VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ZPRACOVÁNÍ ŘEČOVÝCH SIGNÁLŮ PŘES WEBOVÉ ROZHRANÍ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

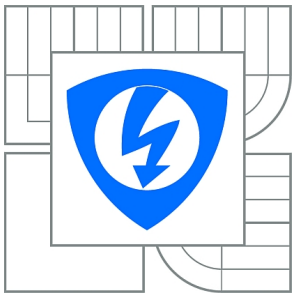
AUTOR PRÁCE
AUTHOR

MICHAL MARKOVIČ

BRNO 2015

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV TELEKOMUNIKACÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF TELECOMMUNICATIONS

ZPRACOVÁNÍ ŘEČOVÝCH SIGNÁLŮ PŘES WEBOVÉ ROZHRANÍ

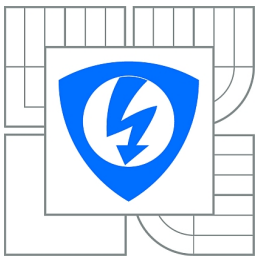
SPEECH SIGNAL PROCESSING VIA WEB INTERFACE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — MICHAL MARKOVIČ
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ MEKYSKA, Ph.D.
SUPERVISOR

BRNO 2015



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav telekomunikací

Bakalářská práce

bakalářský studijní obor
Teleinformatika

Student:	Michal Markovič	***ID:***	159149
Ročník:	3	***Akademický rok:***	2014/2015

NÁZEV TÉMATU:

Zpracování řečových signálů přes webové rozhraní

POKYNY PRO VYPRACOVÁNÍ:

V rámci této práce bude navržen a implementován systém, který uživateli umožní přes webové rozhraní zaznamenat řeč, nahrát ji na server, zpracovat a nakonec pomocí rozhraní zobrazit výsledky z analýzy. Systém bude schopen simultánně zpracovávat požadavky od více uživatelů. K implementaci rozhraní může být využito jazyků JAVA, PHP, MySQL, HTML5. K samotnému zpracování signálů na straně serveru bude pak využito jazyka C/C++.

DOPORUČENÁ LITERATURA:

[1] LUBBERS, P. HTML5: programujeme moderní webové aplikace. Brno: Computer Press, 2011. 304 s. ISBN 978-80-251-3539-6.
[2] GILMORE, W. J. Velká kniha PHP 5 a MySQL: kompendium znalostí pro začátečníky i profesionály. Brno: Zoner Press, 2011. 736 s. ISBN 978-80-7413-163-9.
[3] LIBERTY, J. Naučte se C++ za 21 dní. Brno: Computer Press, 2007. 796 s. ISBN 978-80-251-1583-1.

Termín zadání:	9.2.2015	***Termín odevzdání:***	2.6.2015

Vedoucí práce: Ing. Jiří Mekyska, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Mišurec, CSc.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT

Táto práca rieši možnosť spracovania rečového signálu pomocou webového rozhrania. Rečový signál je zaznamenaný do nahrávky, ktorá je uložená na servery. Nahrávku je možné ďalej pomocou webovej aplikácie, písanej v programovacom jazyku C++, analyzovať a spracovávať. Výsledky sú vykreslené späť vo webovom rozhraní.

KĽÚČOVÉ SLOVÁ

HTML, CSS, JavaScript, HTML5 API, C++, PHP, webové rozhranie, webový server, Web Audio API, CppCMS, WAV, web framework

ABSTRACT

This work is resolving the possibility of processing a speech signal by using the web interface. A record of the speech signal is made and then saved and stored in servers. Then it's possible to analyse and process the record with the help of a web application written in the C++ programming language. The results are shown in website.

KEYWORDS

HTML, CSS, JavaScript, HTML5 API, C++, PHP, web interface, web server, Web Audio API, CppCMS, WAV, web framework

MARKOVIČ, Michal *Zpracování řečových signálů přes webové rozhraní* : bakalárska práca. BRNO: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2015. 43 s. Vedúci práce bol Ing. Jiří Mekyska, Ph.D.

PREHLÁSENIE

Prehlasujem, že som svoju bakalársku prácu na tému „Zpracování řečových signálů přes webové rozhraní “ vypracoval(a) samostatne pod vedením vedúceho bakalárskej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej bakalárskej práce ďalej prehlasujem, že v súvislosti s vytvorením tejto bakalárskej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/nebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona č. 121/2000 Sb., o právu autorském, o právoch súvisejúcich s právom autorským a o zmeně niektorých zákonov (autorský zákon), vo znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhé, hlavy VI. diel 4 Trestného zákoníka č. 40/2009 Sb.

BRNO ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som poďakoval vedúcemu bakalárskej práce pánovi Ing. Jiřímu Mekyskovi, Ph.D. za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

BRNO ....

podpis autora(-ky)

Faculty of Electrical Engineering
and Communication
Brno University of Technology
Purkynova 118, CZ-61200 Brno
Czech Republic

http://www.six.feec.vutbr.cz

POĎAKOVANIE

Výzkum popsaný v tejto bakalárskej práci bol realizovaný v laboratóriách podporených projektom SIX; registračné číslo CZ.1.05/2.1.00/03.0072, operačný program Výzkum a vývoj pro inovace.

BRNO ....

podpis autora(-ky)

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ÚVOD

V dnešnej dobe je pripojenie na internet takmer nevyhnutnosť. Webové aplikácie majú voči desktopovým aplikáciam niekoľko výhod. Je možné sa pripojiť z ktoréhokoľvek miesta s prístupom na internet a k pripojeniu je potrebný len internetový prehliadač. Nakoľko prehliadače majú podporu na mnohých platformách, či už desktopových alebo mobilných zariadeniach, webové aplikácie sa stávajú nezávislé na type platformy. Túto výhodu sa snažia vývojári využiť, a preto momentálne vo veľkom implementujú rozhrania do internetových prehliadačov, ktoré webové aplikácie čoraz viac približuje k desktopovým aplikáciam.

V práci sa rieši problematika nahrávania zvukového záznamu na server kde beží webová aplikácia, ktorá uloženú nahrávku bude schopná analyzovať a ďalej spracovávať. Preto je potrebné si prácu rozdeliť do niekoľkých hlavných častí.

V prvej časti sú uvedené možnosti spracovania zvukového záznamu z mikrofónu pomocou internetového prehliadača. Ako už bolo spomenuté, v internetových prehliadačoch je implementovaných množstvo nových rozhraní, ako sú aj HTML5 API rozhrania na spracovanie audio signálu (getUserMedia, WebAudiAPI). Bez nich by bolo potrebné využívať zásuvné moduly, ktoré nie sú praktické z mnohých dôvodov.

V druhej časti práce je riešená možnosť spracovania signálu z mikrofónu, kde je potrebné previesť navzorkovaný signál do formátu WAV a poslať na server. To je riešené pomocou jednoduchej funkcie v jazyku PHP.

V tretej časti je navrhnuté riešenie problému spracovania analýzy rečového signálu. Webové rozhranie načíta vybraný zvukový súbor a pomocou implementovaných funkcií zanalyzuje signál. Výsledok sa vykreslí v internetovom prehliadači, kde sa zobrazí pomocou čísel, grafov, poprípade tabuliek. Pre tento typ aplikácie je potrebné neplytvať výkonom, nakoľko programy na spracovanie zvuku sú náročné na výpočetnú techniku. Z tohto dôvodu je navrhnuté webové rozhranie v C++ jazyku, ktorý oproti iným objektovým jazykom šetrí výkon. V tejto časti je vybraný najvhodnejší framework pre prepojenie HTML stránky s programovací jazykom C++ a ukážka implementácie funkcií pomocou tohoto frameworku.

V predposlednej časti sa nachádza už komplexný návrh a funkčnosť konkrétneho naimplementovaného programu. Program je rozdelený do troch častí. Nahrávanie zvuku, analýza a výsledky analýzy. V tejto časti je možné nájsť aj podrobnú dokumentáciu funkčnosti programu.

V poslednej časti je opísané, ako je možné spustiť aplikáciu na lokálnom serveri. Sú spomenuté najdôležitejšie konfiguračné údaje aj s možnosťou, ako upraviť program a opätovne ho skompilovať.

1 TECHNOLÓGIE NA TVORBU WEBOVÝCH APLIKÁCIÍ

Dnešné moderné webové aplikácie sú tvorené hlavne tromi základnými technológiami. Hypertextovým značkovacím jazykom HTML, kaskádovým štýlom CSS, ktorý sa stará o vzhľad stránky, a skriptovacím jazykom JavaScript. Moderné webové prehliadače umožňujú skombinovať tieto technológie a vzniknú multifunkčné webové aplikácie, ktoré nie sú závislé na operačných systémoch a platformách zariadení.

V ďalšej časti práce budú detailnejšie prebraté tieto webové technológie, hlavne rozhrania zamerané na spracovanie zvukových signálov.

1.1 Hypertextový značkovací jazyk HTML

HTML je značkovací jazyk, ktorý umožňuje vytvárať internetové stránky a vkladať do nich obsah zobraziteľný v internetovom prehliadači ako napríklad hypertextové odkazy, formuláre, prostý text, multimediálny obsah ale aj metainformácie, ktoré slúžia internetovým vyhľadávačom.

Začiatky jazyka HTML siahajú do 90 rokov minulého storočia, kedy HTML prinieslo postupne verzie 2.0, 3.2, 4.0 a v roku 1999 konečnú verziu 4.01. Na začiatku tohoto tisícročia sa považoval jazyk HTML skôr za slepú uličku. Až v roku 2004 založila malá skupina ľudí, v snahe povýšiť webovú platformu na novú úroveň, skupinu WHATWG (Web Hypertext Application Working Group). Vytvorili špecifikáciu HTML5, ktorá v dobe plnej statických stránok umožnila vytvoriť viac dynamické webové stránky. HTML5 okrem sémantických prvkov ďalej definuje aj nové rozhrania (Web Audio API, Web Socket,...), pomocou ktorých je možné spracovávať audio záznam bez inštalácie flash playeru. Nakoľko HTML5 je už finálna verzia, ale väčšina jej aplikácií sú ešte pracovnou verziou a špecifikácie ešte nie sú definitívne ustanovené, nie je potrebné sa obávať kompatibility a funkčnosti jednotlivých rozhraní v internetových prehliadačoch. Tvorci prehliadačov horúčkovito implementujú rozhrania. Momentálnu podporu HTML5 API v prehliadačoch možno overiť na stránke „Can I USE[1]“. Pri každej novšej verzii prehliadačov sa však ponuka podporovaných rozhraní rozširuje.[2]

Webový prehliadač sa čoraz viac stáva ďalšou platformou pre aplikácie. Desktopové a mobilné zariadenia umožňujú webovým prehliadačom skladovať údaje a spúšťať aplikácie v offline režime, čo webové prehliadače približuje k iným platformám. Vďaka tomu, že prehliadače na všetkých moderných elektronických zariadeniach podporujú HTML5, aplikácia vytvorená pomocou HTML5 sa stáva multiplatformovou.

1.2 Kaskádové štýly CSS

CSS je skratka pre Cascading Style Sheet, takže slovensky Kaskádové štýly. CSS je všeobecné rozšírenie HTML a zatiaľčo HTML určuje obsah stránky, CSS definuje jej formu. Možnosť oddeliť štruktúru od vzhľadu je veľmi elegantné riešenie, získa sa tým prehľadnosť kódu, nakoľko je možné vytvoriť samostatný súbor, ktorý bude definovať vzhľad pre celú webovú stránku. Názov kaskádové je zavedený z dôvodu, že je možné na seba vrstviť definície štýlu, ale platí vždy ta posledná.

CSS vzniklo okolo roku 1997. Je to súhrn metód pre grafickú úpravu web stránok. Na začiatku umožňovalo prácu s písmami, okrajmi a farbami. Najnovšie vydanie je CSS3, ktoré rovnako ako HTML5 je stále pracovnou verziou skupiny W3C. Pomocou nových funkcií, ktoré CSS3 prináša, je možné animovať aj priamo elementy dokumentu HTML. Aj vďaka tomuto sa stáva CSS3 viac a viac populárnym, pretože niektoré jednoduché zmeny štýlov nebolo možné realizovať bez použitia JavaScript.[3]

1.3 JavaScript

JavaScript je objektovo orientovaný skriptovací jazyk používaný pre vývoj internetových aplikácií. Len pomocou HTML je možné vytvoriť iba statické stránky, nie však internetové aplikácie. Aby bolo možne spraviť zo stránky vytvorenej pomocou HTML5 dynamickú webovú aplikáciu, je potrebné použiť jazyk JavaScript.

Programovací jazyk JavaScript vytvorila koncom minulého storočia spoločnosť Netscape. Už z názvu je možné vidieť spojitosť s jazykom Java, táto predstava je však klamlivá, pretože jediné čo tieto dva jazyky majú spoločné je podobná syntax, nakoľko ich tvorcovia boli inšpirovaný spoločným jazykom. Vývojári JavaScriptu sa rozhodli pre tento názov hlavne z marketingového hľadiska, keďže v tom období zažíval veľkú popularitu jazyk Java.

Jednou z najdôležitejších vlastností JavaScriptu je fakt, že ide o programovací jazyk, jeho činnosť zabezpečuje prehliadač internetových stránok návštevníka stránky a nie server, na ktorom sú stránky uložené. Z tohto faktu plynie niekoľko omedzení. Dlho nebolo možné pomocou JavaScriptu komunikovať so serverom, čo spôsobovalo vývojárom webových aplikácii značné problémy, pretože nebolo možne vytvoriť jednoduché aplikácie ako počítadlo návštev, fórum, knihu návštev. Preto bol navrhnutý JavaScript objekt XMLHttpRequest. Poskytuje jednoduchý spôsob, ako načítať dáta z adresy URL, bez toho aby muselo dôjsť k obnoveniu stránky.[4]

Pomocou opísanej trojice HTML5, CSS3 a JavaScript je možné vytvoriť zložité webové aplikácie. Prehliadače sa čoraz viac stávajú platformami, a nakoľko moderné prehliadače sú implementované v takmer všetkých elektronických zariadeniach, internetové aplikácie sa stávajú veľmi silným nástrojom vývojárov.

1.4 Možnosti nahrávania zvuku v prehliadači

Moderné internetové stránky, okrem statického a dynamického obsahu, môžu obsahovať aj aplikácie na spracovávanie videa a zvuku. Jedná sa napríklad o jednoduchý audio prehrávač (internetové rádio, YouTube,...), ale aj o aplikáciu možnú nahrávať, zaznamenávať a ďalej spracovávať audio signál. Preto, aby bol prehliadač schopný nahrávať audio záznam je možné použiť tzv. zásuvný modul (Plug-in).

Zásuvné moduly sú aplikácie, ktoré môžu byť ľahko inštalované a používané ako súčasť internetového prehliadača. V začiatkoch, ešte na prehliadači Netscape, bolo možné stiahnuť a nainštalovať doplnkový program, ktorý prehrával zvuk alebo video. Tieto programy však bežali ako samostatné aplikácie a vyžadovali aby sa otvorilo nové okno prehliadača. Dnešný zásuvný modul je rozpoznaný prehliadačom a je automaticky vložený do hlavného súboru HTML, ktorým je prekladaný. Jedným z najznámejších a najpoužívanejších zásuvných modulov na zobrazenie alebo zdieľanie videa a zvuku je v súčasnej dobe Adobe Flash Player.[5]

Adobe Flash player je softvér používaný na spracovávanie zvuku v internetovom prehliadači. Flash Player bol vytvorený spoločnosťou Macromedia, ale teraz je vyvíjaný a distribuovaný spoločnosťou Adobe Systems Inc. Flash Player plug-in je možné stiahnuť zadarmo na novšie verzie internetových prehliadačov. Výhodou je veľké množstvo podporovaných multimediálnych formátov ako je napríklad mp3, FLV, PNG, JPEG, GIF.[6]

Podobným zásuvným modulom ako je Adobe Flash Player je aplikačná platforma Silverlight od spoločnosti Microsoft. Je určená okrem iného pre vývoj multimediálnych aplikácii a ako vývojové prostredie sa zvyčajne používa Visual Studio.

Výhodou zásuvných modulov môže byť ich dobrá podpora v internetových prehliadačoch, a kvalitná podpora spracovania multimediálnych formátov. Ďalej je dobré, že je ich možné stiahnuť k internetovým prehliadačom zdarma, ale ich hlavnou nevýhodou je práve to, že nie sú implementované v prehliadačoch. Užívatelia neradi inštalujú a používajú externé aplikácie, zdá sa im to zdĺhavé a možno aj rizikové. Preto momentálne pravdepodobne najlepším riešením ako spracovať audio vo webovej aplikácii je pomocou HTML5 aplikácii ako sú getUserMedia a WebAudio Api. Kde pomocou implementovaných funkcií cez JavaScript je možné priamo v HTML5 spracovávať zvuk, bez toho aby užívateľ musel niečo inštalovať a povoľovať v prehliadači.

2 SPRACOVANIE ZVUKU POMOCOU HTML5 API

Ako už bolo spomenuté, po dlhú dobu sa používali pre prístup k audio zariadeniam (webkamere a mikrofónu) pluginy (Flash, Silverlight). HTML5 však okrem iného priniesol aj väčšiu možnosť pristupovať k zariadeniam (hardwaru). Príkladmi sú orientation API (akcelerometer), WebGL (GPU), geolokácia (GPS)[7], ale aj Web Audio API. Tieto funkcie sú ukážkou, že pomocou HTML5 je možné dosiahnuť prístup a priamo spracovávať dáta zo senzorov pripojených k zariadeniu.

2.1 Rozhranie getUserMedia()

Rozhranie `getUserMedia API` umožňuje získať priamy prístup k multimediálnym streamom (kamera, mikrofón). K naviazaniu kontaktu s mikrofónom teraz už nie je potrebné inštalovať plugin, stačí povoliť prístup k mikrofónu pre webovú aplikáciu. Rozhranie `getUserMedia API` je stále pracovným návrhom organizácie W3C, takže špecifikácie rozhrania môžu prejsť zmenami [8].

Metoda `getUserMedia` nie je zatiaľ podporovaná vo všetkých webových prehliadačoch. Momentálne je podporovaná prehliadačom Chrome (od verzie 21), Firefox (od verzie 17) a Opera (od verzie 12). Ďalej je rozhranie implementované aj na väčšine mobilných prehliadačov (Firefox a Chrome pre android, Opera mobile...)[9]. Z tohoto dôvodu výrobci prehliadačov pripojujú k rozhraniu `getUserMedia` tzv. vendor prefix (webkit, moz...). Toto je možné jednoducho vyriešiť vložením kódu do scriptu, ktorý umožní funkčnosť na rôznych prehliadačoch.

```
navigator.getUserMedia = ( navigator.getUserMedia ||
                           navigator.webkitGetUserMedia ||
                           navigator.mozGetUserMedia ||
                           navigator.msGetUserMedia);

if (navigator.getUserMedia) {

        //prehliadač rozhranie podporuje, je možné ju implementovať

}else {

            alert ('getUserMedia() nie je vo vašom prehliadači
            podporovaný - prosím skúste Chrome alebo Firefox');
}
```

2.1.1 Získanie prístupu k zariadeniu

Syntax rozhrania `getUserMedia()` je `navigator.getUserMedia(constraints, successCallback, errorCallback)`. Pred použitím webovej kamery alebo mikrofónu, je potrebné žiadať o povolenie. Prvý parameter `constraints` je objekt, ktorý určuje, ku akému typu média sa bude pristupovať. Napríklad, ak je potrebné pripojiť webovú kameru, prvý parameter by mal vyzerať takto: `video:true`. Ak je vyžadované použiť aj mikrofón, parameter `constraints` bude vyzerať takto: `video:true, audio:true`. Tento parameter je povinný a je vrátený vo forme objektu vo funkcií `successCallback`.[10]

Druhý parameter `successCallback` ako už bolo spomenuté volá prvý parameter `constraints`, aby získal prístup ku mediálnym zariadeniam. Tento objekt obsahuje dátový prúd médií,ktorý sa môže priradiť k príslušnému prvku a ďalej spracovávať.

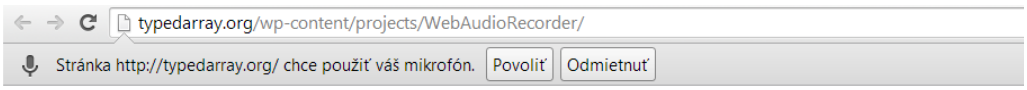
Tretí parameter `errorCallback` volá chybové volania ktoré sú popísané v tab. 2.1.

Tab. 2.1: Chybové volania parametra `errorCallback`.

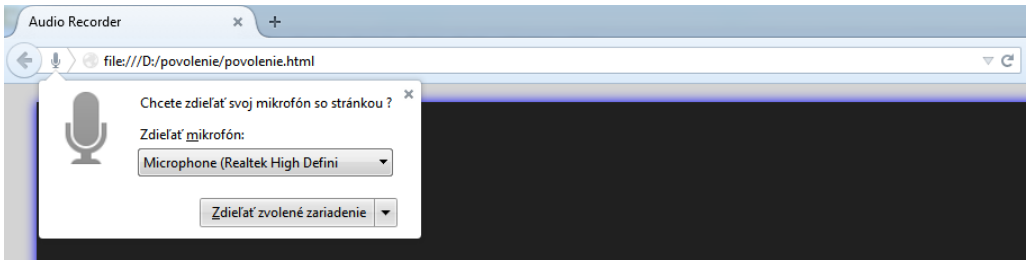
Chyba	Popis
PERMISSION_DENIED	Užívateľom odopreté oprávnenie na používanie zariadenia potrebného k prevádzke.
NOT_SUPPORTED_ERROR	`getUserMedia()` nie je podporovaný prehliadačom.
MANDATORY_UNSATISFIED	Žiadne médium typu uvedeného v `constraints` nie je prístupné.

2.1.2 Zabezpečenie

Predtým ako sa začne zaznamenávať video/zvuk, na väčšine prehliadačov vyskočí informačný panel, čo dáva možnosť používateľom udeliť alebo odoprieť prístup ku svojej kamere/mikrofónu. Napríklad, na obrázku 2.1 je zobrazené dialógové povolenie prehliadača Chrome, na obrázku 2.2 je dialógové okno povolenia Firefox. Ak však



Obr. 2.1: povolenie prístupu k médiam, prehliadač Chrome



Obr. 2.2: povolenie prístupu k médiam, prehliadač Firefox

aplikácia beží pod zabezpečeným protokolom https, toto oprávnenie bude trvalé. To znamená, že používatelia nebudú musieť udeliť/zakázať prístup zakaždým.

2.2 Rozhranie Web Audio API

Ďalším rozhraním HTML5 je `Web Audio API`. Získaný audio signál z rozhrania `getUserMedia API`, ktorý je odovzdaný ako parameter funkcie spätného volania pri úspešnom načítaní (successCallback), sa spracováva pomocou rozhrania `Web Audio API`.

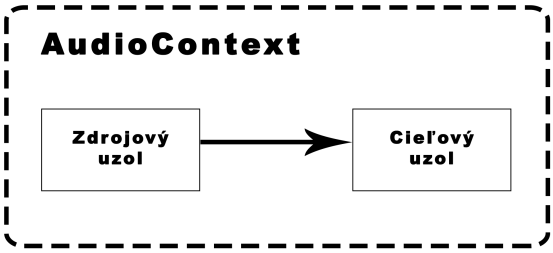
Rozhranie `Web Audio API` je vysoko-úrovňový JavaScript API pre spracovanie zvuku vo webových aplikáciach. Hlavným cieľom tohto rozhrania je implementovať funkcie používané v moderných herných enginoch a umožniť mixovanie, spracovanie a filtrovanie, ktoré je možné vidieť v desktopových audio aplikáciách. Výsledkom je komplexné rozhranie umožňujúce vývojárom vybrať zdroj audia, pridávať audio efekty, vytvárať vizualizácie zvuku, a oveľa viac.

Základnou stavebnou jednotkou rozhrania `Web Audio API` je `AudioContext`. Je určený pre správu a prehrávanie všetkých zvukov. Je to pomyselný graf audio uzlov, ktoré pôsobia ako moduly pre spracovanie signálu. Inštancia `AudioContext` podpotuje viac zvukových vstupov, takže je potrebné vytvoriť pre každú audio aplikáciu len jednu inštanciu. Opäť bude potrebné použiť vendor prefixy.

```
var createContext;
window.addEventListener('load', init, false);
function init() {
 try {
  window.AudioContext=window.AudioContext||window.webkitAudioContext;
  createContext=new AudioContext();
  }  catch(e) {
  alert('Web Audio API nie je podporovaná vo vašom prehliadači);  }
}
```

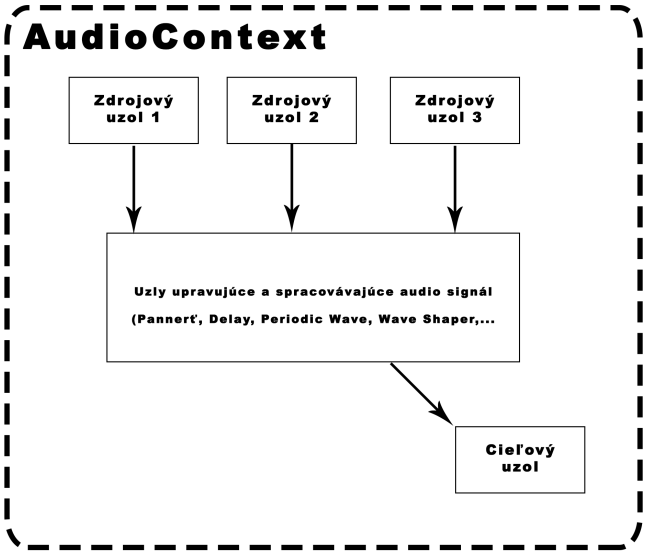
Momentálne rozhranie, podobne ako `getUserMedia`, nie je podporované vo všetkých prehliadačoch, nakoľko je to ešte stále stále pracovným návrhom organizácie W3C. V čase písania práce bolo podporované prehliadačmi Firefox, Chrome, Safari, Opera. [11]

`AudioContext`, teda pomyselný graf audio uzlov určuje ako zvukový signál prechádza zo zdroja (audio súbor alebo živý stream), až do cieľového uzla (vytvorenie nahrávky, reproduktory zariadenia). Audio signál počas tejto cesty prechádza jednotlivými uzlami, ktoré môžu jeho vlastnosti modifikovať alebo kontrolovať. Každý uzol môže mať vstupy a výstupy. Zdrojový uzol nemá žiadne vstupy a jeden výstup. Podobne cieľový uzol má jeden vstup a žiaden výstup. Najjednoduchší `AudioContext` je, keď sa prepojí priamo zdrojový uzol s cieľovým uzlom obr. 2.3. Príklad zloži-



Obr. 2.3: Najjednoduchší audioContext

tého `AudioContextu`, ktorý obsahuje 3 vstupy, vykonávajúceho ľubovolné funkcie so zvukovým médiom je znázornený na obrázku 2.4



Obr. 2.4: Zložitý audioContext

Ukážka jednoduchého prehratia zvuku by vyzerala nejako takto.

```
var context = new AudioContext();

function playAudio(buffer) {
  var source = context.createBufferSource();
  source.buffer = buffer;
  source.connect(context.destination);
  source.start(0);
}
```

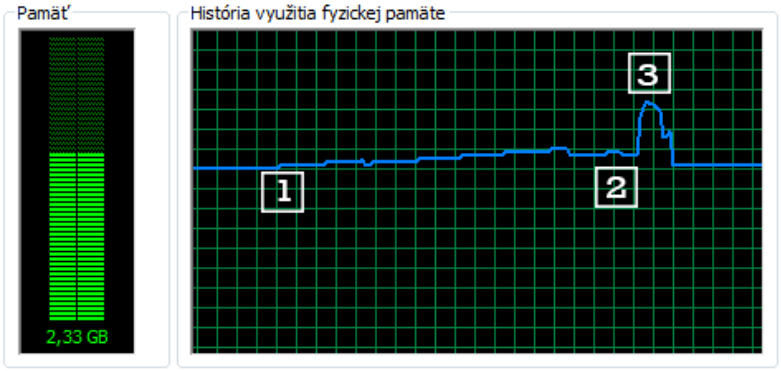
Akonáhle je jeden alebo viac `audioBuffers` načítaných, je možné prehrať zvuk. Funkcia `playAudio` môže byť volaná napríklad po stlačení tlačítka PLAY. V ukážke je použitá metóda `connect()`, ktorá slúži na prepojenie výstupu jedného uzla so vstupom druhého uzla. V tomto prípade je vstup (buffer) prepojený priamo na výstup (destination). Metóda start() už len spustí prehrávanie v presne stanovenom čase.[12]

Veľmi požadovaný a užitočný prvok rozhrania `Web Audio API` je možnosť komunikácie s rozhraním `getUserMedia`, ktorý umožní prehliadaču prístup k streamu z pripojeného mikrofónu. K tomuto slúži funkcia `MediaStreamSourceNode()`, ktorá získa prístup k audiostremu a vytvorí zdrojový uzol pre `AudioContext`.[13]

```
var AudioInput = context.createMediaStreamSource(stream);
```

Ďalším potrebným rozhraním `Web Audio API` je `ScriptProcessorNode`, ktoré umožní spracovávať audio priamo pomocou JavaScript príkazov. Materskou triedou je `AudioNode`. Pozostáva z dvoch vyrovnávacích pamätí (input buffer, output buffer), z ktorých prvá obsahuje vstupné audio dáta a druhá obsahuje výstupné, spracované dáta. Vyrovnávacia pamäť musí byť definovaná jednou z hodnôt 256, 512, 1024, 2048, 4096, 8192 alebo 16384. Malá vyrovnávacia pamäť znižuje latenciu, veľká znižuje možnosť rozpadnutia a závady audio streamu. Hodnota vyrovnávacej pamäte určuje počet vzoriek signálu na jeden rámec. Po naplnení rámca za zavolá udalosť onaudioprocess z rozhrania `ScriptProcessorNode`, ktorá pomocou metódy `GetChannelData()` z rozhrania `AudioBuffer` vráti pole (Float32Array) obsahujúce vzorkované dáta. Takto sa napĺňa pamäť rámcami, pokiaľ užívateľ nepreruší nahrávanie. Vyrovnávacia pamäť sa ukladá do pamäte prehliadača, a preto je možné pozorovať významný nárast využívanej pamäte prehliadača v zariadení, narastajúcej dĺžkou nahrávania obr. 2.5. Pre vytvorenie `ScriptProcessorNode` sa zavolá metóda `CreateScriptProcessor()`, ktorá obsahuje okrem veľkosti vyrovnávacej pamäte ďalšie dva parametre, počet vstupných a výstupných kanálov.[14]

```
var nacitanie = context.createScriptProcessor(veľkosťPamäte,
početVstupnýchKanálov, početVýstupnýchKanálov);
```



Obr. 2.5: Využitie pamäte počas nahrávania audio záznamu. 1: začiatok nahrania, 2: ukončenie nahrávania, 3: Spracovanie a odoslanie audio záznamu na server

2.3 Zasielanie dát na server

Po ukončení nahrávania audio záznamu je v JavaScript aplikácii uložené pole rámcov, ktoré obsahuje navzorkované dáta. Toto pole je potrebné zapuzdriť do formátu WAV, ktorý ďalej bude možné poslať na server k neskoršiemu spracovaniu. Na výmenu dát medzi užívateľom a serverom slúži JavaScript objekt `XMLHttpRequest`.

`XMLHttpRequest` je JavaScript objekt, ktory bol navrhnutý spoločnosťou Microsoft a prijatý Mozillou, Applom a Googlom. V súčastnej dobe je štandardom skupiny W3C. Poskytuje jednoduchý spôsob načítania dát z adresy URL bez toho, aby bolo potrebné obnoviť internetovú stránku. Dokáže aktualizovať len časť stránky.[15] Pre prenos dát používa protokol HTTP (HTTPS), ale nie je to podmienkou. Odosielanie súborov pomocou `XMLHttpRequest` je možné dvomi spôsobmi:

1. pomocou technológie AJAX
2. pomocou rozhrania FormData API

Druhá možnosť je jednoduchšia a rýchlejšia. FormData objekty poskytujú spôsob, ako jednoducho vytvárať súbor predstavujúci pole formulárov, ktoré potom môžu byť ľahko odoslané. Pomocou metódy `open()` sa nastaví typ požiadavky a url adresa súboru kde sa dáta majú zaslať a nakoniec pomocou metódy `send()` sa zašlú dáta. Jeden z príkladov zaslania audio súboru na server môže byť pomocou PHP súboru, kde by zaslanie vyzeralo nasledujúco:

```
var formData = new FormData();
formData.append(audio + '-filename', test.wav);
formData.append(audio + '-blob', DATA);
var xhr = new XMLHttpRequest();
xhr.open('POST', save.php);
xhr.send(formData);
```

Dáta sú predané pomocou rozhrania formData objektu `XMLHttpRequest`, ktorému je nastavená požiadavka na zaslanie (POST) a url adresa PHP súboru, ktorý nahrá audio súbor na server. Jednoduchá ukážka PHP súboru by vyzerala napríklad takto:

```
<?
                if (isset($_FILES["${type}-blob"])) {

    $fileName = $_POST["${type}-filename"];
    $uploadDirectory = 'uploads/'.$fileName;

    move_uploaded_file($_FILES["${type}-blob"]["tmp_name"],
    $uploadDirectory))
} ?>
```

2.3.1 WebSocket API

Rozhranie `WebSocket API` umožňuje internetovým aplikáciam obojsmernú komunikáciu medzi klientom a serverom v reálnom čase. Je to rozhranie HTML5 API ktoré umožňuje aplikácií kontrolovať WebSocket protokol. Je vyvynuté organizáciou W3C. [16]

Protokol WebSocket je navrhnutý tak, aby čo najlepšie vyhovoval existujúcej internetovej infraštruktúre. Pripojenie WebSocket začína ako spojenie HTTP a po úspešnom vytvorení spojenia dáta môžu posielať ako klient tak aj server navzájom samostatne. Používa rovnaké porty ako HTTP (80) a pre šifrovaný prenos ako HTTPS(443).

3 WEBOVÉ ROZHRANIA

V minulosti aplikácie, ktoré boli typu klient–server, mali vlastný klientský program. Ten bolo potrebné pred používaním aplikácie nainštalovať na zariadenie klienta. Slúžil ako užívateľské rozhranie pre aplikáciu, ktorá bežala na servery. Bolo to však neefektívne už len z pohľadu ak vyšla nová verzia programu bolo potrebné aktualizovať aj užívateľské rozhranie na všetkých počítačoch.

Ako už bolo niekoľkokrát spomenuté, tento problém sa dá riešiť elegantným spôsobom a to pomocou JavaScript jazyka, ktorý generuje HTML kód. Stránka na prvý pohľad vyzerá staticky ale ďalej pomocou spomínaného rozhrania `HMLHttpRequest` je možné napríklad načítavať užívateľské vstupy zadané do internetového prehliadača a ďalej ich spracovávať. Ďalšou veľkou výhodou je, že užívateľ nemusí nič inštalovať a narozdiel od desktopových programov, kde je treba riešiť funkčnosť na rôznych operačných systémoch, webová aplikácia beží v internetovom prehliadači ktorý tento problém už vyriešil. Dáta sú uchovávané na servery a teda užívateľ nemusí nič zálohovať a sú mu prístupné z ktoréhokoľvek zariadenia pripojené na internet. Nutnosť pripojenie na internet môže byť jedna z mála nevýhod, no vzhľadom nato, že dnes je už takmer povinnosť pripojenie k internetu, táto nevýhoda sa stráca.

Webová aplikácia je teda taká aplikácia, ktorú nie je potrebné inštalovať na zariadenie užívateľa a je ju možné spustiť z ktoréhokoľvek zariadenia, ktoré podporuje webový prehliadač, pretože beží na servere.[17]

Napriek mnohým výhodám webové aplikácie vyvolávajú mnohé bezpečnostné obavy. Mnohé webové aplikácie uchovávajú v databázach na serveri citlivé dáta, ako napríklad databázy zákazníkov s osobnými údajmi, číslami kreditných kárt, prihlasovacími údajmi,... Nakoľko webové aplikácie musia byť nepretržite prístupné pre všetkých užívateľov internetu (internetové obchody), Firewall neposkytuje žiadnu ochranu proti útočníkom. Preto sa využíva množstvo techník ako zabrániť nepovolenému vniknutiu, ako je blokovanie IP adries, pre prístup do databázy sa použije šifrovaný komunikačný protokol HTTPS a ďalšie.

3.1 Web framework

Web application framework je softwarová štruktúra, ktorá slúži pre podporu rozvoja dynamických webových stránok a internetových aplikácií. Framework si kladie za cieľ uľahčiť prácu webovým vývojárom. Väčšinou je vo forme knihovien, ktoré sú implementované do programovacieho jazyka. Knihovne zahŕňajú funkcie na prácu s HTML, JavaScript a inými internetovými technológiami. Webové frameworky sú teda nadstavby programovacích jazykov. Najproduktívnejšie sú jazyky ako Python,

PHP, Java, C#. Pokiaľ však je potrebné vo webovej aplikácii veľkú výpočetnú techniku, je dobre využiť menej náročné jazyky na výkon ako sú C/C++, Assembler.

Nakoľko spracovanie veľkých audio súborov je náročné na výkon, pre testovanú aplikáciu bol zvolený jazyk C++.

3.1.1 CppCMS

CppCMS je vysokovýkonný Web framework zameraný na Rapid Application Development, čo umožňuje vizuálny návrh aplikácie bez písania kódu. Uľahčuje to a veľmi urýchľuje prácu programátorom, nakoľko nemusia pre každý komponent (tlačítko, správu,...) písať kód, stačí ho graficky vložiť do pripravenej vrstvy. Základné vlastnosti CppCMS sú:

1. Je navrhnutý a vyladený pre spracovávanie extrémne náročných programov.
2. K tomu aby dosiahol prvého bodu je postavený na jazyku C++
3. Je určený pre vývoj ako webových stránok tak webových služieb

Hlavným dôvodom prečo si vybrať práve tento framework je výkon. CppCMS umožňuje efektívne spracovávať stovky až tisíce HTTP pripojení s využitím minimálnych nárokov na výkon. Jedným z možných rizík by mohlo byť zabezpečenie, avšak vývojári ubezpečujú, že moderné C++ aplikácie, sú bezpečné podobne ako aplikácie Java alebo C#.

CppCMS napríklad využíva webová služba Picasa.net, ktorá umožňuje zdielať fotky a preto sú kladené veľké nároky na výkon.[18]

3.1.2 C++ REST SDK

C++ REST SDK, tiež známy ako Casablanca, je open source projekt spoločnosti Microsoft. Slúži pre komunikáciu medzi klientom a serverom, je postavený na modernom asynchrónnom C++ dizajne. Využíva úloh na báze programovacieho modelu PPL.

Obsahuje vlastnosti HTTP klient/server, JSON, URI, rozhranie WebSocket. Má podporu vo Visual Studio 2012 a 2013 so všetkými ladiacimi možnosťami, čo uľahčuje prácu webovým vývojárom.[19]

3.1.3 SWILL

SWILL (Simple Web Interface Link Library) je knihovňa, ktorá umožňuje pomerne ľahko vytvárať webové aplikácie napísané v C/C++. Umožňuje aplikáciam fungovať ako jednoduchý server, na ktorý sa je možne pomocou prehliadača pripojiť. Opäť je vhodný hlavne do aplikácii kde je kladený veľký dôraz na výpočetnú techniku.

Vo svojej podstate SWILL nie je teda nič viac ako jednoduchý integrovaný webový server, ktorý poskytuje nasledujúcu sadu funkcií:

- Podporu pre HTTP, GET a POST operácie
- Prístup k premenným formulára HTML
- Bezpečnostné mechanizmy. IP filtre, základné autentizácie
- Podpora SSL pomocou knižnice Open SSL

Narozdiel od webových serverov, SWILL je navrhnutý tak aby sa pridal do existujúcich programov. Je primárne určený ako doplnok.[20]

3.1.4 POCO C++

POCO C++ je kolekcia open sorce knižníc, ktoré vytvoril Günter Obiltschnig v roku 2004. Sú to moderné, výkonné knižnice a framoworky pre stavbu internetových aplikácii, ktoré bežia na počítači, servery, mobile, alebo vstavanom systéme. POCO C++ používajú spoločnosti ako HP, Lumiplan, Riverbed, Schneider Electric,... Sú napísané v modernom štandarde ANSI C ++, majú veľmi malú externú závislosť, dobrý mix objektovo orientovaného návrhu s moderným C++. Zahŕňajú funkcie pre sieťové protokoly(HTTP, SMTP, FTP), HTTP server, analyzátor XML, Sax2 a DOM rozhranie a prístup do databázy pomocou SQL.[21]

3.1.5 Webtoolkit

WT je C++ knižnica pre vývoj webových aplikácii. Pre vývojárov ponúka abstrakcie mnohých internetových špecifikácii, vrátane client-server protokolov ako sú HTTP, Ajax, WebSockets a oslobodzuje ich od HTML komunikácie potrebnej cez JavaScript. Narozdiel od komunikácie HTML pomocou JavaScriptu so serverovou aplikáciou, WT umožňuje vytvoriť stavové aplikácie, ktoré sú zároveň vysoko interaktívne ale stále podporujú prosté HTML.

Nevýhoda je, že jazyk HTML sa implementuje priamo do WT aplikácie ktorá beží na servery bez použitia JavaScript, čo na jednu stranu uľahčuje prácu, ale na druhú stranu okrem bezpečnostných rizík vytvára aj mnohé omedzenia nakoľko WT je limitujúce oproti použitia kombinácie Javascript s iným C++ frameworkom.[22]

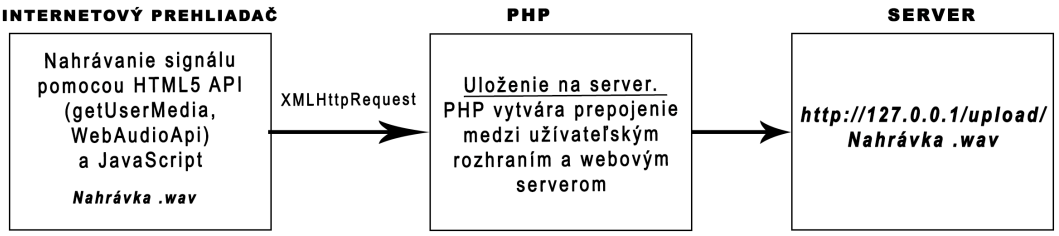
4 NÁVRH WEBOVÉHO ROZHRANIA

Na základe podrobnej analýzy možností internetových aplikácii a možností spracovania rečových signálov cez webové rozhranie, bolo vybraté najvhodnejšie riešenie na tvorbu zadaného webového rozhrania. Nakoľko je potrebné zaznamenaný signál uložiť na webový server, odkiaľ bude možné ho ďalej pomocou webovej aplikácie spracovať, je potrebné v prvej časti vyriešiť možnosť nahratia záznamu na server.

4.1 Nahratie rečového signálu na server

Nahratie rečového signálu prebieha v internetovom prehliadači pomocou HTML5 rozhraní `WebAudioAppi` a `GetUserMedia`, ktoré umožňujú prístup k mikrofónu a rečový signál navzorkujú do poľa rámcov, ktoré sa ukladajú do vyrovnávacej pamäte. Tie sú potom zabalené do formátu WAV.

WAV súbor, obsahujúci nahrávku rečového signálu, je v ďalšom kroku potrebné poslať na server. Pomocou HTLM5 rozhrania XMLHttpRequest je preposlaný do PHP skriptu. PHP vytvára pomyselné spojenie medzi užívateľským rozhraním (internetovým prehliadačom) a webovým serverom. Nakoľko PHP beží už na strane servera nie je problém získaný súbor nahrávky uložiť na server. Priebeh nahrávania signálu na server je znázornený na obrázku 4.1.



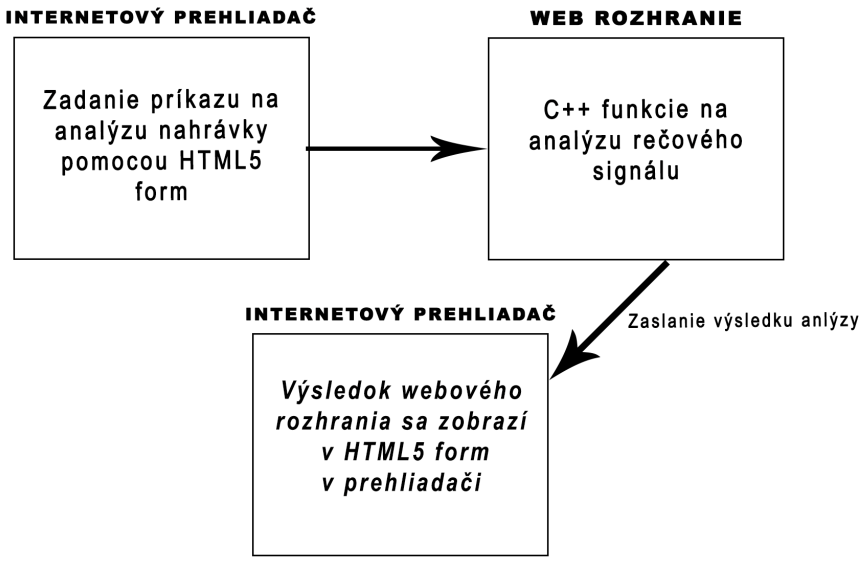
Obr. 4.1: Nahrávanie rečového signálu na server

4.2 Spracovanie rečového signálu

V prípade, že je audio súbor nahraný na server, je možné ho začať analyzovať pomocou C++ internetového rozhrania. V HTML5 nepretržite beží rozhranie WebSocket Api. Umožňuje obnovenie internetovej stránky bez jej nového načítania. Užívateľ vyberie jeden z nahraných súbor a vyberie si jednu z možností analýzy. Požiadavka sa zašle do webového rozhrania C++. Rozhranie načíta nahrávku vo formáte WAV

a pomocou implementovaných funkcií, spracuje nahrávku a zašle výsledok späť do webového rozhrania.

Pre spracovanie rečového signálu na strane servera bolo zvolené webové rozhranie CppCMS. Hlavným dôvodom výberu práve tohoto frameworku je jedna z najlepších dokumentácí a neustála podpora, ktorá je viditeľná hlavne v častom vynovovaní tohoto frameworku. Ďalším dôvodom je kvalitné hodnotenie, možnosť využívania šablón pri tvorbe internetovej stránky, jednoduché a intuitívne používanie, kvalitné zabezpečenie. Priebeh spracovania rečového signálu pomocou webového rozhrania je znázornený na obrázku 4.2.



Obr. 4.2: Spracovanie rečového signálu pomocou webového rozhrania

5 ANALÝZA ZVUKU POMOCOU WEBOVÉHO FRAMEWORKU

Spracovanie zvukových nahrávok je náročné na výpočetnú techniku. Preto ako už bolo spomínané je potrebné zvoliť pre výpočet analýzy nahrávky na strane serveru framework CppCMS, je implementovaný v jazyku C++. Pre prepojenie jazyka C++ s jazykom HTML sa využívajú v CppCMS šablóny. Tie umožňujú vykreslenie webovej stránky v statickej podobe teda pomocou HTML kódu. Aby bolo možné pracovať so šablónami, je treba rozdeliť zdrojové kódy do troch častí:

1. Obsah: obsahuje objekty, ktoré sa majú dynamicky vykresľovať.
2. Samotná aplikácia v C++ kóde, ktorá vytvára dynamický obsah
3. HTML šablóna, ktorá vytvorený obsah v aplikácii vkladá do statickej stránky.

Obsah sa nachádza v súbore `content.h`, v ktorom je hneď na začiatku potrebné definovať hlavičku `<cppcms / view.h>`, ktorá nám poskytne všetky definície potrebné pre triedy. Každý objekt, ktorý sa má dynamicky vykresľovať by mal byť odvodený od triedy `cppcms :: base_content` a mal by ukladať všetky relevantné údaje. Ukážka definície štruktúry s názvom `message`, ktorý obsahuje jeden dynamicky vykresľovaný objekt `string text` by vyzerala takto:

```
namespace content  {
    struct message : public cppcms::base_content {
        std::string text;
    };
}
```

Controller obsahujúci priamo C++ kód musí definovať `#include "content.h"`, aby mohla byť nadviazaná komunikácia. Samotný kód môže vyzerať napríklad takto:

```
virtual  void  main (std :: string  / * url * / )
{
    content::message c;
    c.text=">>>Hello<<<";
    render("message",c);
}
```

V ukážke je vytvorený objekt `c` typu `content::message` a pridelená hodnota typu `c.text`, ktorá bude vložená do HTML kódu v šablóne. Nakoniec je potrebné zavolať funkciu `render("message",c)`, po ktorej zavolaní sa vykreslí automaticky výstup funkcie do šablóny.

Šablóna vykresľujúca HTML kód na začiatku musí zahrnúť obsah pomocou príkazu `<% C ++ #include "content.h"%>`. Šablóna je rozdelená do `skin`. Každá

`skin` je umiestnená v mennom priestore s rovnakým názvom. `Skin` sa skladá z viacerých `view`. Tieto `view` sú reprezentované triedami, každá z nich je zodpovedná za vykreslenie určitého obsahu objektu. Ukážka jednoduchej šablóny, ktorá bude pozostávať zo `skin` s názvom `my_skin`, ďalej definuje `view` s názvom `message` z vopred definovaného objektu `content::message`, potom je potrebné treba definovať funkciu `<% template render() %>`, ktorá skutočne vykreslí HTML kód v ktorom priamo vložíme pomocou príkazu `<%= text %>` dynamický objekt `text` z triedy `content::message`.[23]

```
<% c++ #include "content.h" %>
<% skin my_skin %>
<% view message uses content::message %>
<% template render() %>
<html>
  <body>
    <h1><%= text %> World!</h1>
  </body>
</html>
<% end template %>
<% end view %>
<% end skin %>
```

Zostavenie a zkompilovanie súborov pre HTML stránku prebieha vo viacerých krokoch. Najprv je potrebné šablóny zostaviť do C++ kódu, čo zabezpečí osobitý kompilátor pomocou príkazu `cppcms_tmpl_cc my_skin.tmpl -o my_skin.cpp` . Vytvorený súbor `my_skin.cpp` obsahuje všetky kódy potrebné pre vykresľovanie a registráciu týchto `skin` v systéme CppCMS. Ďalej je potrebné vytvoriť aplikáciu príkazom `g ++ hello.cpp my_skin.cpp -o hello -lcppcms -lbooster`. A nakoniec len jednoducho spustiť `./hello -c config.js`. Súbor `config.js` je samozrejme potrebné vytvoriť. Používa formát JSON. Je potrebné definovať port na ktorom bude služba načúvať a protokol HTTP. Je možné tiež definovať `script_names`, ktorý definuje cestu ku službe. Ukážka súboru `config.js` môže vyzerať takto:

```
{   "service" : {
          "api" : "http",
          "port" : 8080  },
    "http" : {
          "script_names" : [ "/hello" ]  }
}
```

5.1 Formuláre v CppCMS

Ako už bolo spomenuté, CppCMS dokáže veľmi jednoducho a intuitívne prepájať pomocou šablón jazyk HTML s jazykom C++. Ďalšou výhodou tohoto frameforku je práca s HTML formulármi. CppCMS umožňuje jednoduchú reakciu na vytvorený formulár, či už v podobe spustenia nejakých overovacích funkcií vo formulári (vek musí byť vyšší ako 18 rokov), alebo možnosť spustenia výpočetných funkcií. Napríklad užívateľ vo formulári zadá názov nahrávky, zadá akciu aká sa prevedie s touto nahrávkou (zmaže sa súbor, súbor sa bude analyzovať), pri analýze vyberie typ merania a po stlačení tlačítka sa vytvorí statická stránka s výsledkom merania (analýzy) vybranej nahrávky pomocou formulára.

Pri vytvorení šablóny je opäť potrebné editovať súbor s obsahom, `content.h`, do ktorého je potrebné vložiť hlavičky:

```
#include <cppcms/view.h>
#include <cppcms/form.h>
```

Ďalej je potrebné vytvoriť triedu pre formulár, ktorá je odvodená z `cppcms::form`. V triede budú definované ovládacie prvky formulára a vytvorený konštruktor triedy `info_form()`. V konštruktore sú zadefinované názvy ovládacích prvkov, ak sa napríklad jedná o multiple choice ovládací prvok, sú tu zadefinované aj hodnoty. Ukážka vytvorenia formulára pomocou CppCMS vyzerá napríklad takto:

```
namespace content  {
struct info_form : public cppcms::form {
    cppcms::widgets::text meno;
    cppcms::widgets::radio pohlavie;
    cppcms::widgets::submit submit;
    info_form()
    {
        meno.message("Vaše meno");
        pohlavie.message("pohlavie");
        submit.value("Send");
        add(meno);
        add(pohlavie);
        add(submit);
        pohlavie.add("Muž","muž"); pohlavie.add("Žena","žena");
        meno.non_empty();
    }
```

V ukážke sú vytvorené tri ovládacie prvky. Textové pole s názvom meno, kde v konštruktore je možné vidieť ako je textovému poľu priradený popis `"Vaše meno"` a ešte

jedna akcia `meno.non_empty()`, ktorá pri odoslaní formulára s prázdnym textovým poľom meno vypíše chybu. Ďalší ovládací prvok je takzvaný radio widget, kde je možné vybrať práve jednu hodnotu z viacerých možných. Taktiež je možné vidieť v konštruktore priradenie popisu `pohlavie` a pridanie hodnôt do zoznamu `pohlavie.add("Muž","muž"); pohlavie.add("Žena","žena");`. Tretí ovládací prvok je samozrejme jednoduchý button, ktorý má pridelené meno `Send`. V poslednom rade treba v konštruktore zavolať funkciu add, ktorá pridá tieto ovládacie prvky po jednom do formulára v statickej stránke HTML.

Šablóna s použitím formulára môže v tomto prípade vyzerať približne takto:

```
<html> <body>
    <% if not empty name %>
        <h1>Ahoj <%= meno %></h1>
        <p>Ty si <%= pohlavie %></p>
    <% else %>
        <h1>Vložte data do formulára</h1>
    <% end %>
        <form method="post" action="" >
    <% form as_p info %>
        </form>
</body> <html>
```

Prvé dva riadky HTML kódu sa vypíšu len v prípade, ak ovládací prvok meno nie je prázdny. V prvom riadku sa zobrazí pozdrav s Vašim menom, v druhom bude vypísané vaše pohlavie. V druhej časti kódu je vloženie formulára do šablóny.

Práca s formulármi v C++ je taktiež veľmi intuitívna. Je potrebné len načítať dáta z formulára v prípade ich zaslania. To sa overuje jednoduchou podmienkou `if(request().request_method()=="POST")`. Ak došlo ku zaslaniu ďalej je potrebné dáta načítať a overiť ich načítanie pomocou `c.info.load(context());` `if(c.info.validate())` kódu. Ak je formulár platný a načítanie prebehne úspešne je možne s dátami robiť hocičo. Pre ukážku je napríklad možné dáta poslať rovno na výstup do vopred vytvorenej šablóny.

```
c.meno=c.info.meno.value();
c.pohlavie=c.info.pohlavie.selected_id();
```

Nakoniec je potrebné formulár vyčistiť aby mohol užívateľ zadať nové dáta. To sa robí príkazom `c.info.clear()`[24].

6 NAHRÁVANIE A ANALÝZA ZVUKU CEZ WEBOVÚ APLIKÁCIU

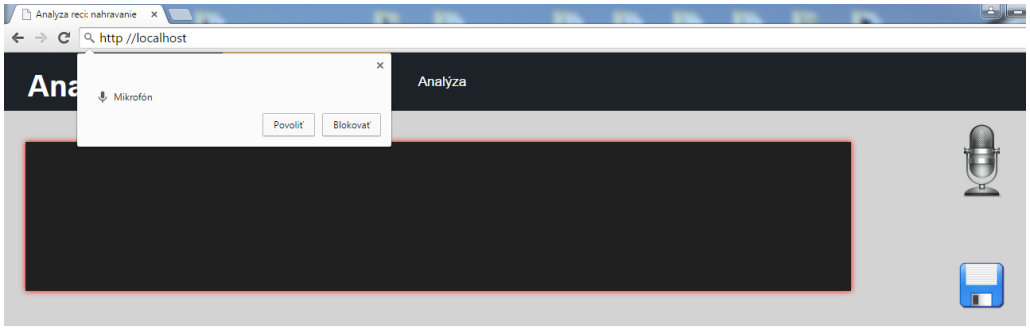
Webová aplikácia sa dá rozdeliť na dve základné časti.

1. Nahrávanie zvuku
2. Spracovanie a analýza zvuku

Sekcia nahrávanie zvuku umožňuje užívateľovi pomocou webového rozhrania zaznamenať zvuk, ktorý ďalej môže uložiť na server. Druhá časť, spracovanie a analýza zvuku, umožňuje užívateľovi vybrať zo zoznamu nahrávok na serveri zaznamenaný zvuk, ktorý môže podrobiť rôznym analýzam.

6.1 Nahrávanie zvuku

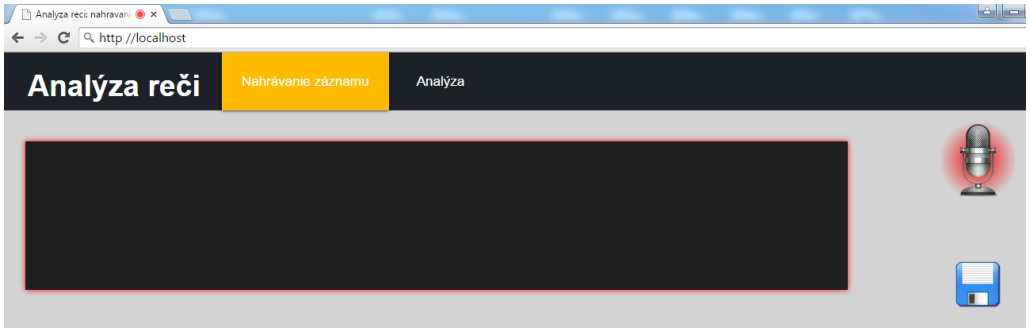
Nahrávanie prebieha v jednoduchej HTML aplikácii, kde sa pomocou HTML5 audio rozhraní a Javascriptu zaznamená zvuk, ktorý sa pomocou jednoduchého PHP kódu uloží do zložky na server. Rozhranie na zaznamenávanie zvuku je veľmi jednoduché. Pri prvom načítaní stránky si každý prehliadač vyžiada povolenie prístupu k mikrofónu ako je možné vidieť na obrázku 6.1.



Obr. 6.1: povolenie prístupu k mikrofónu v prehliadači Chrome

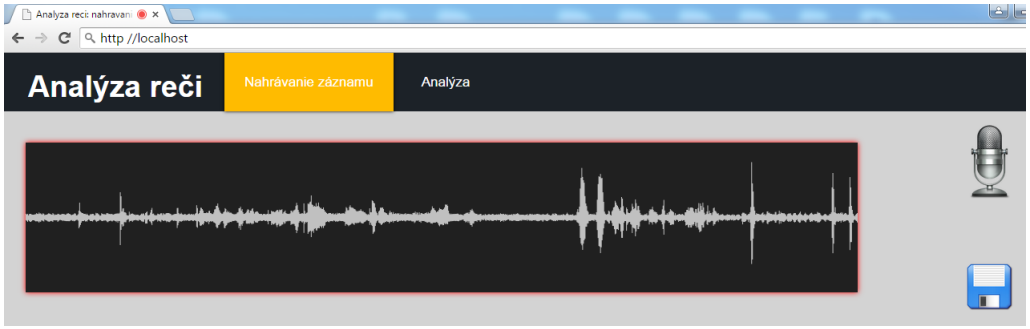
V hornej časti stránky sa načíta horizontálne menu, kde je možné prepínať medzi nahrávaním zvuku a analýzou zvuku. V hlavnej Časti sa nachádzajú tri objekty. Prvý slúži na vykreslenie nahraného zvuku v čase. Ďalej sa na stránke nachádzajú dve tlačítka, jedno slúži na zaznamenávanie zvuku, druhé na zaslanie nahrávky na server. Tieto aktívne obrázky sú prepojene s `JavaScriptovou` funkciu, ktorá v prípade stlačenia mikrofónu, vyvolá funkciu zaznamenávania zvuku z mikrofónu do vyrovnávacej pamäte. Druhá možnosť je stlačenie tlačítka disketa, po ktorej sa zavolá funkcia umožňujúca uloženie zaznamenanej nahrávky z vyrovnávacej pamäti na server. V prípade povolenia prístupu k mikrofónu a kliknutia na mikrofón, sa

teda spustí nahrávanie zvuku. Je vyznačené červeným podsvietením mikrofónu, čo je patrné z obrázka 6.2.



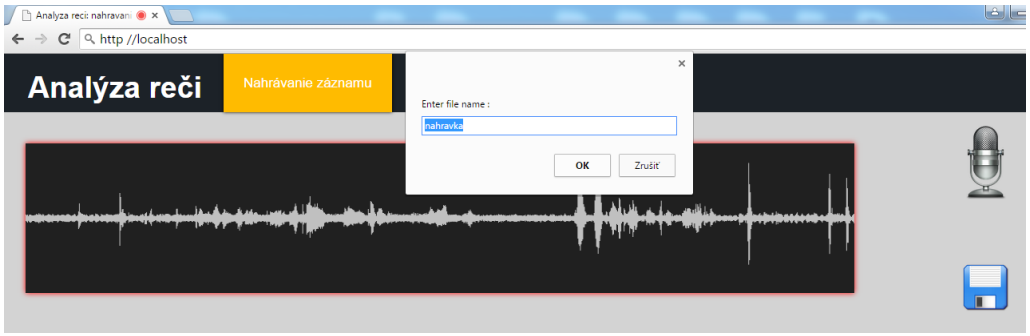
Obr. 6.2: zaznamenávanie zvuku

Prerušenie zaznamenávania zvuku je možne iniciovať opätovným kliknutím na mikrofón okolo ktorého zmizne červené podsvietenie. Je teda patrné že nahrávanie je zastavené. Hneď po ukončení nahrávania sa do tmavého okna načíta graf, v ktorom je možné pozorovať vykreslenie priebehu nahratého zvuku. Priebeh ukážkovej nahrávky je znázornený na obrázku 6.3.



Obr. 6.3: ukončenie záznamu zvuku, vykreslenie priebehu nahrávky

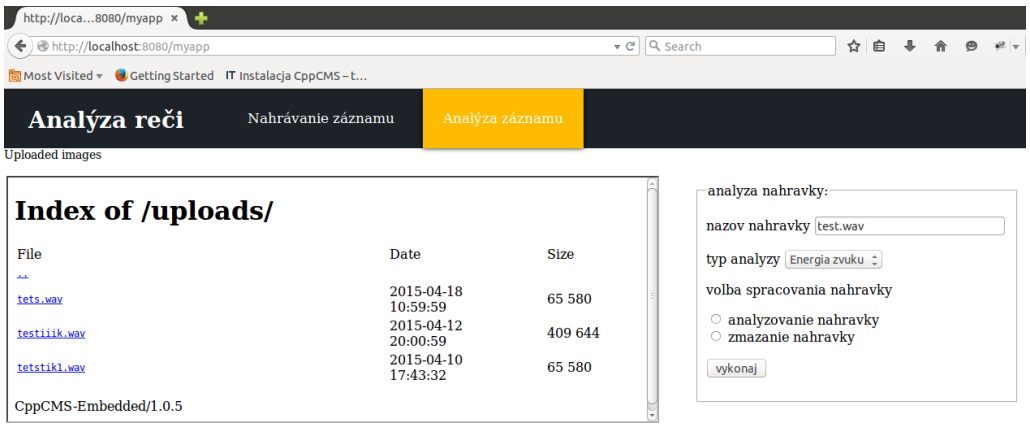
Pokiaľ z akéhokoľvek dôvodu je potrebné nahrávku zaznamenať znova, nie je problém stisnúť opäť mikrofón a zvuk sa začne prepisovať vo vyrovnávacej pamäti. Pre uloženie nahrávky na server slúži druhá ikona v tvare diskety. Po kliknutí práve na toto tlačítko sa spustí `JavaScript` funkcia, ktorá vyvolá vyskočenie dialógového okna. Užívateľ bude vyzvaný ku vloženiu názvu nahrávky. Názov sa vkladá bez prípony `.wav`. Po zadaní názvu súboru a potvrdení sa nahrávka okamžite uloží na server. Tento postup je patrný z obrázka 6.4.



Obr. 6.4: uloženie zaznamenanej nahrávky na server

6.2 Analýza zvuku

Analýza je druhou časťou webovej aplikácie. Táto úloha už je naprogramovaná pomocou frameworku CppCMS v jazyku C++ a preto musí bežať na servery. Aby s ním mohla komunikovať je potrebné prideliť aplikácií port. V tomto prípade je to port 8080, pomocou ktorého bude prebiehať http komunikácia. Táto web stránka je generovaná pomocou CppCMS šablón, ktorých funkčnosť bola vysvetlená v predošlej kapitole. Ďalej je tu taktiež použitý spomínaný formulár, pomocou ktorého užívateľ dokáže zadávať príkazy na spracovanie, analyzovanie alebo vymazanie nahrávky zo servera.



Obr. 6.5: webové rozhranie pre analýzu zvuku

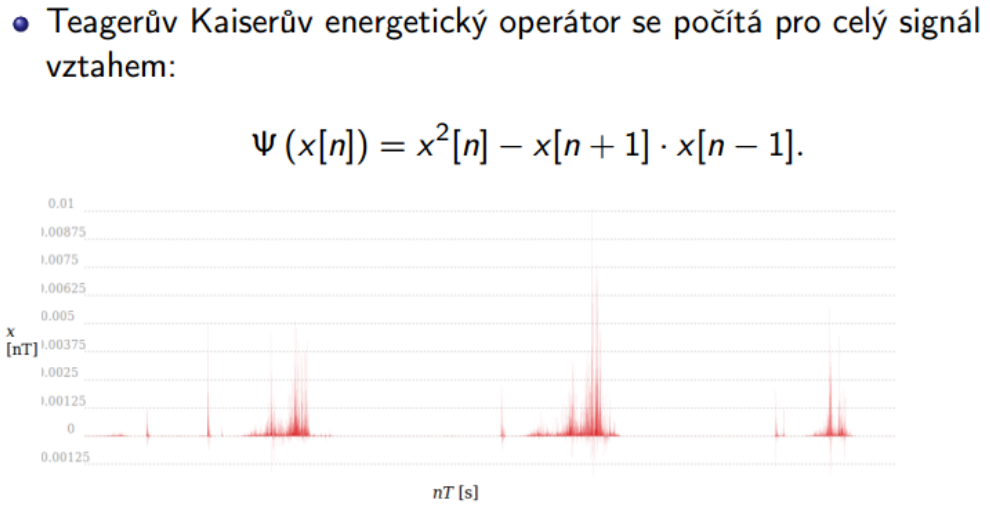
Na obrázku 6.5 je možné vidieť vykreslenú HTML stránku slúžiacu na spracovanie a analýzu nahrávok. V hornej časti je umiestnené opäť navigačné menu, pomocou ktorého sa dá preklikávať medzi nahrávaním a analýzou nahrávky. Hlavnú Časť stránky je možné rozdeliť na dve časti. Na ľavej strane je vidieť okno, ktoré obsahuje zoznam všetkých nahrávok umiestnených na servery. V okne je zobrazený názov nahrávky, čas nahratia a veľkosť audio súboru. Na pravej strane webovej

stránky sa nachádza jednoduchý HTML formulár, ktorý je prepojený s funkciami C++ pomocou frameworku CppCMS. Slúži pre vybratie udalosti, ktorá sa bude realizovať s vybranou nahrávkou. Nahrávku je teda možne analyzovať alebo zmazať. Pri analýze nahrávky je ďalej potrebné vybrať typ analýzy. Pre spustenie analýzy stačí jednoducho zadať názov nahrávky, typ analýzy a zaškrtnúť políčko `analyzovanie zvuku`. Po kliknutí na tlačítko `vykonaj` sa obnoví stránka a v dolnej časti sa objaví výsledok analýzy.

6.3 Výsledky analýzy

Pri žiadosti o spracovanie nahrávky sa spustí C++ aplikácia, ktorá načíta vybratú zvukovú nahrávku vo formáte WAV. Túto nahrávku vloží do poľa s ktorým sa neskor pracuje pri konkrétnej analýze. Ďalej sa pomocou formulára vyberie typ analýzy. Podľa vybratej možnosti sa spustí funkcia C++ ktorá vypočíta hodnoty výslednej analýzy.

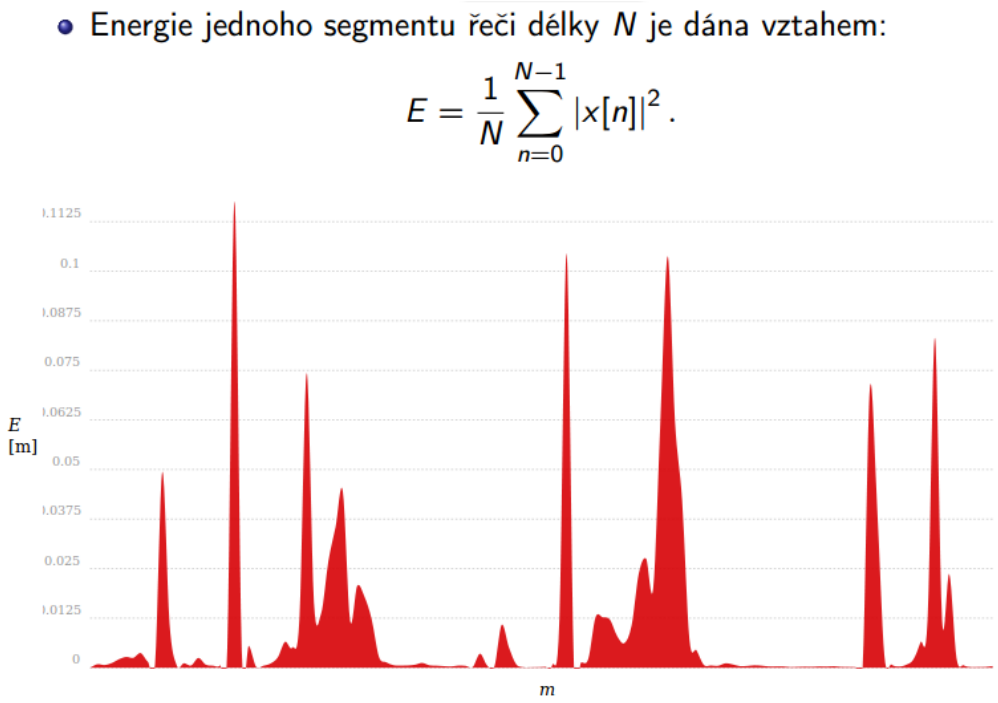
V testovacom režime boli do aplikácie implementované dve analýzy. Analýza krátkodobej energie zvuku a analýza pomocou Teagreového Kaiserovho energetického operátora. Výtup z analýzy pomocou spomínaného Teagreového Kaiserovho energetického operátora je možné vidieť na obrázku 6.6. Na obrázku je znázornený aj vzorec pre výpočet jednotlivých hodnôt. Úplne ako prve sa vo výpise výsledku analýzy zobrazí názov analyzovanej nahrávky.



Obr. 6.6: výstup analýzy pomocou Teagreového Kaiserovho energetického operátora

Druhou možnosťou analýzy nahrávky, ktorá je implementovaná v aplikácii je krátkodobá energia. Pri tejto analýze je ešte potrebné zadať dĺžku segmentu v milisekundách. Výstup z tejto analýzy je taktiež ako v predošlom prípade graf, ktorý

je možno vidieť na obrázku 6.7.



Obr. 6.7: výstup analýzy pomocou krátkodobej energie

Oba naimplementované typy analýzy používajú pre vykreslenie výsledku zmieňovaný graf. Pre používanie grafov vo webovej stránke bola použitá knižnica s názvom `CHARTIST.JS`. Hodnoty do grafu sa zadávajú pomocou premenných, kroté sú typu string a majú názov `c.xOs` a `c.yOs`.

6.3.1 Vloženie nového typu analýzy

Pre pridanie novej analýzy do aplikácie je potrebné vytvoriť jej názov vo formulári. Konfiguračné údaje formulára sa nachádzajú v súbore `content.h`. Ďalej je už len potrebné vhodne naimplementovať funkciu novej analýzy do súboru obsahujúceho kód v jazyku C++ s názvom `myapp.cpp`. Výsledky sa vykresľujú pomocou šablóny do HTML stránky cez premennú `c.vysledok`. Konkrétna ukážka vloženia výsledného grafu môže vyzerať takto:

```
c.vysledok += "<div class=\"ct-chart ct-perfect-fourth\"></div>\n";
```

7 SPUSTENIE APLIKÁCIE NA LOKÁLNOM WEBOVOM SERVERY

V prvom rade je potrebné na lokálny počítač nainštalovať webový server. Aplikácia bola testovaná v operačnom systéme `Ubuntu`, kde bol nahraný open-source webový sever `lighttpd`. Ďalej je potrebné nainštalovať framework `CppCMS`. V dobe testovania bola použitá verzia `CppCMS 1.0.5`.

V aplikácií sú použité šablóny a to konkrétne hlavná šablóna `Master` a šablóna `Intro`. V hlavnej šablóne je zadefinovaná hlavička stránky, čo je navigačné menu, ktoré slúži na preklikávanie medzi nahrávaním a analýzou nahrávky. Ďalej sa do tejto šablóny načítava už konkrétny obsah šablóny `Intro`, pomocou ktorej sa vykreslí na stránke obsah nahrávok a formulár na analýzu audio súborov. Tieto šablóny je potrebné skompilovať. Pomocou príkazu uvedeného nižšie je možné obe šablóny skompilovať do spustiteľného súboru s názvom `my_skin.cpp`.

```
cppcms_tmpl_cc master.tmpl intro.tmpl -o my_skin.cpp
```

Ďalej je potrebné prepojiť súbor so šablónami s C++ súborom, ktorý obsahuje priamo funkcie implementované pre chod aplikácie. To sa prevedie pomocou nasledujúceho príkazu:

```
g++ myapp.cpp my_skin.cpp -o hello -lcppcms -lbooster -lsndfile
```

Parametere `-lcppcms` a `-lbooster` sú parametrami frameworku CppCMS, zatiaľčo parameter `-lsndfile` slúži ku preloženiu funkcií z knižnice `libsndfile`, ktoré spracovávajú audio nahrávku. Konkrétne umožňujú načítanie audio súbora vo formáte `WAV` a jeho uloženie do poľa.

Pre spustenie aplikácie, ktorá bude bežať ako démon na lokálnom servery slúži nasledujúci príkaz.

```
./hello -c config.js
```

Po zadaní príkazu sa spustí na pozadí aplikácia a po načítaní cieľovej lokálnej adresy sa zobrazí webová stránka, skonštruovaná zo šablón skompilovaných v predchádzajúcej časti.

Súbor `config.js` obsahuje dôležité údaje pre spustenie aplikácie a to v konkrétnom prípade napríklad službu, pomocou ktorej bude prebiehať komunikácia. V tomto prípade bude aplikácia počúvať na porte 8080 pomocou protokolu `http`. Ďalej je zadefinované takzvané `script_name`, ktoré definuje cestu ku aplikácii. Pri použití `/myapp` bude cesta ku aplikácii vyzerať takto:

```
http://localhost:8080/myapp
```

Ďalšia dôležitá vec, ktorá musí byť v tomto prípade zadefinovaná v konfiguračnom súbore je cesta ku nahrávkam uloženým na servery. Takzvaný `file_server` je predvolene vypnutý. V prvom rade je ho potrebné povoliť a ďalej je potrebné zadať konkrétnu cestu ku adresáru, ktorá sa zadáva pomocou `document_root`.Ukážka súboru `config.js` je zobrazená nižšie.

```
{
"service" : {
"api" : "http",
"port" : 8080
},
"http" : {
"script" : "/myapp"
},
  "file_server" : {
  "enable": true,
  "listing" : true,
  "document_root" : "/home/markii/bc"
}
}
```

7.1 Dôležité údaje pre spustenie aplikácie na inom servery

Pri spustení aplikácie na inom servery je potrebné nakonfigurovať niekoľko adries ku súborom. V provom rade je potrebné zadať adresu webovej stránky, ktorá umožňuje nahrávanie zvuku, do šablóny `master.tmpl`. Kde v hlavičke je treba zmeniť adresu `<li><a href="http://localhost/index.html»Nahrávanie záznamu</a></li>`.

Ďalšou adresu, ktorú treba nastaviť, je cesta ku súborom ktoré sa ukladajú na servery. V súbore `myapp.cpp` je treba zmeniť cestu uloženú v premennej `Upload`. Ukážková cesta vyzerá takto:

```
const char* Upload = "/home/markii/bc/uploads/";
```

Poslednou adresou ktorú je potrebné zmeniť, je tak ako pri časti analýza zvuku, aj pri časti nahrávanie zvuku v hlavičke `HTML` súboru `index.html` adresu, ktorá pomocou horizontálneho menu dokáže odkazovať na analýzu zvuku. Ukážka súboru index.html, kde je potrebné zmeniť adresu vyzerá takto:

```
<li><a href="http://localhost:8080/myapp">Analýza</a></li>
```

8 ZÁVER

Cieľom bakalárskej práce bolo nájsť ideálnu možnosť na spracovanie zvukových signálov pomocou webového rozhrania. V prvej časti práce bol teoreticky rozobratý postup nahrávania zvuku cez internetovú stránku, ďalej bolo riešené uloženie nahrávky na server. Potom boli v práci rozobraté možnosti analýzy zvuku cez webové rozhranie. Na konci práce bolo navrhnuté a implementované komplexné riešenie.

V prvej časti sú podrobnejšie rozobraté možnosti nahrávania audio signálu a zaznamenávania ho z mikrofónu pomocou internetového prehliadača. Nakoľko vývoj ide rýchlo dopredu, zásuvné moduly už nie sú jedinou možnosťou ako zaznamenávať zvuk. Preto v práci bola zvolená možnosť nahrávania audio signálu pomocou rozhraní HTML5. Sú jednoduché a nenútia užívateľov inštalovať žiadne externé aplikácie. Po nahratí audio signálu je potrebné nahrávku zaslať na server, kde sa uloží.

V druhej časti práce je riešený problém zasielania nahrávky na server. Zaisťuje to rozhranie **XMLHttpRequest**, ktoré spolu z jednoduchou funkciou v PHP ukladá nahrávku na server.

V ďalšej časti práce sú rozobraté možnosti webových rozhraní. Nakoľko analýza zvuku je náročná na výpočetnú techniku, pre túto prácu bolo zvolené webové rozhranie postavené na programovacom jazyku C++. V tejto časti boli rozobraté dostupné frameworky a vybratá najlepšia možnosť. Pre tento typ úlohy bol zvolený framevork **CppCMS**. Dôvod voľby bol hlavne kvôli jednoduchosti, vynikajúcej dokumentácii a nenáročnosti na výkon, čo je v tomto prípade veľmi dôležité.

V nasledujúcej kapitole je možné vidieť navrhnutie celkovej kostry webového rozhrania. Kde je rozobraté stručné riešenie teoreticky pomocou blokových schém, ktoré rozdeľujú aplikáciu na konkrétne bloky.

V ďalšej časti je už spomenutý priamo framework CppCMS a jeho funkcie. Sú to hlavne možnosti použitia šablón a prepojenia HTML form s jazykom C++, ktoré sú využité priamo v programe.

V predposlednej časti je dopodrobna naznačená funkčnosť programu. Od nahrávania nahrávky cez posielanie na server a samozrejme aj jej spracovanie. Ďalej sú vykreslené ukážky výstupu analýzy v podobe grafov pre výpočty energie zvuku.

V poslednej časti bakalárskej práce je v stručnosti naznačená možnosť kompilácie a spustenia programu pomocou lokálneho servera. Ďalej je v tejto časti možné nájsť dôležité údaje pre spustenie aplikácie na inom servery a teda konfiguračné údaje.

V ďalšej časti práce by bolo možné ešte vylepšiť aplikáciu o mnoho ďalších analýz, ktoré sa nemusia vykresľovať len pomocou grafov. K vykresleniu výsledkov je možné použiť tabuľky, ale aj všetko ostatné, čo HTML a jazyk JavaScript dovolí. Ďalej by sa dalo popracovať ešte aj na zabezpečení aplikácie, nakoľko prebieha komunikácia so serverom, bolo by potrebné vytvoriť zabezpečenie voči neželanému vniknutiu.

LITERATÚRA

[1] Can I Use. *caniuse.com* [cit. 2014-12-09]. Online: `http://caniuse.com/`

[2] LUBBERS, Peter, Brian ALBERS a Frank SALIM. *HTML5: programujeme moderní webové aplikace* Vyd. 1. Brno: Computer Press, 2011, 304 s. ISBN 978-80-251-3539-6.

[3] Mozilla developer. *CSS3* [cit. 2014-12-09]. Online: `https://developer.mozilla.org/en-US/docs/Web/CSS/CSS3`

[4] ŠKULTÉTY, R. *Programujeme internetové aplikace JavaScript ISBN 80-251-0144-4* 2001, posledná aktualizácia 11. 11. 2004 [cit. 17. 2. 2005]

[5] Whatls. TECHTARGET. *Plug-in* [cit. 2014-12-09]. Online: `http://whatis.techtarget.com/definition/plug-in`

[6] TECHTARGET. *Adobe flash player* [cit. 2014-12-09]. Online: `http://searchcio.techtarget.com/definition/Adobe-Flash-Player`

[7] W3C. *Geolocation API Specification* [cit. 2014-12-09]. Online: `http://dev.w3.org/geo/api/spec-source`

[8] W3C. *GetUserMedia API* [cit. 2014-12-09]. Online: `http://dev.w3.org/2011/webrtc/editor/getusermedia.html`

[9] Can I Use. *getUserMedia* [cit. 2014-12-09]. Online: `http://caniuse.com/#search=getUserMedia`

[10] Mozilla develooper. *NavigatorUserMedia* [cit. 2014-12-09]. Online: `https://developer.mozilla.org/en-US/docs/NavigatorUserMedia.getUserMedia`

[11] Can I Use. *WebAudio API* [cit. 2014-12-09]. Online: `http://caniuse.com/#feat=audio-api`

[12] SMUS, Boris. *Web Audio API* 2011, poslední aktualizace 19. 11. 2014 [cit. 2014-11-12]. Online: `http://www.html5rocks.com/en/tutorials/webaudio/intro/`

[13] Mozilla developer. *MediaStreamAudioSourceNode* [cit. 2014-11-12]. Online: `https://developer.mozilla.org/en-US/docs/Web/API/MediaStreamAudioSourceNode`

[14] Mozilla developer. *ScriptProcessorNode* [cit. 2014-11-12]. Online: `https://developer.mozilla.org/en-US/docs/Web/API/ScriptProcessorNode`

[15] W3C. *XMLHttpRequest* [cit. 2014-11-20]. Online: http://www.w3.org/TR/XMLHttpRequest/

[16] W3C. *Websockets* [cit. 2014-11-20]. Online: http://www.w3.org/TR/2011/WD-websockets-20110419/

[17] Managementmania. *Webová aplikácia* 2013, poslední aktualizace 7. 6. 2013 [cit. 2014-11-20]. Online: https://managementmania.com/sk/webova-aplikacia-web-application

[18] GFDL *CppCMS* [cit. 2014-11-20]. Online: http://cppcms.com/

[19] Microsoft *C++ REST SDK* [cit. 2014-11-20]. Online: http://casablanca.codeplex.com/

[20] David Beazley, Sotiria Lampoudi, Gonzalo Diethelm *SWILL* [cit. 2014-11-22]. Online: http://swill.sourceforge.net/

[21] Applied Informatics Software Engineering GmbH *POCO* [cit. 2014-11-22]. Online: http://pocoproject.org/

[22] *C++ Web Toolkit* [cit. 2014-11-22]. Online: http://www.webtoolkit.eu/wt/

[23] *CppCMS - Using Templates* [cit. 2015-04-12]. Online: http://cppcms.com/wikipp/en/page/cppcms_1x_tut_hello_templates

[24] *CppCMS - Working With Forms* [cit. 2015-04-12]. Online: http://cppcms.com/wikipp/en/page/cppcms_1x_forms

ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

CppCMS	Framework na prepojenie jayzka HTML s programovacím jayzkom C++
CSS	Kaskádové štyly – Cascading Style Sheet
C++	Programovací jazyk
HTML	HyperText Markup Language
HTTP	Hypertextový prenosový protokol
JavaScript	Skriptovací programovací jazyk
PHP	skriptovací jazyk – Hypertext Preprocessor
WAV	Waveform Audio File Format
WHATWG	Web Hypertext Application Technology Working Group
W3C	Konzorcium produkujúce slobodné štandardy – World Wide Web Consortium

ZOZNAM PRÍLOH

A OBSAH PRILOŽENÉHO CD

Na priloženom CD sa nachádzajú súbory pre spustenie a konfiguráciu aplikácie. Na CD sú dva adresáre pod názvami `server` a `analýza`. V adresári `analýza` sa nachádzajú súbory potrebné pre preklad a spustenie frameworku CppCMS. Nachádzajú sa tam súbory s hlavnou (`master.tmpl`) a vedľajšou (`intro.tmpl`) šablónou, súbor s implementovanými funkciami v C++ (`myapp.cpp`), súbor v jazyku C++ so skompilovanými šablónami (`ma_skin.cpp`), a konfiguračné súbory `content.h` a `config.js`.

V adresári `server` sa nachádzajú obrázky použité v aplikácii, súbor `index.html`, ktorý vykresľuje stránku pre nahrávanie zvuku a súbor `save.php`, ktorý obsahuje funkciu na zasielanie nahrávok na server. Ďalej obsahuje tri adresáre, `uploads`, `js` a `bower_components`. Adresár `bower_components` obsahuje súbory z knižnice `CHARTIST.JS`, ktoré slúžia pre vykresľovanie grafov. Adresár `uploads` obsahuje nahrávky, ktoré sú uložené pomocou zmieňovaného PHP programu práve do tohoto adresára. A posledný adresár má názov `js`. Ten obsahuje `Javascriptové` súbory pre nahrávanie a zaznamenávanie zvuku z mikrofónu.

Aplikácia bola testovaná v prehliadači firefox, v operačnom systéme UBUNTU, ktorý bežal vo VirtualBoxe. Ako lokálny webový server bol použitý server `lighttpd`, pre prácu s grafmi bola použitá knižnica `CHARTIST.JS`.