku.



Obrázek 5.1: Přehrávač s pluginem pro zobrazování časových poznámek

6 Závěr

Předmětem bakalářské práce bylo rozšířit funkcionalitu webového prohlížeče záznamů přednášek vytvořeného Ing. Josefem Žižkou v roce 2009. Pro vývoj dalších rozšíření byly autorem poskytnuty veškeré zdrojové kódy, jenž tento systém pohání.

Prvním úkolem bylo seznámit se s prohlížečem audio/video záznamů přednášek a systémem využívanými technologiemi. Jednalo se především o PHP třídy, JavaScriptové funkce, AJAXové spolupráce se systémem a v neposlední řadě způsob uložení dat v XML souborech. Z velké části šlo o prohloubení znalostí z absolvovaných předmětů Informační systémy a Tvorba webových stránek. Největším přínosem z nastudování daného prohlížeče záznamů bylo využití techniky OOP v PHP, s níž jsem se zatím příliš nesetkal.

Následovalo stádium, kdy bylo třeba stanovit mnou implementované rozšíření. V této fázi mi velmi pomohl vedoucí Ing. Szöke, jenž nabízel různé druhy funkcionalit možných k implementaci.

Původně byl záměr této práce kompletně implementovat program na detekci a extrahování, ovšem po přečtení diplomové práce autora webového prohlížeče záznamů přednášek byla zjištěna existence takového softwaru. Proběhla konzultace z níž vzešlo, že program není zkompilovaný v prostředí Linux, a z toho důvodu jej nelze začlenit do automatizovaného procesu. Proto jsem na požádání autora Ing. Vítězslava Berana obdržel zdrojové kódy s informacemi, které knihovny jsou pro vývoj využívány. Tento software na detekci a extrahování slajdů napsaný v jazyce C/C++ byl vyvíjen pod MS Windows ve Visual Studiu. Naprosto zásadní věcí se stalo, aby zpracovával PDF soubory a hlavně mohl být provozován v prostředí Linux, což byla pro mě neznámá. Další rozšíření funkcionality byla implementace pluginů pro flashový přehrávač JW Player zobrazujícího záznamy přednášek. Prvním z nich se stalo zrychlené přehrávání záznamu a druhým zobrazování komentáře vázané na určitý časový úsek.

Nezbytnou pro vývoj byla komunikace s paralelně vznikajícími pracemi, jež participovaly na vývoji webového prohlížeče záznamů přednášek. Prvním milníkem ve vytváření bakalářské práce bylo přecházení k systému, který ukládal svá data v databázi. Návrh i realizaci databáze provedl Bc. Jakub Janovič a jeho práce byla v průběhu letního semestru připojena k systému, na kterém vznikaly zmiňovaná rozšíření. Dalším důležitým krokem se stalo rozdělení práce vývoje systému přidávání komentářů mezi mě a kolegu Radima Glajce.

Tato bakalářská práce mi rozšířila znalosti v oblasti prostředí OS Linux a kooperaci se systémovými utilitami. Další nevyhnutelnou a nejrozsáhlejší záležitostí této práce se stalo nastudování jazyka ActionScript, prostředí Flash a jeho principů, jež bylo pro vytvoření JW Player pluginů nezbytné.

Práce se dělila na část pomocných nástrojů pro správu záznamů přednášek a na přidání funkcionality, která by byla přínosná pro uživatele tohoto systému. Důraz byl především kladen

na automatizaci v části správy systému a využitelnost na straně uživatelů v případě pluginů. Všechny funkcionality byly úspěšně implementovány tak, jak je zadání vyžadovalo a jejich vývoj může pokračovat i nadále.

Pohled na další vývoj aplikace *det_slajds*, řešené ve třetí kapitole:
Skript pro *det_slajds* by mohl za předpokladu, že se v jednotlivých učebnách nebudou měnit pozice plátna na kameře, obsahovat přepínač určující, o jakou učebnu se jedná a následně by pro ni vybral konfigurační soubor.

Nyní pracuje *det_slajds* tak, že nejprve proběhne detekce pozice plátna, díky které se vytvoří konfigurační soubor s pozicí plátna a poté se použije na detekci slajdů. Konfigurační soubor ovšem není vždy vygenerován optimálně a díky tomu nejsou některé slajdy rozpoznány korektně.

Program det_slajds je však stále ve vývoji a nevylučuje se optimalizace detekce. Proto je toto spíše provizorní řešení.

Budoucí vývoj pluginu pro časové poznámky z páté kapitoly:
V pluginu Notes je velký potenciál, jeho využití může být např.:
Studenti by mohli přidávat různé druhy poznámky.

- Typ „warning“ značící nesmírně důležitou látku, který by se zobrazoval na rudém podkladu.
- Typ „offtopic“ znamenající mluvení přednášejícího o nijak nesouvisícím tématu, zobrazovaný žlutě a spolu s ním by byla nabízena možnost přeskočení tohoto úseku.
- A další možné typy, které by mohli nabízet různé interakce se studenty.

Plugin pro rychlé přehrávání obsažený v kapitole čtvrté je ve své finální podobě a neumím si představit jeho další možné rozšíření.

Literatura

[1] *HyperText Markup Language - Wikipedie,otevřená encyklopedie* [online]. [2010] , 04.05.2010 [cit.2010-05-05]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/HyperText_Markup_Language >.

[2] *Úvod do CSS – Jak psát web* [online]. [2010] , 02.05.2010 [cit.2010-05-05]. Dostupný z WWW: <http://www.jakpsatweb.cz/css/css-uvod.html>.

[3] *Php - Wikipedie,otevřená encyklopedie* [online]. [2010] , 05.05.2010 [cit.2010-05-05]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Php >.

[4] *Extensible HyperText Markup Language - Wikipedie,otevřená encyklopedie* [online]. [2010] , 01.04.2010 [cit.2010-05-05]. Dostupný z WWW: <http://cs.wikipedia.org/wiki/Extensible_HyperText_Markup_Language>.

[5] *Mysql - Wikipedie,otevřená encyklopedie* [online]. [2010] , 06.05.2010 [cit.2010-05-06]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Mysql>.

[6] *A re-introduction to JavaScript - MDC* [online]. [2010] , 01.02.2010 [cit.2010-05-05]. Dostupný z WWW: <https://developer.mozilla.org/en/A_reintroduction_to_JavaScript>.

[7] *Adobe Flash- Wikipedia, the free encyclopedia* [online]. [2010] , 05.05.2010 [cit.2010-05-05]. Dostupný z WWW: <http://en.wikipedia.org/wiki/Adobe_Flash>.

[8] *ActionScript- Wikipedia, the free encyclopedia* [online]. [2010] , 04.05.2010 [cit.2010-05-05]. Dostupný z WWW: <http://en.wikipedia.org/wiki/ActionScript>.

[9] *GRAFIKA- pracujeme s Ghostscriptem* [online]. [2010] , 05.01.2001 [cit.2010-05-05]. Dostupný z WWW: <http://www.grafika.cz/art/polygrafie/ghostscript_pod.html>.

[10] Žižka, J.: Webový prohlížeč přednášek, diplomová práce, FIT VUT v Brně, 2009

[11] *HOWTO:shift tempoin mplayer without changing pitch* [online]. [2010] , 30.06.2009 [cit.2010-05-05]. Dostupný z WWW: <http://ubuntuforums.org/showthread.php?t=1226982>.

[12] *Timed Text Markup Language (TTML) 1.0* [online]. [2010] , 23.02.2009 [cit.2010-05-05]. Dostupný z WWW: < http://www.w3.org/TR/ttaf1-dfxp/>.

[13] *Co je karma? Blog – BLOG info (blog.IDNES.cz)* [online]. [2010] , 05.01.2001 [cit.2010-05-05]. Dostupný z WWW: <http://info.blog.idnes.cz/c/56569/Co-je-to-karma.html>.

Seznam příloh

Příloha 1. Instalace knihoven pro překlad aplikace *det_slajds*

Instalace knihoven v prostředí OS Linux prostřednictvím internetu.

Instalace bude nastíněna v prostředí Ubuntu 9.10

Příprava instalace:

```
sudo apt-get remove ffmpeg
sudo apt-get update

sudo apt-get install build-essential subversion git-core checkinstall yasm texi2html libfaac-dev libfaad-dev libmp3lame-dev libopencore-amrnb-dev libopencore-amrwb-dev libsdl1.2-dev libx11-dev libxfixes-dev libxvidcore4-dev zlib1g-dev
```

Instalace FFMPEG:

```
svn checkout svn://svn.ffmpeg.org/ffmpeg/trunk ffmpeg
cd ffmpeg
./configure --enable-gpl --enable-version3 --enable-nonfree --enable-postproc --enable-pthreads --enable-libfaac --enable-libfaad --enable-libmp3lame --enable-libopencore-amrnb --enable-libopencore-amrwb --enable-libxvid --enable-x11grab
make
sudo checkinstall --pkgname=ffmpeg --pkgversion "4:0.5+svn`date +%Y%m%d`" --backup=no --default
hash ffmpeg
```

Instalace OpenCV:

```
apt-get install libavformat-dev libgtk2.0-dev pkg-config cmake libswscale-dev bzip2
wget http://sourceforge.net/projects/opencvlibrary/files/opencv-unix/2.1/OpenCV-2.1.0.tar.bz2
tar xvjf OpenCV-2.1.0.tar.bz2
cd OpenCV-2.1.0/
mkdir release
cd release
```

```
cmake -D CMAKE_BUILD_TYPE=RELEASE -D CMAKE_INSTALL_PREFIX=/usr/local
-D BUILD_PYTHON_SUPPORT=ON ..
make
make install
export LD_LIBRARY_PATH=/usr/local/lib/:$LD_LIBRARY_PATH
ldconfig
```

Instalace TAP-Plugins:

```
sudo apt-get install tap-plugins
```

Příloha 2. Architektura a aplikace rozšíření

V této části je nastíněno, jak spustit popřípadě kompilovat vytvořená rozšíření.

Kompilace programu na detekci slajdů v prostředí Linux

Všechny zdrojové kódy potřebné k přeložení programu jsou umístěny ve složce *det_slajds_src/*, jež se nachází na přiloženém CD.

Nejprve je nutné překopírovat na disk soubor *det_slajds_src.tar.gz* a rozbalit jej příkazem:

```
gunzip det_slajds_src.tar.gz; tar -xvf det_slajds_src.tar
```

Vytvořená složka nyní obsahuje Makefile, který musí být umístěn ve složce se zdrojovými soubory. Před spuštěním kompilace je nutná instalace všech potřebných knihoven, jejichž popis se nachází v příloze 1. Poté je vhodné překontrolovat defaultní cesty nastavené v Makefile.
Spuštění kompilace se provádí příkazem:

```
make
```

Při správném dodržení instalačního postupu knihoven by se měl vytvořit spouštěcí soubor s názvem *det_slajds*.

Program na detekci slajdů v prostředí Linux

Program det_slajds je umístěn na CD ve složce *det_slajds/*, kde se nachází jeho zabalená verze. Nejprve je nutné ji překopírovat na disk a následně rozbalit příkazem:

```
gunzip det_slajds.tar.gz; tar -xvf det_slajds.tar
```

Nyní je obsahem složky spustitelný soubor det_slajds a podsložka *library/* s knihovnami, které jsou potřebné ke spuštění.
Pro správné načtení knihoven je třeba ve složce s programem *det_slajds* použít příkaz:

```
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:library/
```

Tím se přilinkují soubory obsažené v *library/* do cesty pro knihovny a už nic nebrání program spustit klasicky příkazem:

```
./det_slajds
```

Pomocný skript pro det_slajds

Skript *auto_det_slajds* je lokalizován na CD ve složce *auto_det_slajds/*.

Nejprve je nutné překopírovat na disk soubor *auto_det_slajds.tar.gz* a následně rozbalit příkazem:

```
gunzip auto_det_slajds.tar.gz; tar -xvf auto_det_slajds.tar
```

Skript vyžaduje pro správné spuštění nainstalovaný GhostScript, kterým se extrahují slajdy, a samozřejmě správně nastavenou cestu k program det_slajds, jehož knihovny se nachází v podsložce *library/*.

Použití je pak zcela intuitivní, stačí nakopírovat tento skript do složky s lekcí, kde se nachází soubory pro zpracování včetně pdf souborů se slajdy a pak jej spustit příkazem:

```
./auto_det_slajds
```

V případě nejistoty lze spustit nápovědu příkazem:

```
./auto_det_slajds -h
```

Kompilace vytvořených pluginů k JW Playeru

Zdrojové kódy k pluginům včetně JW Playeru jsou umístěny na CD ve složce *player_src/*.
Pro kompilaci jednotlivých pluginů stačí otevřít FLA soubor požadovaného rozšíření v Adobe Flash CS3/CS4 a stiskem Ctrl+Enter vytvořit zkompilovaný soubor se stejným jménem a koncovkou *.swf, který slouží jako zásuvný plugin.
Pluginy vytvořené na základě mé bakalářské práce jsou:

`Notes` – poznámky k přehrávané přednášce

`Fastplay` – zrychlené přehrávání přednášek

Pozn.: pro správnou funkci pluginu Fastplay je třeba mít i upravenou verzi JW Playeru, taktéž se nacházejícího na CD ve složce *player/*, jehož vývojářská verze se skrývá pod názvem player.fla

Zakomponování pluginů k JW Playeru do projektu

Zkompilované pluginy se nachází na CD ve složce *player/*, ty lze překopírovat do projektové složky s přehrávačem a JavaScriptu přidat následující řádky:

```
mediaplayer.addVariable("plugins",base_url+"mediaplayer/fastplay,"
+base_url+"mediaplayer/next,"+base_url+"mediaplayer/captions");
```

Tím se připojí pluginy k přehrávači.

Plugin Fastplay

Pro správnou funkci pluginu fastplay je třeba nastavit proměnnou *fastplay.speeds* tímto způsobem:

```
mediaplayer.addVariable("fastplay.speeds","1.0;1.3;1.5");
```

Toto vyjadřuje všechny rychlosti, které jsou u dané přednášky k dispozici. Tyto přednášky musí být umístěny stejně jako načtený videosoubor s rychlostí 1.0x, neboli proměnná JW Playeru *file*. Soubory s různou rychlostí musí být pojmenovány tímto způsobem:

Název souboru	Rychlost přehrávání
BIN-2008-02-05.flv	1.0x
BIN-2008-02-05-13.flv	1.3x
BIN-2008-02-05-15.flv	1.5x

Tabulka 1: Souvislost mezi názvem souboru a rychlosti přehrávání

Ve výše položené tabulce (Tabulka 1) je nastíněno, že název záznamu pro rychlost 1.0x je zachován pro kompatibilitu s přehrávačem JW Player bez použití pluginu. Zbytek má připojenou rychlost za pomlčkou, přičemž tečka se vynechává. Plugin při změně rychlosti generuje JS volání **SetSpeed(***playing_speed***)**, kde argument *playing_speed* obsahuje aktuální rychlost přehrávání. Pro správné posouvání ve videu (seek) je třeba násobit čas přesunu rychlostí přehrávání.

Plugin Notes

Plugin Notes pro zobrazení komentářů potřebuje vstup ve formě TTML[13], který byl použit pro transskripty. Tento vstup je generován z databáze pomocí PHP skriptu *data.php*, umístěného taktéž ve složce *player/*. Důležité je také předání tohoto vstupu JW Playeru.

```
mediaplayer.addVariable("notes.file",base_url+"mediaplayer/data
.php?lect="+lecture_id);
```

kde *lecture_id* je identifikátor přednášky.