

Návrh a implementace software určeného k rozeznávání intonace a rytmiky při zpěvu

Design and implementation of software for detection of intonation
and rhythm while singing

Bc. Petr Majtán

Diplomová práce
2011



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2010/2011

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Petr MAJTÁN**
Osobní číslo: **A09482**
Studijní program: **N 3902 Inženýrská informatika**
Studijní obor: **Informační technologie**

Téma práce: **Návrh a implementace software určeného
k rozeznávání intonace a rytmiky při zpěvu**

Zásady pro vypracování:

1. Provedte literární rešerši na téma programy pro výuky intonace a rytmiky.
2. Popište, jak lze v digitální podobě popsat zpěv a jaké jsou parametry určující správnou rytmiku a intonaci.
3. Vytvořte program umožňující záznam zpěvu a jeho následnou analýzu.
4. Navrhněte a realizujte webovou prezentaci s možností personifikace a možností uploadů nahrávek a notových základů.
5. Naplňte systém příklady prezentujícími funkčnost celého systému.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. NAGEL, Christian, et al. C 2008 : Programujeme profesionálně. David Dirga. 1. vydání. Brno : Computer Press, a.s., 2009. 1898 s. ISBN 978-80-251-2401-7.
2. MCCONNELL, Steve. Dokonalý Kód : Umění programování a techniky tvorby software. Bogdan Kiszka. Brno : Computer Press, a.s., 2006. 894 s. ISBN 80-251-0849-X.
3. PETZOLD, Charles. Mistrovství ve Windows Presentation Foundation. Jakub Mikulaščík, Jiří Fadrný. Brno : Computer Press, a.s., 2008. 928 s. ISBN 978-80-251-2141-2.
4. GUTMANS, Andi; BAKKEN, Saether Stig; RETHANS, Derick. Mistrovství v PHP 5. Ivo Magera; Bogdan Kiszka. 2. vydání. Brno : Computer Press, a.s., 2007. 655 s. ISBN 978-80-251-1519-0.
5. KOFLER, Michael. Mistrovství v MySQL 5 : Kompletní průvodce webového vývoje. Martin Domes; Jan Svoboda, Ondřej Baše, Jaroslav Černý. 1. vydání. Brno : Computer Press, a.s., 2007. 805 s. ISBN 978-80-251-1502-2.
6. WILLIAMS, E. Hugh; LANE, David. Programujeme webové aplikace pomocí PHP a MySQL. Ivo Magera; David Krásenský. 1. vydání. Praha : Computer Press, a.s., 2002. 530 s. ISBN 80-7226-760-4.
7. CROFT, Jeff; LLOYD, Ian; RUBIN, Dan. Mistrovství v CSS : Pokročilé techniky pro webové designéry a vývoje. Martin Domes; Josef Bábík. 1. vydání. Brno : Computer Press, a.s., 2007. 409 s. ISBN 978-80-251-1705-7.

Vedoucí diplomové práce:

Ing. Tomáš Sysala, Ph.D.

Ústav automatizace a řídicí techniky

Datum zadání diplomové práce:

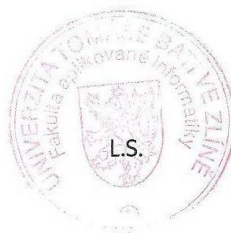
24. února 2011

Termín odevzdání diplomové práce:

18. května 2011

Ve Zlíně dne 24. února 2011

prof. Ing. Vladimír Vašek, CSc.
děkan



doc. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

ABSTRAKT

Diplomová práce si klade za cíl využití znalostí programování, analýzy zvuku a implementovat software určeného zejména k rozeznávání intonace ze zvukových dat, ale také rytmiky testované osoby.

V teoretické části je popsána akustika, hudební názvosloví a technické prostředky nutné k tvorbě aplikace. V praktické části jsou vysvětleny nejdůležitější výňatky zdrojových kódů, struktury tabulek a princip funkcí webové prezentace a v neposlední řadě také testování samotné aplikace na zvukových záznamech.

Klíčová slova: FFT, C#, analýza, rytmika, PHP, třída, noty.

ABSTRACT

The purpose of this diploma thesis is to use the knowledge in a field of programming and sound-analysis and putting it into an application used for recognizing intonation from audio files and also rhythmic of tested person. Music terminology, acoustics and technical instruments used in making of this application are described in theoretical part. Analytical part is about explaining extracts of source-codes, charts structures, methods of the web presentation and testing the sound records as well.

Keywords: FFT, C#, analysis, rhythmic, PHP, class, tones.

Úvodem bych chtěl poděkovat mému vedoucímu práce, panu Ing. Tomáši Sysalovi, Ph.D., za pomoc a rady při tvorbě bakalářské práce a také mým rodičům za velkou podporu po celou dobu studia.

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

OBSAH

ÚVOD	9
I TEORETICKÁ ČÁST	10
1 AKUSTIKA	11
1.1 FYZIKÁLNÍ CHARAKTER ZVUKU	11
1.1.1 Kmitání.....	11
1.1.1.1 Tlumené kmity	13
1.1.1.2 Nucené kmity	14
1.1.1.3 Skládání kmitů	14
1.1.2 Vlnění.....	15
1.2 ZVUKOVÁ ANALÝZA.....	16
1.3 FYZIOLOGICKÁ AKUSTIKA	17
2 HUDEBNÍ TEORIE	19
3 TECHNICKÉ PROSTŘEDKY	21
3.1 ARCHITEKTURA .NET A C#.....	21
3.1.1 Platformní nezávislost.....	21
3.1.2 Výkon.....	21
3.1.3 Jazyk IL	21
3.2 PHP.....	22
3.2.1 SQL injection	22
3.2.1.1 Manipulace s URL	22
3.2.1.2 Pomocí formuláře	23
3.2.2 Vzdálené skriptování (XSS).....	23
3.2.3 MD5 cracking.....	23
3.2.3.1 Bruce-force attack	24
3.2.3.2 Rainbow tables.....	24
3.2.3.3 Dictionary attack	24
3.2.3.4 MD5 online cracking	24
3.3 DATABÁZE	25
3.3.1 Tabulka, datový záznam.....	25
3.4 KASKÁDOVÉ STYLY (CSS).....	25
II PRAKTICKÁ ČÁST	27
4 POPIS PROGRAMU GOLD-VOICE	28
4.1 SPECIFIKACE NÁZVŮ OBJEKTŮ	29
4.2 NOTACE.....	30
4.2.1 Tvorba notového základu	30
4.2.2 Zobrazení dat.....	31
4.2.3 Souborový typ GVO.....	32
4.2.4 Otevírání GVO souborů	33
4.2.5 Ukládání GVO souborů.....	34

4.3	ANALÝZA WAV SOUBORU.....	35
4.3.1	Import WAV souboru.....	35
4.3.2	Rychlá fourierova transformace	36
4.3.3	Zobrazení výsledků	39
4.3.4	Ukládání výsledků.....	39
5	DYNAMICKÁ WEBOVÁ PREZENTACE.....	41
5.1	REGISTRACE.....	41
5.2	PŘIHLAŠOVÁNÍ	43
5.3	PERSONALIZACE UŽIVATELŮ	44
6	TESTOVÁNÍ PROGRAMU.....	45
6.1	TESTOVÁNÍ INTONACE	45
6.1.1	Analýza C-DUR MIDI klavír	46
6.1.2	Analýza C-DUR MIDI viola	47
6.1.3	Srovnání analýz C-DUR MIDI nástrojů.....	48
6.1.4	Analýza C-DUR zpěv.....	49
6.1.5	Srovnání C-DUR MIDI viola a zpěv.....	50
6.2	TESTOVÁNÍ RYTMIKY	51
7	UŽIVATELSKÁ PŘÍRUČKA	52
7.1	OVLÁDÁNÍ PROGRAMU	52
7.1.1	Práce s notovým základem.....	52
7.1.2	Analýza souboru WAV	52
7.1.3	Práce s rytmickými cvičeními	53
	ZÁVĚR	54
	CONCLUSION	55
	SEZNAM POUŽITÉ LITERATURY	57
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK	58
	SEZNAM OBRÁZKŮ	59
	SEZNAM TABULEK.....	60
	SEZNAM PŘÍLOH.....	61

ÚVOD

Diplomová práce je zaměřena na vývoj aplikace určené k rozeznávání intonace a rytmiky při zpěvu.

V dnešní době existuje na trhu několik karaoke aplikací, které zároveň při zpěvu kontrolují intonaci. Existují ve verzích jak pro PC, tak i pro konzolovou stanici Sony Playstation 2. Ovšem žádná z nich nenabízí detailnější pohled na vývoj jednotlivých frekvencí. Tuto funkci nabízí studiové programy, které nejen vykreslí vývoj frekvence do každého vzorku, ale také umožní jejich případné doladění na tíženou frekvenci. Toho se hojně využívá při nahrávání zpěvu na CD. Studiové programy však nejsou levnou záležitostí a nemůže si je dovolit každý.

Tato práce pojednává o návrhu a implementaci software s možností analýzy nejen intonace, ale i rytmiky testované osoby. Proto je zde detailně popsána samotná funkčnost programu z pohledu zdrojových kódů, kde jsou vysvětleny nejdůležitější metody a postupy při vývoji této aplikace. Nejdůležitějšími úryvky zdrojových kódů byly zvoleny úryvky k práci s notací, jejich tvorba, zobrazení, otevírání, ukládání a přehrávání nebo také úryvky kódů při analýze souboru wav. Dále je obsahem práce popis jednotlivých metod dynamické webové stránky, metody registrace a přihlašování. Je zde také uvedeno testování funkčnosti celého programu za pomoci zvukových nahrávek, umístěných na doprovodném CD.

Teoretická část je rozdělena do tří kapitol. První kapitola obsahuje popis vzniku zvukového signálu, jejich analýzu a fyziologickou podstatu. Druhá kapitola krátce popisuje hudební názvosloví a lidské vnímání samotných tónů. V kapitole třetí lze nalézt popis technických prostředků využitých při vývoji programu, jako jsou programovací jazyk C#, PHP, popis databáze MySQL a stylovacího jazyka CSS.

Kapitoly 4-7 obsahují praktickou část této diplomové práce. Ve čtvrté kapitole je popsán návrh notace, práce s ní a analýza zvuku. Pátá kapitola se zabývá implementací dynamické webové stránky s možností personifikace osob. Šestá kapitola se věnuje testování programu a v poslední sedmé kapitole je uživatelský manuál.

I. TEORETICKÁ ČÁST

1 AKUSTIKA

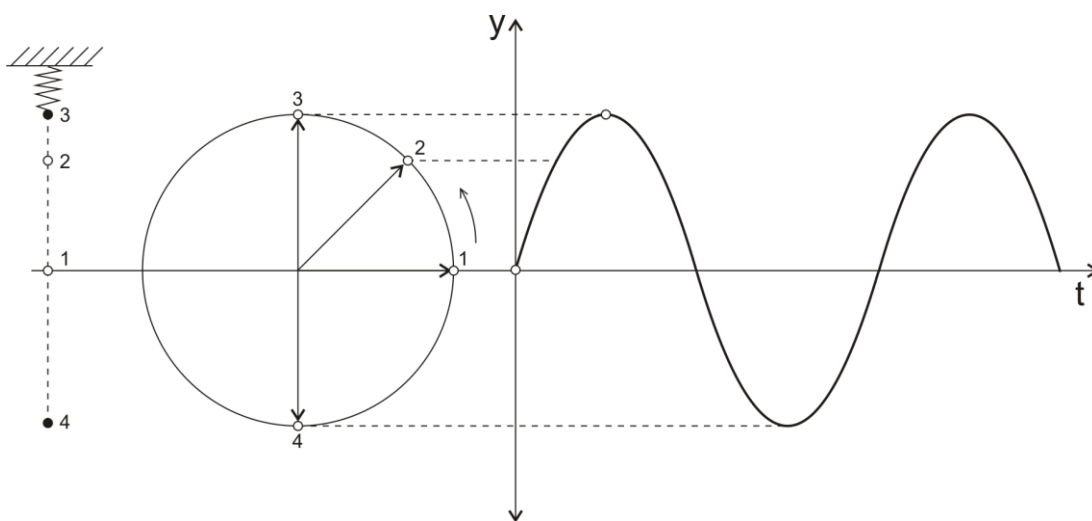
Akustika je široký vědní obor, zabývající se souhrnně zvukem. Ať už jeho vznikem, přenosem v prostoru a v neposlední řadě také vnímáním lidskými smysly. Akustika má celou řadu svých podoborů, ovšem vzhledem k této práci jsou nejdůležitějšími *hudební akustika* a z části také *fyzilogická akustika*. Hudební akustika se zaměřuje na zkoumání fyzikálních základů hudby a fyziologická akustika se zabývá vznikem zvuku v hlasovém orgánu a následném vnímání v uchu [8].

1.1 Fyzikální charakter zvuku

Zvukem se obecně rozumí každý kmitavý pohyb hmoty v pevném, plynném či kapalném skupenství, který je pak schopen vyvolat sluchový vjem. Kmitavý pohyb může mít dva charaktery. Při prvním dochází ke kmitání soustavy hmotných bodů jako celek a jedná se tedy o *kmitání*. Ovšem kmitá-li soustava takovým způsobem, že výchylky jednotlivých bodů jsou různé, jde o druhý případ charakteru kmitání tzv. *vlnění* [8].

1.1.1 Kmitání

Kmitání mnohdy označované také jako oscilace, je fyzikální děj, při kterém se střídavě mění hodnota charakteristické veličiny v závislosti na čase. Může se jednat o pohyb hmotného bodu kolem své klidové polohy, elektrické napětí, odpor, proud či magnetický tok apod. Při pohybu jsou rozlišovány dva základní typy. A to buď po přímce, nebo po křivce [8].



Obr. 1 Odvození kmitavého harmonického pohybu

V případě pohybu po křivce se může jednat o rovnoměrný kruhový pohyb, při kterém se u periodicky opakujících se otáčkách (stálou rychlostí) projevuje jako harmonické kmitání (nerovnoměrný přímočarý pohyb).

Harmonické kmity se charakterizují třemi základními veličinami. A to dobou periody, amplitudou a okamžitou amplitudou [8].

Doba periody

Udává dobu trvání jednoho kmitu (periodického děje) tj. doba, za kterou bod oběhne dráhu kružnice ($0^\circ - 360^\circ$). Symbol veličiny je T a základní jednotkou je 1 sekunda (s).

$$T = \frac{1}{f} \quad (1)$$

,kde veličina f nazývaná *frekvence* je definována jako počet kmitů (period) za jednu sekundu. Jednotkou je 1 hertz (Hz).

Amplituda

Symbol veličiny je velké písmeno A . Udává poloměr kružnice, tj. maximální hodnota bodu od rovnovážné polohy. *Efektivní amplituda* (A_{ef}) je taková amplituda, při níž má kinetická a potenciální energie střední hodnotu. Její hodnota se získá výpočtem dle následujících vztahů. První platí pro kmitavý pohyb sinusového průběhu a druhý platí pro kmitavý pohyb obecného průběhu [8].

$$A_{ef} = \frac{A}{\sqrt{2}} = 0.707A \quad (2)$$

$$A_{ef}^2 = \frac{1}{T} \int_0^T y^2(t) dt \quad (3)$$

Střídavá střední hodnota amplitudy pro sinusový průběh je:

$$A_{st} = 0.636A \quad (4)$$

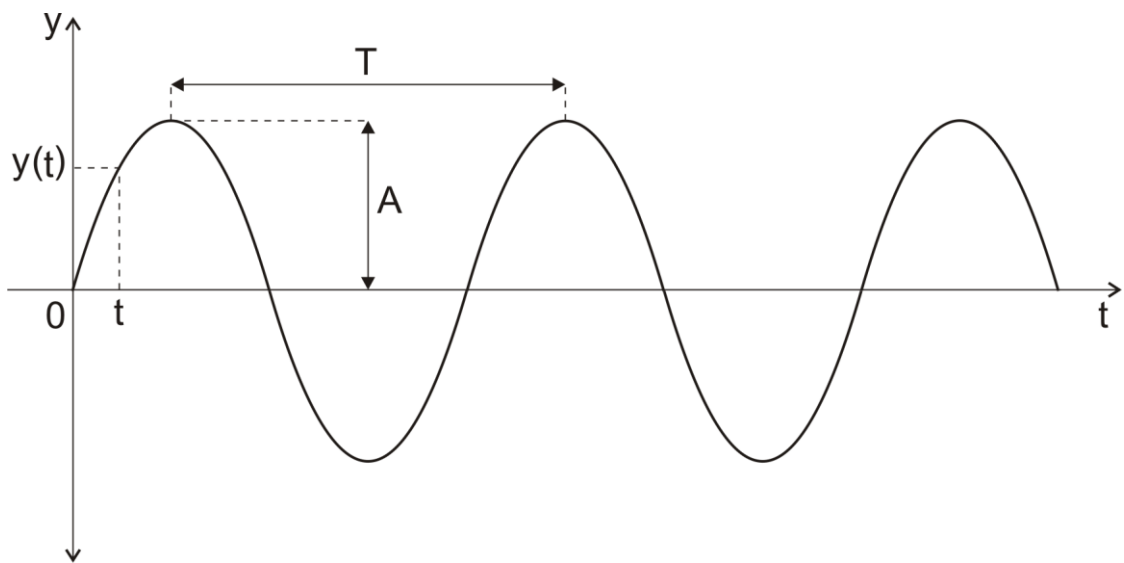
$$A_{st} = \frac{1}{T} \int_0^T |y(t)| dt \quad (5)$$

Okamžitá hodnota

Vertikální výchylka bodu X v aktuálním okamžiku, mnohdy označována symbolem $y(t)$, kde parametr je čas.

Následující diferenciální rovnice popisuje vzájemný vztah mezi výše uvedenými veličinami, popisující harmonické kmitání:

$$\frac{d^2y}{dt^2} + \omega^2 y = 0 \quad (6)$$

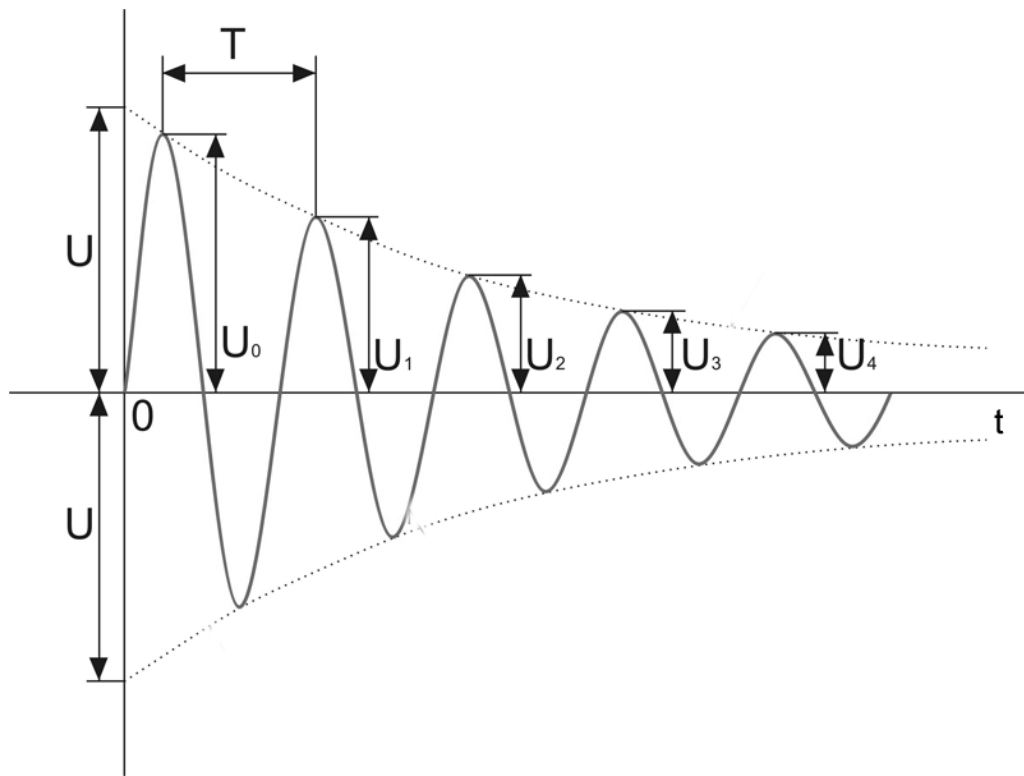


Obr. 2. Sinusoida s vyznačenými parametry

1.1.1.1 Tlumené kmitý

U harmonického pohybu hmotného bodu, který je zavěšený na pružině se mluví o tzv. vlastním kmitání, čili nedochází zde k přispění vnější síly. V přírodě tyto kmitý v netlumené formě nalézt nelze, jelikož v reálném prostředí dochází k vnějším vlivům, jako je například tření. Tření přeměňuje pohybovou energii postupem času na teplo a tím je kmitání tlumeno, až zcela ustane. Amplituda těchto kmitů klesá exponenciálně v závislosti

na velikosti konstanty tlumení. Frekvence tlumených kmitů se také s útlumem snižuje až na hodnotu nula [8].



Obr. 3. Tlumené kmity s vyznačenými parametry

1.1.1.2 Nucené kmity

Pokud dochází k působení vnější periodické síly, označované též jako budící kmity, na hmotný bod nebo soustavu hmotných bodů, vykonává soustava tzv. nucené kmity. Tato vnější síla nahrazuje ztrátu při působení tření [8].

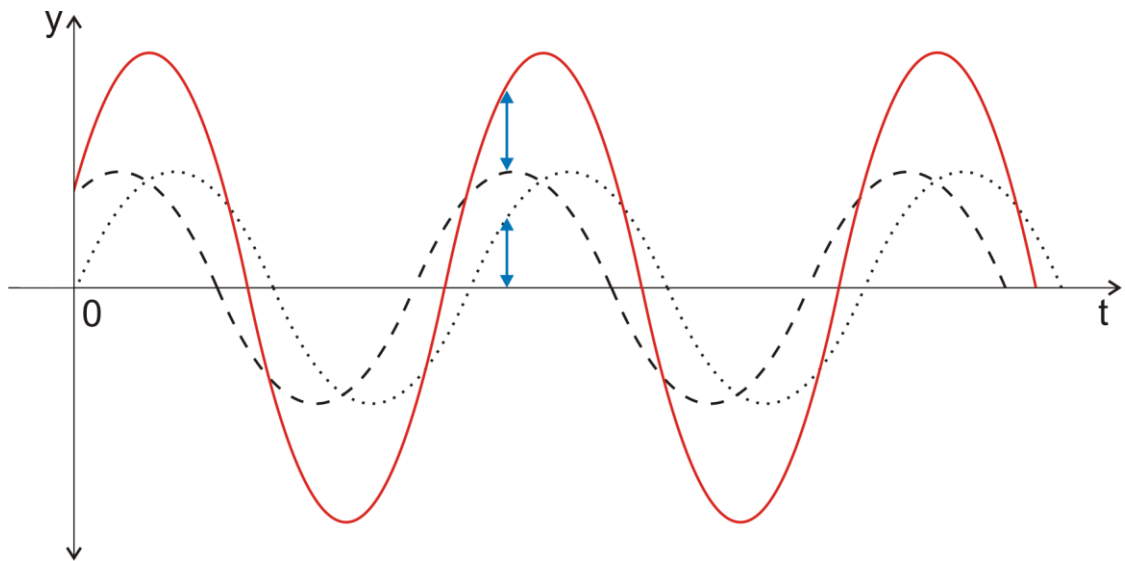
1.1.1.3 Skládání kmitů

O skládání kmitů se dá hovořit v případě, kdy má soustava vykonávat více nucených kmitů najednou. Působí-li na soustavu současně dvě síly, které jsou na ní schopny vyvolat samostatný harmonický pohyb, pak se jedná o složené kmitání. To může mít libovolný průběh: harmonický i neharmonický. Pokud nucené kmity mají stejné parametry, pak se frekvence výsledného signálu nemění a výsledkem bude harmonický signál. V opačném případě, kdy nucené kmity nebudou stejných parametrů, výsledkem bude signál neharmonický, který je ovšem periodický. Při skládání kmitů o různých frekvencích, amplitudách platí princip superpozice. Ten je definován jako: Koná-li hmotný bod

současně dva nebo více harmonických pohybů v jedné přímce s okamžitými výchylkami y_1 , y_2 , pak je okamžitá výchylka výsledného kmitání rovna [8]:

$$y = y_1 + y_2 + \dots + y_n \quad (7)$$

Na principu superpozice je založena metoda grafického sčítání.



Obr. 4. Skládání fázově posunutých kmitů

1.1.2 Vlnění

Mechanické vlnění je zvláštním případem kmitavého pohybu pružného prostředí. V takovém prostředí jsou částice mezi sebou vázány pružnými silami. Rozkmitá-li se některá částice prostředí, pak se postupně rozkmitávají prostřednictvím zmíněných pružných sil sousední částice a ty zase rozkmitávají další a rozruch se šíří prostředím. Vlnění je charakterizováno šířením vln, které přenáší zvukovou energii. Nejjednodušším případem vlnění je šíření rozruchu pouze v bodové řadě. Pokud prvotní rozruch rozkmitá bod řady v kolmém směru na tuto řadu, pak také i ostatní body řady se rozkmitají kolmo – příčně na směr šíření vln. V takovém případě se jedná o postupné vlnění příčné, u kterého šířící vlna vykazuje maxima a minima [8].

Veličina, která je charakteristická pro toto vlnění je vzdálenost, jakou šířící vlna urazí za dobu jedné periody – vlnová délka. Vlna se šíří konstantní rychlostí a platí pro ni následující vztah:

$$\lambda = c \cdot T \quad (8)$$

kde T je doba periody v sekundách a c je rychlost šíření vln v metrech za sekundu.

Dalším případem vlnění je postupné podélné šíření. Zde kmitají všechny body řady ve směru šíření vlnění. Výchylka každého bodu pak leží přímo v bodové řadě, na které se nevytvoří maxima a minima. Ale v závislosti na souřadnicích x a y – zhuštění a zředění. Zhuštění je místo, kde jsou body blíže u sebe a zředění naopak kde jsou body dál od sebe [8].

1.2 Zvuková analýza

Digitální podoba zvukového signálu dovoluje převést obecný problém jeho analýzy do roviny numerického řešení Fourierových řad a integrálů a dalších vztahů. Digitální analýza pracuje s diskretní podobou Fourierovy transformace DFT, která je adekvátní vztahům (9) a (10), a (11) platí pro diskretní časovou funkci $g(n)$ [8].

$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} F(\omega) e^{j\omega t} d\omega \quad (9)$$

$$F(\omega) = \int_{-\infty}^{+\infty} f(t) e^{-j\omega t} dt \quad (10)$$

$$g(n) = \sum_{k=0}^{N-1} G(k) e^{j\frac{2\pi kn}{N}} \quad (11)$$

A pro diskretní spektrální funkci $G(k)$

$$G(k) = \frac{1}{N} \sum_{n=0}^{N-1} g(n) e^{-j\frac{2\pi kn}{N}} \quad (12)$$

- kde n ...odpovídající vzorek v časové doméně
 k ...odpovídající vzorek ve frekvenční doméně
 N ...počet vzorků signálu

U digitální analýzy odpovídá vzorkovanému signálu v časové doméně obdobně vzorkované spektrum ve frekvenční doméně.

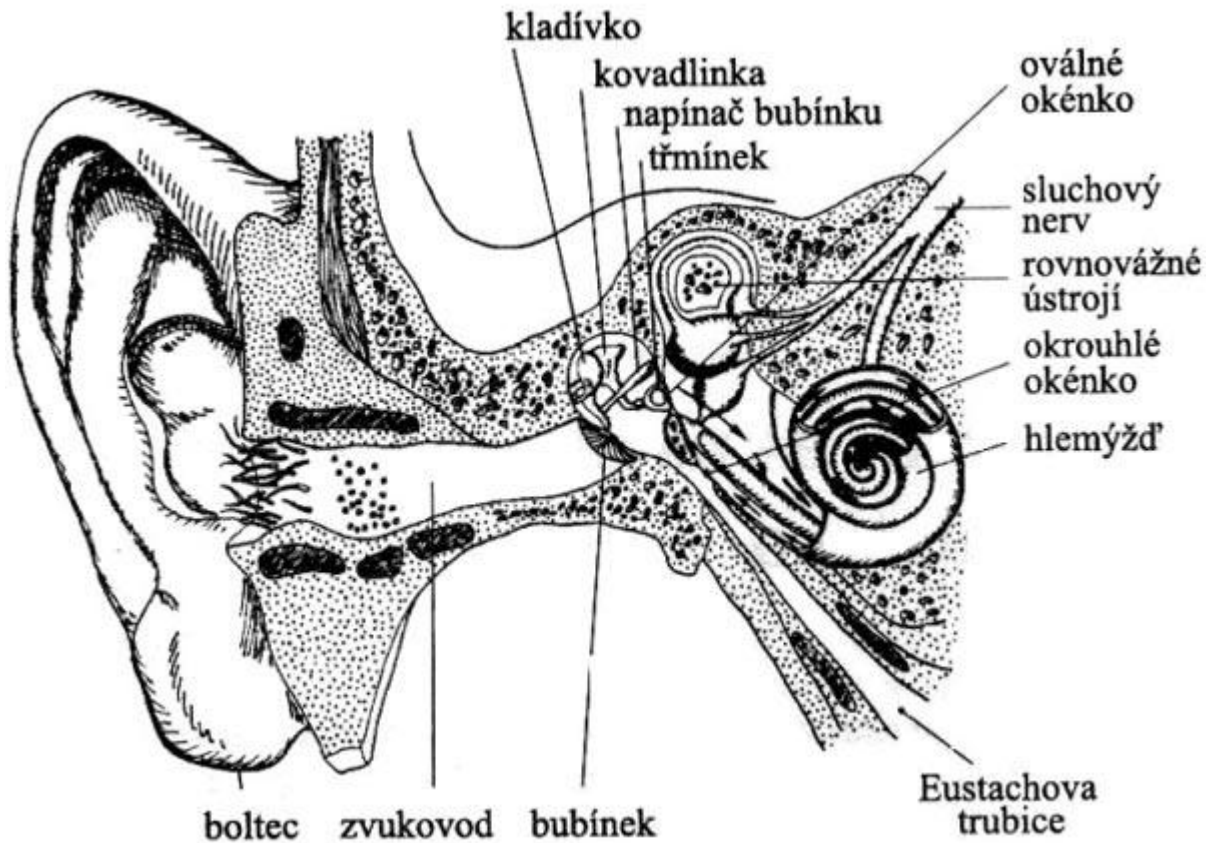
Digitální analýza používá z důvodu urychlení výpočtu spektra algoritmus rychlé Fourierovy transformace *FFT* (Fast Fourier Transform), který ve své nejjednodušší podobě pracuje s délkou časového okénka vyjádřené počtem vzorků v mocninách 2. Diskontinuity vzniklé pevným vymezením délky okénka (např. na 4096 vzorků) se eliminují jeho vhodným tvarem, který ke krajům okénka postupně zmenšuje velikost vzorků nejčastěji na nulovou hodnotu. Použití Hanningova okénka podstatně sníží úroveň zkreslujících frekvenčních složek ve výkonovém spektru, zejména mezi harmonickými nižších pořadových čísel, avšak pozmění fázové spektrum. Míra potlačení zkreslujících složek a změn ve fázovém spektru je závislá na typu (tvaru) časového okénka, proto se v praxi vybírá okénko z několika desítek ustálených typů podle konkrétního zaměření zvukové analýzy [8].

1.3 Fyziologická akustika

Fyziologická podstata zvuku je taková, že člověk za pomoci sluchového ústrojí, který zprostředkovává přenos zvukové informace do lidského vědomí, vnímá kmitavý pohyb hmoty. Ten představují částice vzduchu v blízkosti lidského ucha. Tyto vlastnosti pak v lidském vědomí představují sluchový vjem. Lidské ucho vnímá zvuk v rozmezí 16Hz – 20kHz a největší citlivost vykazuje mezi frekvencemi 1000 – 3000Hz. Ovšem s věkem se tato rozmezí zužují.

Zvuková informace prochází nejprve v našem vnějším uchu podobou kmitavého pohybu částic, dále pak mechanickým přenosem vibrací uvnitř středního ucha. Ve vnitřním uchu dochází k šíření vln v kapalném prostředí a nakonec ve smyslových buňkách dochází k elektrickým dějům. Vnější ucho je tvořeno boltcem a zvukovodem. Boltce díky svým záhybům plní funkci směřování akustických vln dovnitř zvukovodu. Ten si lze představit jako chrupavčitou a kostěnou trubici, která je zakončená blanou bubínku. Bubínek zvuk

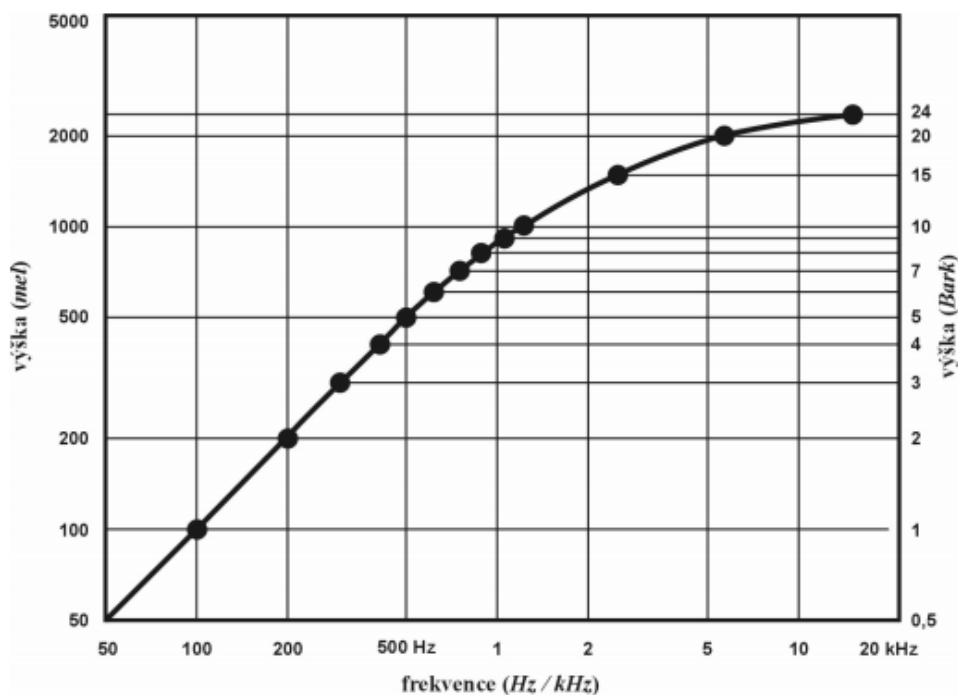
zesiluje a předává dál do středního ucha. Převodní systém středního ucha je tvořen třemi sluchovými kůstkami a to kladívkem, kovádlíčkem a třmínkem. Ty převádí zvuk z vnějšího do vnitřního ucha a jsou umístěny v dutině vyplněné vzduchem. Zde je důležitá funkce Eustachovy trubice, která při každém polknutí vyrovnává tlak s vnějším okolím. Vnitřní ucho začíná oválným okénkem, které je obklopeno nejtvrdějším kostí lidského těla. Skládá se z rovnovážného systému sloužící k detekci poloh.



Obr. 5. Stavba lidského ucha [8]

2 HUDEBNÍ TEORIE

Tón je základní stavební prvek celé hudební teorie a v hudbě má hlavní význam. Dal by se definovat jako zvuk, který vzniká pravidelným chvěním tělesa. Charakterizuje se svou výškou, délkou, intenzitou a barvou. Například celý tón je mezi tónem C - D. Ovšem mezi ním jsou ještě kratší intervaly C - C# a C# - D. Tyto intervaly se označují jako tzv. *půltóny*. Ovšem při potřebě ladění je zapotřebí i menší jednotky jako je půltón. Proto vznikla jednotka *cent*, která je 1/100 půltónu. Cent není, jak by se zprvu zdálo frekvenční, ale logaritmická jednotka a to z důvodu lidského vnímání výšky tónu, které je logaritmické. To má za následek, že odchylka 1Hz například u tónu D1 a D3 není stejná. To lze názorně vyčíst z následujícího obrázku [8].



Obr. 6. Logaritmické vnímání výšky tónu [8]

Pro výpočet počtu centů mezi dvěma známými frekvencemi slouží následující vzorec:

$$100 \times \log_{\sqrt[12]{2}} \frac{f_1}{f_2} = 100 \times \frac{\log \frac{f_1}{f_2}}{\log \sqrt[12]{2}} = \dots \text{centu} \quad (13)$$

Pokud je třeba znát frekvenci tónu, který je od známého tónu vzdálen určitý počet půltónů, lze výše uvedený vzorec upravit do následujícího tvaru:

$$f_2 = f_1 \times q^P \quad (14)$$

$$q = \sqrt[12]{2} \cong 1.059463094 \quad (15)$$

f_2 ...je výsledná frekvence

f_1 ...frekvence známého tónu

q ...konstanta

P ...počet půltónů mezi známým a hledaným tónem

3 TECHNICKÉ PROSTŘEDKY

3.1 Architektura .NET a C#

C# je moderní programovací jazyk určený pro programování pod systémem Windows a navržený speciálně k spolupráci s platformou .NET Framework. Tento jazyk vychází ze zkušeností, které Microsoft získal při přibližně dvacetiletém vývoji podobných objektově orientovaných jazyků. Jádrem platformy .NET je modul CLR (Common Language Runtime). Modul CLR je tzv. běhové prostředí, které vyžaduje překlad fungující tak, že se nejprve provede překlad zdrojového kódu do jazyka IL. Následně je jazyk IL přeložen do kódu pro cílové platformy. Platforma .NET nabízí také spolupráci jazyků [1].

3.1.1 Platformní nezávislost

Znamená to, že využitím jazyka IL a následné kompilace do něj poskytuje .NET nezávislost na platformě obdobně jako technologie Java a její překlad do bajtového kódu. V dnešní době existuje několik implementací platformy .NET na jiných operačních systémech než je OS Windows. Například projekt Portable .NET či Mono, které implementují .NET na GNU/Linux, Mac OS a jiných [1].

3.1.2 Výkon

Výkon jazyka IL lze velice dobře srovnat s jazykem Java. Jazyk IL je překládán metodou JIT (Just-in-time) což znamená, že překlad se provádí nad každou částí kódu právě v okamžiku jejího volání. Výkon v tomto případě znamená, že je pravděpodobné že se všechny části kódu nebudou pro aktuální spuštění potřebné. Na rozdíl od bajtového jazyka Java, kdy dochází k překladu celého kódu při jeho spuštění. Ovšem existují i případy, kdy se Java překládá na některých platformách metodou JIT [1].

3.1.3 Jazyk IL

Jak bylo výše uvedeno kód v jazyce C# se nejprve před jeho spuštěním přeloží do jazyka IL, což je v architektuře .NET hlavní a také základní myšlenkou [1].

Objektová orientace má zde standardní koncepci s jednoduchou dědičností tříd. Proto musí všechny jazyky s výstupem do jazyka IL tuto koncepci splňovat. To ovšem pro jazyk pro

spolupráci s touto platformou samozřejmě nestačí. Pro spolupráci jsou nutné tyto podmínky:

- Dvě třídy napsané v různých jazycích mohou od sebe dědit
- Třída může obsahovat instanci třídy napsané v jiném jazyce
- Metody objektu jednoho jazyka mohou být volány z jiného objektu druhého jazyka
- Mezi metodami je možno předávat objekty nebo odkazy na ně samotné

3.2 PHP

PHP je skriptovací jazyk, který je široce použitelný obzvláště je vhodný pro vývoj webových aplikací a lze jej zapouzdřit do HTML. PHP skripty jsou prováděny na straně serveru a k uživateli je přenášen pouze výsledek. Tento jazyk přebírá částečně syntaxi z jiných jazyků, jako jsou například C, Java a Perl, čímž se stal velice silným nástrojem pro tvorbu složitých webových aplikací [9].

3.2.1 SQL injection

Je hackerská technika využívající bezpečnostní chyby založené na možnosti manipulování s daty, aniž by útočník měl legitimní přístup k databázi. SQL injection je problém všech webových prezentací pracujících s databází ať už načítající svůj obsah z této databáze nebo ho do ní dále ukládají. Zneužitím této chyby se může útočník velmi snadno dostat k citlivým údajům, jako jsou například: rodná čísla, bankovní účty a jiné. Celý princip útoku je velmi snadný a to, podstrčení SQL dotazu do již existujícího dotazu. Existuje několik případů možných způsobů SQL injection [10]:

3.2.1.1 Manipulace s URL

Velmi jednoduchý způsob jak při neošetřeném URL nechtěně přijít nebo dát k dispozici svá data. Následuje příklad nechráněného kódu proti SQL inject z URL adresy, možné způsoby zneužití a bezpečná obrana [10].

3.2.1.2 Pomocí formuláře

SQL inject za pomoci formuláře se děje obdobně jako přes URL, pouze s tím rozdílem, že je třeba chránit vstupy z textového pole. Útočník zadává příkazy stejným způsobem jako přes URL [10].

3.2.2 Vzdálené skriptování (XSS)

Cross-site scripting je jeden z nejstarších hackerských útoků na webové aplikace. Označuje se jako CSS, ale vzhledem k jednoduše zaměnitelnosti se zkratkou kaskádových stylů bývá častěji, a také vhodněji, označována jako XSS. Tato metoda je založena na injektování nebezpečného kódu tam, kde programátor netušil možnost jakéhokoliv útoku. Ačkoliv obrana proti XSS je velmi snadná, je ještě stále mnoho webů, které tuto bezpečnostní chybu neřeší. Jedná se o webové stránky, nebo lépe řečeno aplikace, jako jsou diskusní fóra, návštěvní knihy apod. Jednoduše řečeno, jsou to aplikace, kde uživatel ukládá pomocí vstupu (například textového pole) určitá data, která jsou uložena do databáze a pak následně pravidelně spouštěna. Typickým příkladem Cross-site scriptingu je vložení javascriptu do diskusního fóra, který automaticky přesměrovává všechny uživatele fóra na jiný web. To může být velmi závažné ve chvíli, kdy je web infikován viry nebo trojskými koni a tím pádem každý nepozorný uživatel může být potenciálně v nebezpečí.

Dostatečná ochrana před XSS útoky je obdobná jako před SQL injectem. Pro všechny uživatelské vstupy, ale i pro URL je třeba provést ošetření před zadáním speciálních znaků [10].

3.2.3 MD5 cracking

MD5 je jednosměrná hashovací funkce, sloužící k vytváření otisku hesla tvořeného 32 znaky. Jednosměrná funkce znamená, že vytvoření otisku je snadné například za použití funkce `md5()` v PHP, ale už zpětné zjištění hesla je nemožné. Pro příklad hash slova „heslo“ vypadá následovně: `955db0b81ef1989b4a4dfeae8061a9a6`. V roce 2004 byl zjištěn postup k nalezení kolize dvou řetězců, ovšem to k nalezení původního hesla nestačí. Kolize dvou řetězců v tomto případě znamená, že jelikož je možné hashovat řetězce i velmi dlouhé, vznikají pak hashe, které jsou shodné pro dva různé vstupní řetězce. K prolomení MD5, respektive k nalezení původního řetězce existují čtyři známé způsoby a to: Bruce-force attack, Rainbow tables, Dictionary Attack a MD5 online cracking [10].

3.2.3.1 *Bruce-force attack*

Tento způsob prolomení je způsob tzv. hrubou silou. Jedná se o to, že se vygenerují všechny možné kombinace nad zvolenou abecedou (sada znaků) a pro každého zvlášť se následně vygeneruje hash, který se porovnává se vstupem. Například zvolená abeceda je (A-Z) a počet znaků je 5, tvorba kombinací by pak probíhala následným způsobem:

V tabulce je uveden názorně postup, jak by cyklus prohledával jednotlivé kombinace nad zadanou abecedou. Pokud algoritmus najde shodu, je ukončen a vrací příslušnou kombinaci abecedy [10].

Pro zjištění md5 hashe zadaného řetězce je možné na tomto odkaze:

<http://www.maxiorel.cz/md5-online-generator>

3.2.3.2 *Rainbow tables*

Velice rychlá metoda k nalezení klíčového hashe. Oproti Bruce-force attack je mnohem efektivnější, protože nemusí generovat klíče postupně sám, ale má je již v nějaké tabulce předem generované. Vyhledávání pak v těchto tabulkách je velmi rychlé. Vždy když se najde shoda hashů, tak je u příslušného hashe i jeho řetězec znaků a tím je heslo nalezeno. O všem s rostoucím počtem kombinací prudce roste i datová náročnost celé této struktury [10].

3.2.3.3 *Dictionary attack*

Velice rychlá metoda k nalezení klíčového hashe. Oproti Bruce-force attack je mnohem efektivnější, protože nemusí generovat klíče postupně sám, ale má je již v nějaké tabulce předem generované. Vyhledávání pak v těchto tabulkách je velmi rychlé. Vždy když se najde shoda hashů, tak je u příslušného hashe i jeho řetězec znaků a tím je heslo nalezeno. O všem s rostoucím počtem kombinací prudce roste i datová náročnost celé této struktury [10].

3.2.3.4 *MD5 online cracking*

Způsob prolomení hashe, který je velmi nebezpečný a to v tom, že využívá, jak rychlosti serverů, na níž databáze funguje, tak vysoké objemnosti uložených dat. Je to metoda, která je opět principiálně velmi podobná Rainbow tables, pouze s tím rozdílem že jako datové

úložiště funguje databáze. Vše může fungovat jako webová aplikace přes webový prohlížeč. Výhoda útočníků také může spočívat v tom, že mohou po celém světě generovat obrovské množství řetězců a tím spojit své síly [10].

3.3 DATABÁZE

Termín databáze je v dnešní době velmi frekventované slovo. Přímo souvisí s potřebou lidí shromažďovat data, ať už to je evidence zaměstnavatele o svých zaměstnancích, různé záznamy o pacientech u lékaře nebo u evidence sítě mobilních telefonů. První definice termínu databáze vznikla v 60 letech dvacátého století a to jako uspořádaná množina dat, neboli informací uložených na určitém paměťovém médiu a to v jednom nebo více datových souborech. Strukturám tvořícím databázi se říká tabulky a ty v sobě uchovávají konkrétní data. V dnešní době se lze setkat s mnoha druhy databází, například jsou to Oracle, MSSQL (Microsoft SQL), MySQL, DB2, PostgreSQL a Informix [5].

3.3.1 Tabulka, datový záznam

V popisu databází, byl již zmíněn termín tabulka. Jedná se o strukturu, ve které jsou uložena data. Datovým záznamem je v tomto případě každý řádek tabulky rozdělený do více buněk. Jednotlivé buňky mohou být různých formátů a datových typů. Například jsou-li v databázi uchovávány informace o klientech určité firmy, bude zde bezesporu uloženo jméno, příjmení, telefonní číslo apod. a to by se bez různých datových typů zcela jistě neobešlo. Proto se na jméno a příjmení využije datový typ pro uchování řetězce *varchar* o určité délce, na telefonní číslo datový typ *Int* [5].

3.4 KASKÁDOVÉ STYLY (CSS)

Kaskádové styly, anglickým názvem Cascading Style Sheets (proto zkratka CSS), vznikly okolo roku 1997 a jsou to styly pro formátování obsahu. Výhoda CSS je oddělení vrstev dokumentu, čímž je snazší aktualizace jeho obsahu bez strachu o změnu layoutu. To umožňuje efektivnější práci. Další výhodou, je při použití externí šablony kaskádových stylů, že se může provést kompletní změna celého layoutu, bez toho, aniž by se musel upravovat obsah. CSS se deklaruje třemi odlišnými způsoby.

První, a také asi nejméně vhodná, je přímo u formátovaného elementu pomocí vlastnosti `style=""`. Je to tzv. Přímý styl, který se používá nejméně.

Další možností je pomocí stylesheet v hlavičce stránky. Píše se mezi párové tagy `<style></style>`. Poslední třetí způsob je bezesporu nejpoužívanějším a také nejvhodnějším řešením stylování webových stránek. Jedná se o způsob, kdy se styl píše do externího *.css souboru na který odkazuje tag `<link>`. Syntaxe CSS je jednoduchá, pokud ji ovšem i přes svou jednoduchost programátor nebude znát, s největší pravděpodobností styl nebude fungovat [10].

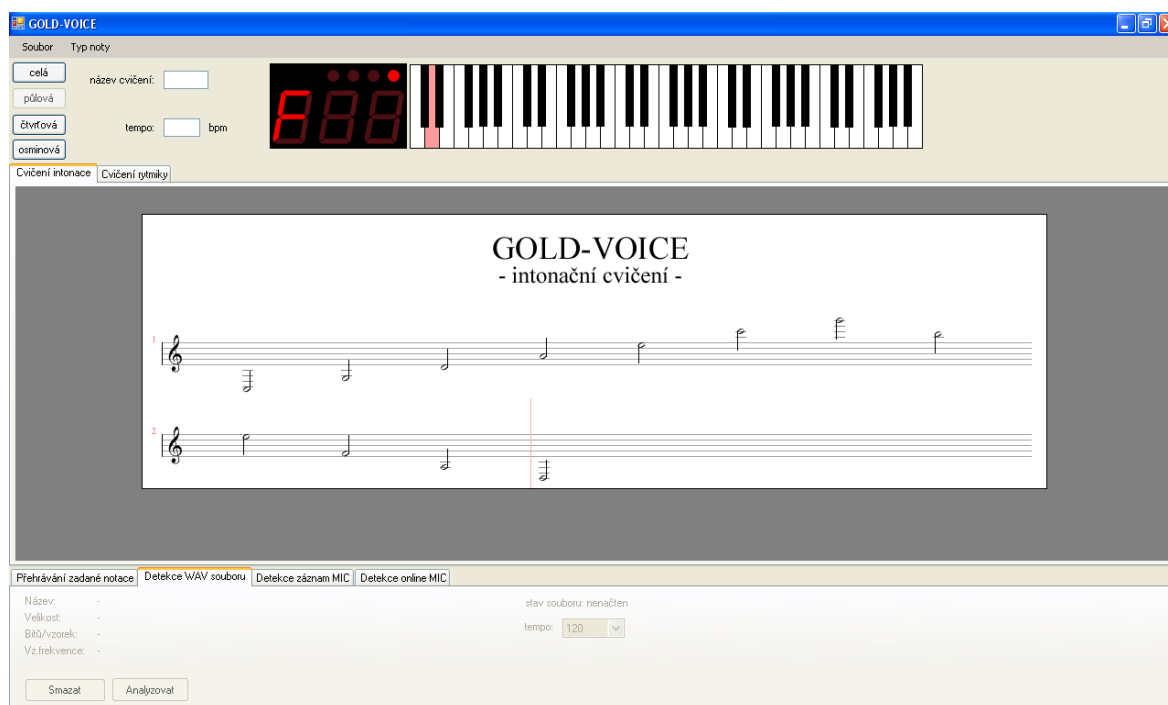
II. PRAKTICKÁ ČÁST

4 POPIS PROGRAMU GOLD-VOICE

Software GOLD-VOICE, určený ke zpracování dat z mikrofonu a jejich následné identifikaci, byl vyvinut ve vývojovém prostředí Visual Studio 2008 v jazyce C# (Framework 3.5). Využívaná technologie při zpracování dat přímo z mikrofonu je *DirectSound* respektive její zapouzdření v *SlimDX*.

Program se ideálně hodí pro tvorbu pěveckých cvičení, analýzu intonace nebo rytmiky. Dále je velmi dobře využitelný při analýze hraných not na hudebním nástroji. Z praktických zkušeností hudebníka je velmi dobře známo, že při skládání zpěvové linky k písni, je hledání not trochu zdlouhavější záležitostí a program GOLD-VOICE tento nedostatek docela úspěšně eliminuje.

V této kapitole je nejprve popsán způsob značení objektů, což je při obsáhlejšímu projektu nesmírně důležité. Zabrání se tak nepřehlednosti ve zdrojovém kódu, jelikož je možné ihned z názvu objektu identifikovat, o kterou část layoutu se jedná. Dále je zde popsán princip notového zápisu, jeho přehrávání, způsob ukládání do speciálního souborového systému GVO a jeho následného otevírání. Po způsobu notového zápisu je zde vysvětlen princip importu wav souboru, jeho analýza rychlou fourierovou transformací a zobrazení vypočtených výsledků. Poslední částí kapitoly je popis zpracování dat z mikrofonu.



Obr. 7. Náhled programu Gold-voice

4.1 Specifikace názvů objektů

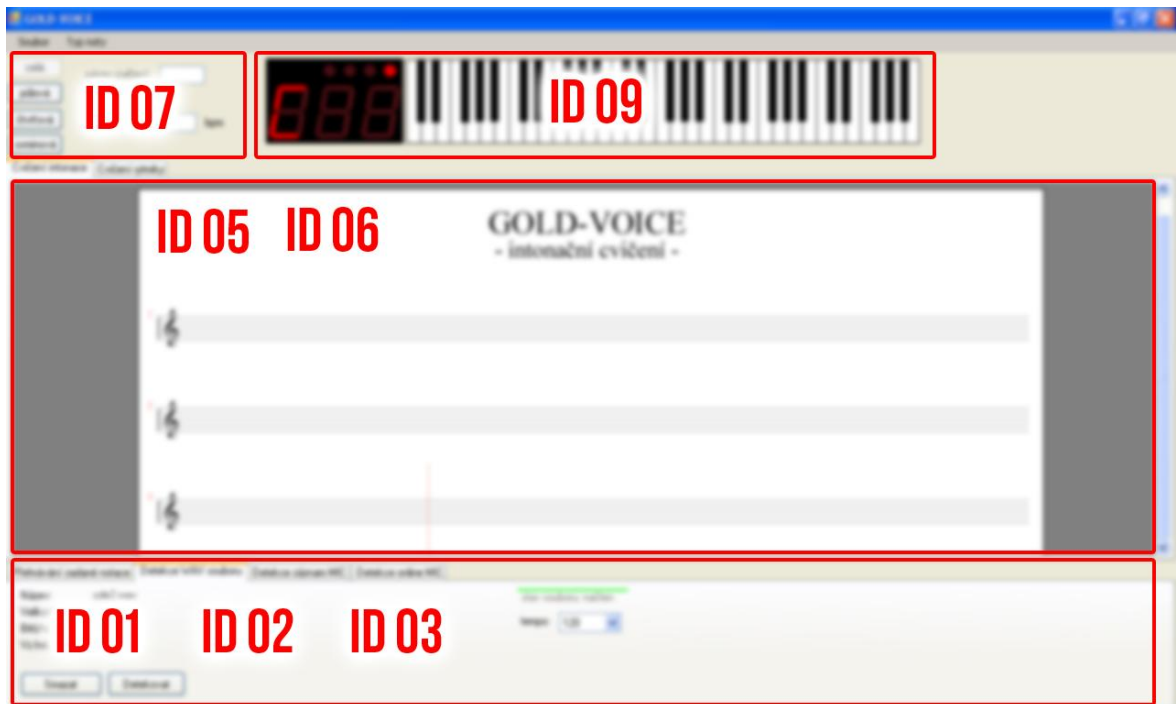
Názvy objektů v programu mají svou přesně danou specifikaci pro jednoznačnou orientaci a přehlednost. Layout je rozdělen do 9 oblastí, které jsou identifikovány označením ID01 - ID09. Všechny objekty ve vybrané oblasti začínají identifikátorem, následuje zkratka objektu oddělená podtržítkem a nakonec počítadlo objektu. Například objekt button situovaný v oblasti ID07 by vypadal následovně: *id07_but_001*.

Seznam zkratk objektů:

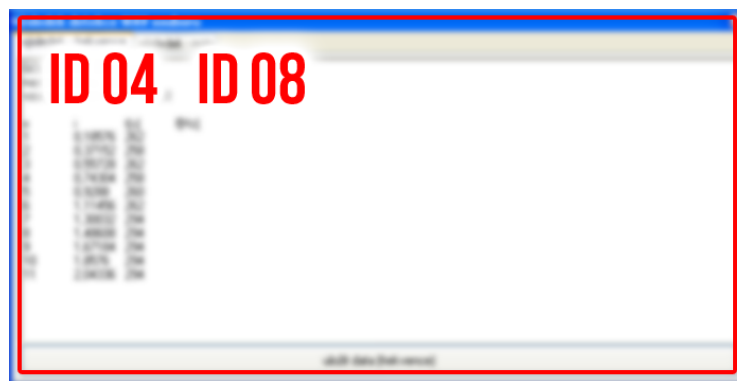
- **but** - button
- **tb** - textBox
- **lbl** - label
- **pnl** - panel
- **pb** - pictureBox

Seznam identifikátorů oblastí:

- **ID01** – oblast formuláře 1 pro přehrávání zadané notace
- **ID02** – oblast formuláře 1 pro analýzu WAV souboru
- **ID03** – oblast formuláře 1 pro analýzu záznamu z mikrofону
- **ID04** – oblast formuláře 2 pro výpis analyzovaných frekvencí
- **ID05** – oblast formuláře 1 pro zobrazení zadané notace
- **ID06** – oblast formuláře 1 pro rytmické cvičení
- **ID07** – oblast formuláře 1 pro zadávání typu not
- **ID08** – oblast formuláře 2 pro výpis analyzovaných not
- **ID09** – oblast formuláře 1 pro klaviaturu a zobrazení



Obr. 8. Rozložení oblastí ve formuláři 1

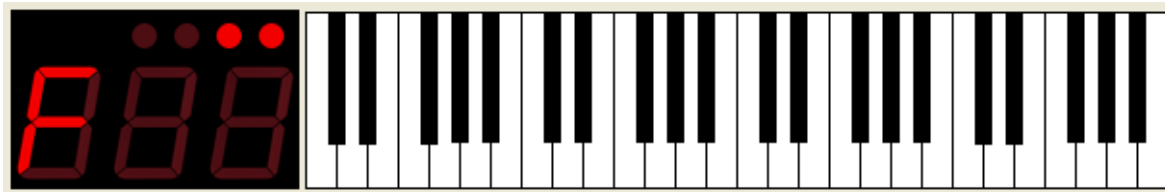


Obr. 9. Rozložení oblastí formulář 2

4.2 Notace

4.2.1 Tvorba notového základu

Notový základ se vytváří pohybem kurzoru po notové osnově a klikem na příslušnou klapku klaviatury. Nejprve při pohybu myši nad klaviaturou dojde k volání metody *Move*, která zobrazí názvy not a označení oktávy na sedmi-segmentovém displeji. Princip je jednoduchý, při inicializaci formuláře jsou načteny všechny bitmapy názvů not do proměnných a po zavolání metody *Move* se příslušná bitmapa vykreslí do *pictureBoxu*. Taktéž se vykresluje i označení oktáv.



Obr. 10. Zobrazovací displej a klaviatura

```
private Bitmap ss_c;
private Bitmap ss_y0;

ss_c = new Bitmap("DP_noty/ss001.bmp");
ss_y0 = new Bitmap("DĚ_noty/ssax.bmp");

private void id09_klv_001_Move(object sender, MouseEventArgs e)
{
    pictureBox3.BackgroundImage = ss_c;
    pictureBox3.Refresh();
    pictureBox4.BackgroundImage = ss_y0;
    pictureBox4.Refresh();
}
```

Jakmile uživatel klikne na klaviaturu, zavolá se metoda *Click* a ta zavolá další dvě funkce. První *zmenaPozadiKlaviaturaCer(1, id)* pro změnu zabarvení klávesy a zpětné vymazání barvy z předešlého výběru klávesy. Druhá funkce zobrazí vybranou notu na notové osnově vykreslovacího plátna.

4.2.2 Zobrazení dat

Pro zobrazení not byl vybrán objekt *pictureBox*. Jako pozadí je mu ve vlastnostech přiřazena bitmapa, která obsahuje čistou notovou osnovu. Jak již bylo zmíněno při tvorbě notového základu, po kliku na klávesu klaviatury se mimo jiné volá metoda *zobrazeniNotyPlatno()*. Její funkce spočívá v načtení bitmapy z instance na třídu *notoveplatno*. Tuto bitmapu vrací metoda *ZobrazitNotu2* s parametry (*typNoty*, *idNota*, *aktualniPozadi*, *posunutiX* a *aktualniRadek*).

- **typNoty** - předává hodnotu o typu noty (půlová, čtvrtěová, osminová). Zde má funkci při načítání bitmapy ze souboru, přesněji při konstrukci jména volaného externího souboru png.
- **idNota** - určuje, o jakou notu se jedná (C, Cis,...) a má stejnou funkci jako *typNoty*.
- **aktualniPozadi** – je typu *Bitmap*. Do ní se pak na určitou pozici uloží vybraná nota.

- **posunutíX, aktuálníRadek** – parametry určující pozici vykreslení noty do notového plátna



Obr. 11. Zobrazení dat na vykreslovacím plátně

4.2.3 Souborový typ GVO

Tento souborový byl vytvořen z důvodu uchování informací o intonačních cvičeních a o jejich samotných notových základech.

Struktura souboru *GVO* je velmi jednoduchá a podobá se jiným typům souborů. Soubor je koncipován do hlavičky a dat. Hlavička obsahuje informace o uložených datech a o souboru obecně. Nachází se zde označení tohoto typu souboru, které má pevnou velikost 4 bajty. U souboru *GVO* ve verzi 1.0 je označení „*GVOI*“. Za označením souboru se na offsetu 4 nachází adresa dat o velikosti 4 bajtů. Jelikož hlavička tohoto souborového typu má různou velikost, z důvodu uchování názvu cvičení s proměnnou délkou, je tento údaj pro správné otevření souboru naprosto nezbytný. Další položkou je 4 bajtové označení taktu, 2 bajty pro jmenovatel a 2 bajty pro číselník taktu. Toto označení není důležité, protože ve verzi programu *Gold-voice 1.0* je počítáno pouze s 4/4 taktem. Ovšem bylo nutné tento údaj zde uvádět, kvůli vyšším verzím programu, kde bude počítáno i s jinými takty například 3/4. Za údajem o taktu následuje 2 bajtová informace o tempu cvičení. Opět se jedná o důležitý údaj, který je využit jak při přehrávání zadaných dat, tak i v detekci zvukových záznamů. Offsetem 14 začíná název cvičení s proměnnou délkou. Pak již následují samotná data určující notový základ. Jedná se vždy o jednotlivé dvojice bajtů.

První z nich určuje typ noty (půlová, čtvrtěová nebo osminová), druhý je identifikátor noty. Například Identifikátor 1 je nota C a identifikátor 45 je nota gis3.

Tab. 1. Struktura souboru GVO

Offset	Funkce	Velikost
0	Označení souboru	4B
4	Adresa dat	4B
8	Takt jmenovatel	2B
10	Takt čítec	2B
12	Tempo	2B
14	název	XB
...	Data	XB

4.2.4 Otevírání GVO souborů

Otevření souboru probíhá klikem v menu na položku Otevřít nebo klávesovou zkratkou Ctrl+O. Tímto se vyvolá privátní metoda *otevřítToolStripMenuItem_Click*, kde před spuštěním dialogového okna pro otevření souboru, dojde k nastavení filtru a jeho popisku. Pak už následuje samotné spuštění a ošetření, zda byl vybrán libovolný soubor typu gvo. Pokud ano, vytvoří se instance na třídu soubor a zavolá se metoda *otevriSoubor*. Pak se pouze načtou jednotlivá data do globálního pole a noty se vykreslí na vykreslovací plátno.

```
private void otevřítToolStripMenuItem_Click(object sender, EventArgs e)
{
    string adresa = "";
    openFileDialog1.Filter = "Gold-voice Files (GVO) |*.gvo;";
    openFileDialog1.Title = "Otevřít GVO soubor";
    openFileDialog1.FileName = "";
    openFileDialog1.ShowDialog();

    adresa = openFileDialog1.FileName;

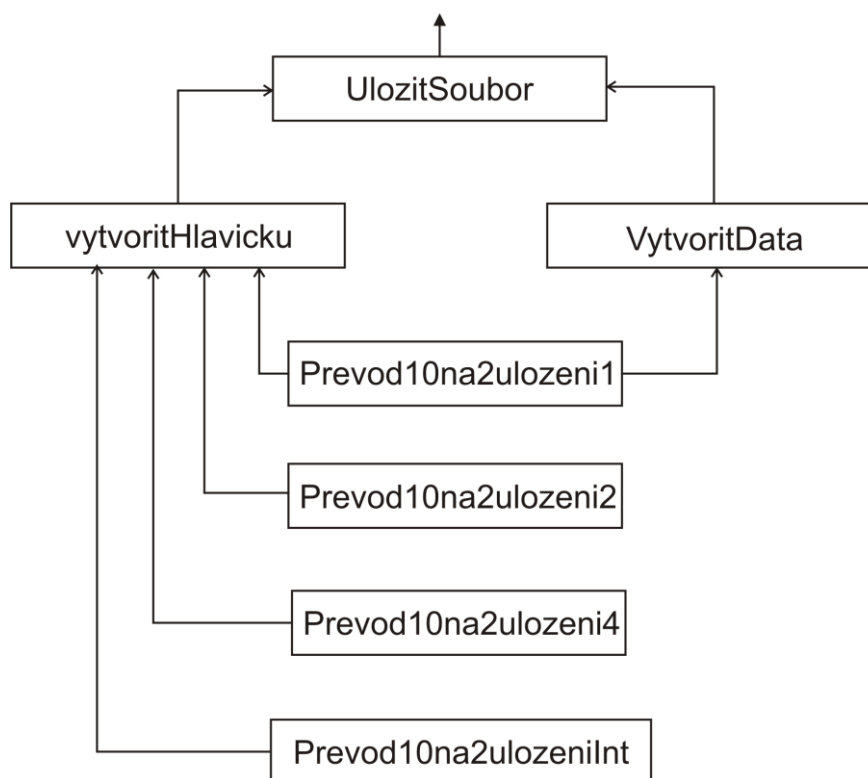
    if (adresa != "")
    {
        soubor soubor = new soubor();
        soubor.otevriSoubor(adresa);
        textBox2.Text = soubor.getNazev;
        textBox4.Text = soubor.getTempo.ToString();
        poleTypuNot = soubor.getNoty;
    }
}
```

V metodě *otevritSoubor* se nejprve vytvoří instance třídy *FileInfo*, díky ní se zjistí velikost souboru, která se pak určí jako konečná hranice cyklu *for* při procházení jednotlivých bajtů souboru. Soubor se čte za pomoci *binaryReaderu* a jednotlivé bajty jsou ukládány do pole typu *byte*. Následuje transformace dat z pole pro využitelnou podobu. To ukazuje následující příklad, kdy je zjišťována adresa dat ze 4 až 7 pozice v poli dat.

```
adresaDat = ((data[7]*256*256*256) + (data[6]*256*256) + (data[5]*256) + data[4]);
```

4.2.5 Ukládání GVO souborů

Ukládání souborů GVO probíhá zpočátku obdobně jako otevírání. Nejprve se klikem v menu spustí dialogové okno pro výběr cesty a názvu k uložení do souboru. Změna přichází při vytvoření instance a volání její metody *ulozSoubor*. Struktura volání pomocných metod při ukládání je následující:



Obr. 12. Schéma volání metod při ukládání souboru GVO

Metoda *ulozitSoubor* vytvoří členskou proměnnou *BinDataUlozeni*, do které se za pomoci funkcí *vytvoritHlavicku* a *vytvoritData* načtou data pro uložení. Je zde využito veřejných funkcí typu *String* *prevod10na2ulozeni1*, *prevod10na2ulozeni2*, *prevod10na2ulozeni4*. Ty

svůj parametr (typu Int) převedou na bity a je-li to potřeba, provede se doplnění nulami. Výsledná data jsou vždy po 8 bitech převáděna vestavěnou funkcí *Convert.ToByte* na bajty, které se vkládají jako parametr metody *Write* u instance třídy *BinaryWriter*. To lze názorně shlédnout v následujícím výňatku zdrojového kódu.

```
try
{
    byte cislo;
    soubor = new FileStream(cesta, FileMode.Create);
    BinaryWriter bw = new BinaryWriter(soubor, Encoding.ASCII);

    for (int i = 0; i < BinDataUlozeni.Length; i += 8)
    {
        cislo =
        Convert.ToByte((BinDataUlozeni.ToString()).Substring(i, 8), 2);
        bw.Write(cislo);
    }

    bw.Close();
}
catch
{
}
finally
{
    soubor.Close();
}
```

4.3 Analýza WAV souboru

4.3.1 Import WAV souboru

Import tohoto souboru je prováděn obdobně jako otevírání souborů gvo. Při kliknutí na příslušnou položku v menu dojde k vyvolání dialogového okna pro import. Zde je nastaven filtr na wav soubory a pokud je soubor vybrán spustí se funkce *otevritWavSoubor*. Zde se nejprve zjistí velikost souboru a jeho název. Všechna data se načtou do pole *binWavSoubor*, který je typu byte. Provede se ověření, zda se opravdu jedná o soubor typu Wav, to nám určují první 4 bajty ze souborové hlavičky. Načtou se data, jako jsou komprese, počet kanálů, vzorkovací frekvence, počet bitů za vteřinu apod. Pro tento program byl zvolen formát wav souborů o vzorkovací frekvenci 44100 (44100 vzorků za vteřinu), mono kanál a počet bitů na vzorek 16. Pokud jsou tyto podmínky splněny, vypíše

se informace do oblasti ID02 a dojde k aktivování tlačítka na analýzu zvukových dat. Tato analýza probíhá za pomoci FFT (rychlé fourierové transformace).

4.3.2 Rychlá fourierova transformace

Základ a jádro tohoto programu tvoří rychlá fourierová transformace (dále jen FFT), která dokáže ze zvukových dat detekovat jednotlivé frekvence v zadaném intervalu. Interval těchto hodnot byl zvolen 4096, protože při pevné vzorkovací frekvenci 44100 vzorků za vteřinu, je tento interval nejvýhodnější. Jako důkaz je dobré si uvést krátký výpočet: Při maximálním tempu 240bpm (240 dob za minutu) trvá jedna doba 0,25 vteřiny. Jelikož je ale v programu myšleno i na osminové noty, nejnižší interval je tedy 0,125 vteřiny. Což je 5512 vzorků za vteřinu.

FFT je v programu zapouzdřena v třídě `fft`. Obsahuje i několik globálních veřejných proměnných, do kterých se načítají vzorkovací frekvence, minimální a maximální frekvence apod. Pro spuštění analýzy je nutné nejprve tyto globální proměnné nadefinovat.

```
public fft instanceFFT;

instanceFFT = new fft();
instanceFFT.vzorkovaciFrekvence = sampleRate;
instanceFFT.minimalniFrekvence = 60;
instanceFFT.maximalniFrekvence = 4000;
```

Nejprve se vytvoří instance na třídu `fft`, nadefinuje se vzorkovací frekvence (44100), minimální frekvence a maximální frekvence v rozsahu 60Hz – 4000Hz, což je pro lidský hlas dostačující rozsah. FFT je spouštěna v cyklu, a vždy se do ní načtou jen data o velikosti vzorku.

```
for (int a = 0; a <= 10; a++)
{
    for (int i = 44 + (velikostVzorku * a); i < velikostVzorku + 44 +
(velikostVzorku * a); i += 2)
    {
        fin[i - 44 - (velikostVzorku * a)] = ((binWavSoubor[i]) +
(binWavSoubor[i + 1] * 256));
    }

    instanceFFT.data = fin;
    instanceFFT.NajitZakladniFrekvenci();
    aktualniFrekvence = (instanceFFT.frekvence * 2);
```

```

        instanceFFT.frekvenceTonu(aktualniFrekvence);
    }

```

Metoda *NajitZakladniFrekvenci* je základní metoda při zjišťování aktuální frekvence. Ta dále volá všechny potřebné funkce z třídy *fft*. Nejprve po definici lokálních proměnných se zavolá funkce pro výpočet spektra *vypocetSpektra*. Tato funkce již volá jednotlivé kroky FFT. Nejdříve funkci pro zjištění délky dat a počtu bitů dat, dále pak volá funkci pro bitovou inverzi *bitovaInverze*.

```

private void bitovaInverze()
{
    dataKomplex = new ComplexNumber[delkaData];
    int z;
    int dalsiBit;
    int obraceni;
    int pocl;

    for (int i = 0; i < data.Length; i++)
    {
        obraceni = 0;
        z = i;

        for (int j = 0; j < pocetBituData; j++)
        {
            DalsiBit = z & 1;
            z = z >> 1;
            obraceni = obraceni << 1;
            obraceni |= dalsiBit;
        }

        pocl = obraceni;
        dataKomplex[pocl] = new ComplexNumber(data[i]);
    }
}

```

Po bitové inverzi následuje funkce pro Cooley-Tukey algoritmus nazvaná *cooleytukey*. Jedná se o tři cykly vnořeny do sebe, první z nich se provádí do velikosti proměnné *pocetBituData*, druhý do velikosti *m*-tého motýlka a třetí do velikosti délky dat. Nejprve dojde k naplnění proměnné *alfa*. Výpočet je oddělen z důvodu rychlosti celého algoritmu, protože v proměnné *alfa* se mění hodnoty pouze při průchodu hlavního cyklu. Bylo by tedy zbytečné umístit výpočet *alfa* přímo do výpočtu $e^{(-2*pi/N*k)}$, který se provádí při každém průchodu druhého cyklu. Výstup z FFT je tedy v komplexním tvaru.

```
private void cooleytukey()
{
    for (int i = 0; i < pocetBituData; i++)
    {
        int m; int n; double alpha;

        m = 1 << i;
        n = m * 2;
        alpha = -(2 * Math.PI / n);

        for (int k = 0; k < m; k++)
        {
            ComplexNumber nasobitel = new ComplexNumber(0,
            alpha * k).PoweredE();

            for (int j = k; j < delkaData; j += n)
            {
                ComplexNumber pom1 = dataKomplex[j];
                ComplexNumber pom2 = dataKomplex[j + m] *
                nasobitel;
                dataKomplex[j] = pom1 + pom2;
                dataKomplex[j + m] = pom1 - pom2;
            }
        }
    }
}
```

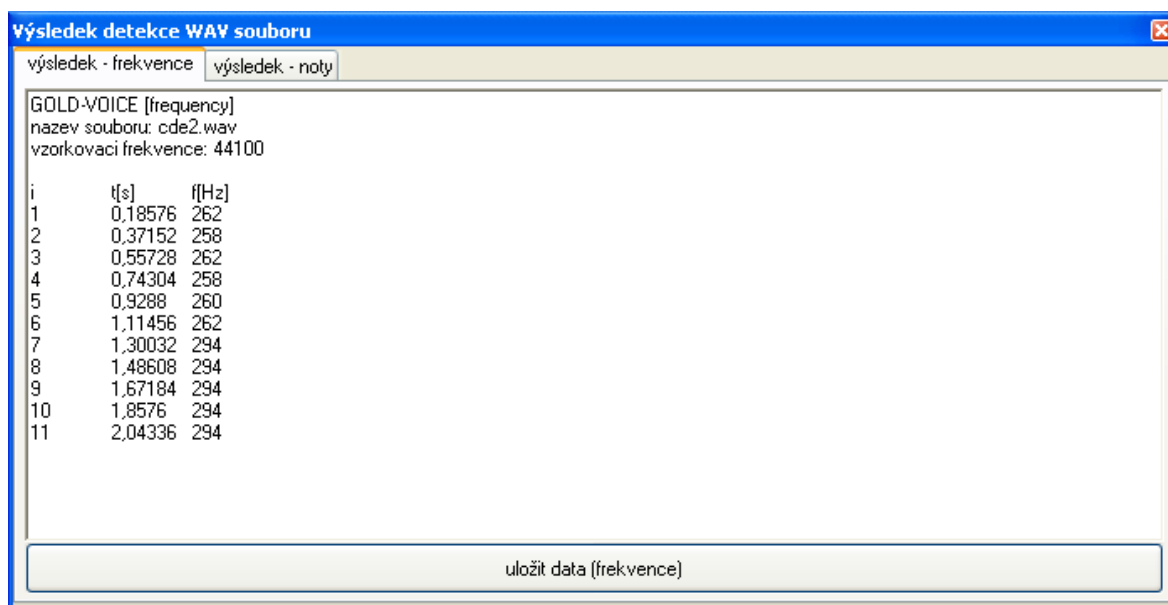
Dalším krokem FFT je získání spektra z komplexních hodnot. Na to v programu slouží metoda *ziskatSpektrum*, která nejdříve inicializuje globální proměnnou *spektrum* s délkou podle délky dat. Samotný převod se provádí součtem druhých mocnin reálných a imaginárních složek.

```
private void ziskatSpektrum()
{
    spektrum = new double[delkaData];
    for (int i = 0; i < spektrum.Length; i++)
    {
        spektrum[i] = dataKomplex[i].AbsPower2();
    }
}
```

V získaném spektru se funkcí *NajitPozicePiku* najdou pozice píků a z nich pak jejich hodnoty, a vybere se ten s největší četností.

4.3.3 Zobrazení výsledků

Výsledky se zobrazují v novém formuláři s názvem *formvysledek*. Po dokončení FFT se vytvoří instance na tento formulář, který má trochu jiné vlastnosti jako hlavní formulář. Například není možné ho minimalizovat nebo měnit jeho velikost. Chová se tedy jako dialogové okno. Na tomto formuláři je situován objekt *tabControl* s dvěma záložkami. První zobrazuje jednotlivé frekvence, tak jak se měnily v čase. Jeden krok je interval vzorkování FFT. Kliknutím na tlačítko dole, je možné data vyexportovat do textového souboru. S tímto souborem se dá pak velmi dobře pracovat například v programu Microsoft Excel a data tak lépe analyzovat například formou grafu. Totéž umožňuje i druhá záložka objektu *tabControl*, pouze s tím rozdílem, že zobrazuje jak jednotlivé tóny, tak i odchylky od těchto tónů v procentech. Je důležité uvést, že 100% odchylka je čtvrt tónu. Kladná hodnota je nad tónem, záporná pod tónem.



Obr. 13. Výsledek analýzy WAV souboru

4.3.4 Ukládání výsledků

Ukládání výsledků, jak již bylo zmíněno, probíhá klikem na tlačítko ve spodní části formuláře. Vyvolá se dialogové okno pro výběr cesty exportovaného souboru a spustí se funkce *ulozSouborTxtFrekvence* s dvěma parametry. První je typu *String* a úkolem je předat adresu exportovaného souboru. Druhý parametr označuje, která záložka se má uložit. Tělo funkce je následující. Nejprve se vytvoří *TextWriter* sloužící k jednoduchému zápisu dat do souboru. Dále je zde definováno pole typu *String*. Jeho každá buňka

představuje jeden řádek *richTextBoxu*. Uvnitř podmínky rozdělující data z obou záložek se data načtou. Ve vestavěné funkci *foreach*, dovolující procházet kompletní pole se načtou z jednotlivých buněk pole do *TextWriteru*. Výjimky této akce odchytává metoda *try-catch-finally*.

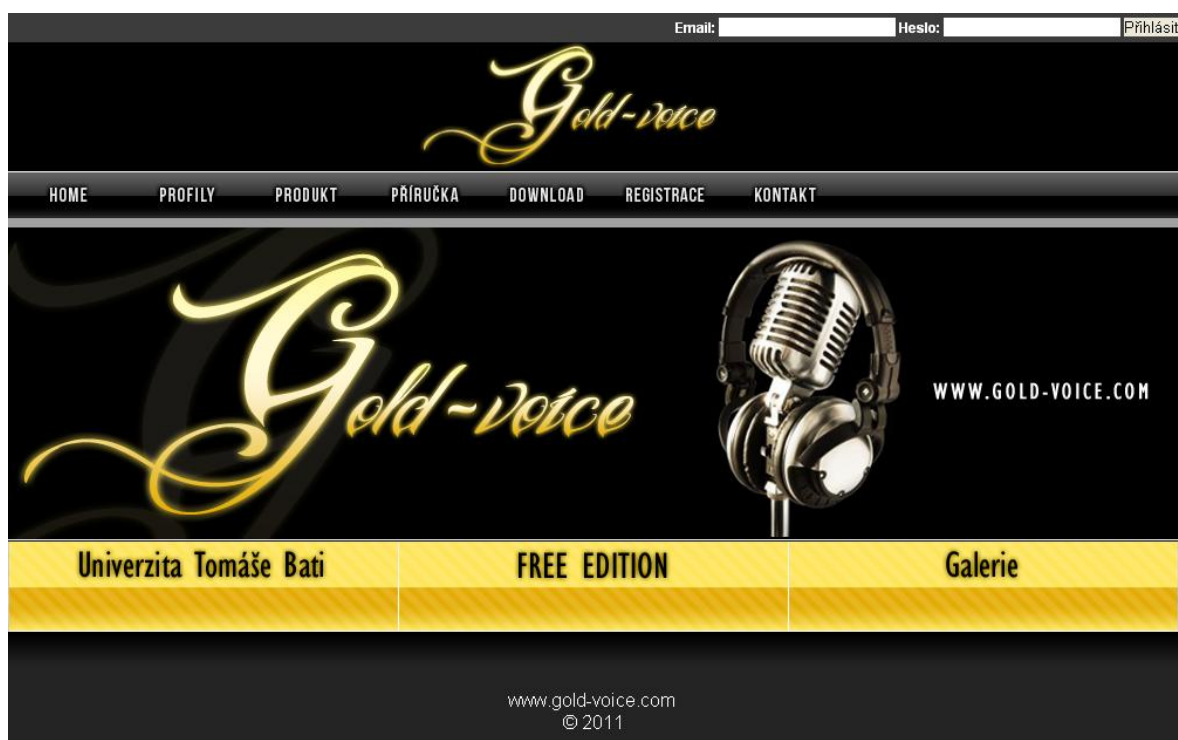
```
private void ulozSouborTxtFrekvence(string adresa, int x)
{
    TextWriter ulozSoubor = new StreamWriter(adresa);
    string[] RichTextBoxLines;

    if (x == 1)
    {
        RichTextBoxLines = id04_rb_001.Lines;
    }
    else
    {
        RichTextBoxLines = id08_rb_001.Lines;
    }

    try
    {
        foreach (string line in RichTextBoxLines)
        {
            ulozSoubor.WriteLine(line);
        }
    }
    catch
    {
    }
    finally
    {
        ulozSoubor.Flush();
        ulozSoubor.Close();
    }
}
```

5 DYNAMICKÁ WEBOVÁ PREZENTACE

K programu GOLD-VOICE byla vytvořena i dynamická webová prezentace s možností personalizace a uploadu pěveckých cvičení. Celá prezentace je napsána v programovacím jazyku PHP 5.0 plně objektově a využívá databáze typu MySQL 5.0. Poskytovatelem webhostingu je firma *Forpsi*, u které byla také zřízena doména WWW.GOLD-VOICE.COM. Typ webhostingu je Normal hosting s prostorem 5GB a operačním systémem Linux. Linux se využívá především pro aplikace v jazyce PHP.



Obr. 14. Náhled webové prezentace

5.1 Registrace

Tabulka určená pro registraci je nazvaná `gvo_registration` a je tvořena 9 sloupci (`id_registration`, `klic_registration`, `email_registration`, `heslo_registration`, `nick_registration`, `datum_registration`, `potvrzeni_registration`, `sstring_registration` a `ip_registration`). Sloupec `id_registration` slouží pro jednoznačnou identifikaci uživatele a je nastaven jako primární klíč, což znamená, že je tato hodnota je v sloupci jedinečná. Tomuto sloupci je také nastavena automatická inkrementace vestavěnou SQL funkcí `auto_increment`. To při vložení nových dat přičte vždy k tomuto číslu jedničku. `klic_registration` je sloupec, který

je naplněn speciálními klíči využité při registraci. Jakmile uživatel nemá stejný registrační klíč jako je uveden v této tabulce, registrace mu bude odepřena. Sloupce `email_registration` a `heslo_registration` najdou své využití při následném přihlašování uživatele. `Nick_registration` je sloupec zobrazovaný při výpisu jmen uživatelů. Ostatní sloupce uchovávají data zejména při přihlašování.

Tab. 2. Struktura tabulky `gvo_registration`

Sloupec	Typ	Porovnání	Extra
<code>id_registration</code>	<code>int(10)</code>		<code>auto_increment</code>
<code>klic_registration</code>	<code>varchar(32)</code>	<code>latin2_general_ci</code>	
<code>email_registration</code>	<code>varchar(45)</code>	<code>latin2_general_ci</code>	
<code>heslo_registration</code>	<code>varchar(40)</code>	<code>latin2_general_ci</code>	
<code>nick_registration</code>	<code>varchar(40)</code>	<code>utf8_czech_ci</code>	
<code>datum_registration</code>	<code>datetime</code>		
<code>potvrzeni_registration</code>	<code>int(1)</code>		
<code>sstring_registration</code>	<code>varchar(40)</code>	<code>latin2_general_ci</code>	
<code>ip_registration</code>	<code>varchar(15)</code>	<code>latin2_general_ci</code>	

Třída `registration` zapouzdřující registraci je v souboru `registration.php`. Obsahuje čtyři metody, dvě veřejné (`drawRegistrationForm`, `registration2`) a dvě privátní (`sql_inject`, `osetreniemail`). Metoda `drawRegistrationForm` vykresluje formulář s 5 textovými poli pro zadání registračního klíče, nicku, emailu, hesla a potvrzení hesla. Po kliku na tlačítko submit je volána další veřejná metoda `registration2`. Ta má za úkol ošetření předešlého formuláře. Nejprve zda jsou vyplněna všechna pole, dále ošetření vůči `sql_injection` za pomoci ostatních dvou privátních metod. Jakmile ošetření projde v pořádku, aktualizuje se výše uvedená tabulka hodnotami z formuláře.

V následujícím výňatku zdrojového kódu je názorně představen princip ošetření proti SQL injection. Superglobální proměnná `$_POST` je parametrem vestavěné funkce `htmlspecialchars`, která převádí speciální znaky na jejich entity a privátní metoda převede znaky `&`, `#`, `,`, `;` na znak `E`. Výsledek se pak porovnává s neošetřenou superglobální proměnnou.

```
$this->sql_inject(htmlspecialchars($_POST["rklic"]))
```

```
private function sql_inject($testinject
{
    $testinject = strtr($testinject, "&", "E");
    $testinject = strtr($testinject, "#", "E");
    $testinject = strtr($testinject, " ", "E");
    $testinject = strtr($testinject, ";", "E");
    return ($testinject);
}
```

5.2 Přihlašování

Přihlašování je zapouzdřeno v třídě *prihlaseni* a princip je následující. V horní hlavičce stránky se vytvoří instance na třídu *prihlaseni* a provede se ověření, zda je uživatel přihlášen či nikoliv. Pokud přihlášen nebude, zobrazí se metodou *zobraz_formular_prihlaseni* formulář přihlášení. Jakmile se bude chtít uživatel přihlásit, vyplní příslušný formulář a data odešle. V hlavičce se tato data odeslaná metodou *POST* odchyťávají a dále zpracovávají. Metoda *prihlaseni* nejprve provede ochranu proti SQL útokům, položí dotaz o shodnosti dat v tabulce, a když se dojde ke shodě, aktualizuje se řádek tabulky. Následně je proveden *refresh*, který má za úkol načíst znova celou stránku. Po znovunačtení stránky se znova vytvoří instance třídy a zavolá se její konstruktor. Ten zavolá funkci *test_prihlaseni* mající za úkol otestovat, zda je uživatel přihlášen. Test proběhne opět položením dotazu s podmínkami na tabulku *gvo_registration*. Jestli má výstup jeden řádek nastaví se veřejné metodě *prihlasen* hodnota 1 a provede se opět aktualizace tabulky. Je zde využito speciálních proměnných *\$_SESSION* pro udržení informace o emailu a nicku. V následující ukázce kódu je možné vidět metodu *prihlaseni*, která provádí prvotní krok po odeslání dat z přihlašovacího formuláře – ošetření a ověření.

```
public function prihlaseni()
{
    if((isset($_POST["femail"])) && (isset($_POST["fheslo"])))
    {
        $this->email = $this->sqlinject(htmlspecialchars(
$_POST["femail"]));

        $this->heslo = $this->sqlinject(htmlspecialchars(sha1(
$_POST["fheslo"]));

        echo $this->email." ".$this->heslo;
```

```
$dotaz="SELECT id_registration FROM gvo_registration WHERE
email_registration='".$this->email.'" AND heslo_registration='".$this-
>heslo.'"';

$vsledek = @mysql_query($dotaz);

if(mysql_num_rows($vsledek) == 1)
{
    $this->stringnick = sha1(uniqid(rand()));
    $dotaz = "UPDATE gvo_registration SET
sstring_registration='".$this->stringnick."',ip_registration='".$this-
>ip."',datum_registration=now() WHERE email_registration='".$this-
>email.'" AND heslo_registration='".$this->heslo.'"';

    mysql_query($dotaz);
    $_SESSION["stringnick"]=$this->stringnick;
    $_SESSION["femail"]=$this->email;

    echo "<meta http-equiv='Refresh' content='0, URL=index.php'>";
}
else
{
    echo "Špatně vyplněné jméno nebo heslo";
}
}
```

5.3 Personalizace uživatelů

Web umožňuje i personalizaci uživatelů. Je zde možno nastavit jméno, příjmení, věk a uploadovat jednotlivá cvičení. Tyto informace jsou pak zobrazeny v záložce profily.

6 TESTOVÁNÍ PROGRAMU

6.1 Testování intonace

Testování programu proběhlo na několika testovacích souborech, v kterých byl vytvořen sled tónů C-DUR za sebou. První byl soubor *cdur_midi_klavir03.wav*, který byl vytvořen v programu Guitar Pro, exportován do midi souboru a nakonec programem Format Factory převeden do wav souboru s parametry vhodnými pro import do programu Gold-voice. Druhý testovací soubor byl *cdur_midi_viola01.wav*, kde jako nástroj byla použita viola. Třetí testovací soubor byl nazpíván 3 pokusy: *cdur_zpev001.wav*, *cdur_zpev002.wav*, *cdur_zpev003.wav*.

Z výsledků byl dokázán předpoklad, že zpěv není intonačně tak přesný jako MIDI nástroj. Všechny výsledky byly exportovány do textových souborů nazvaných obdobně jako testovací soubory.

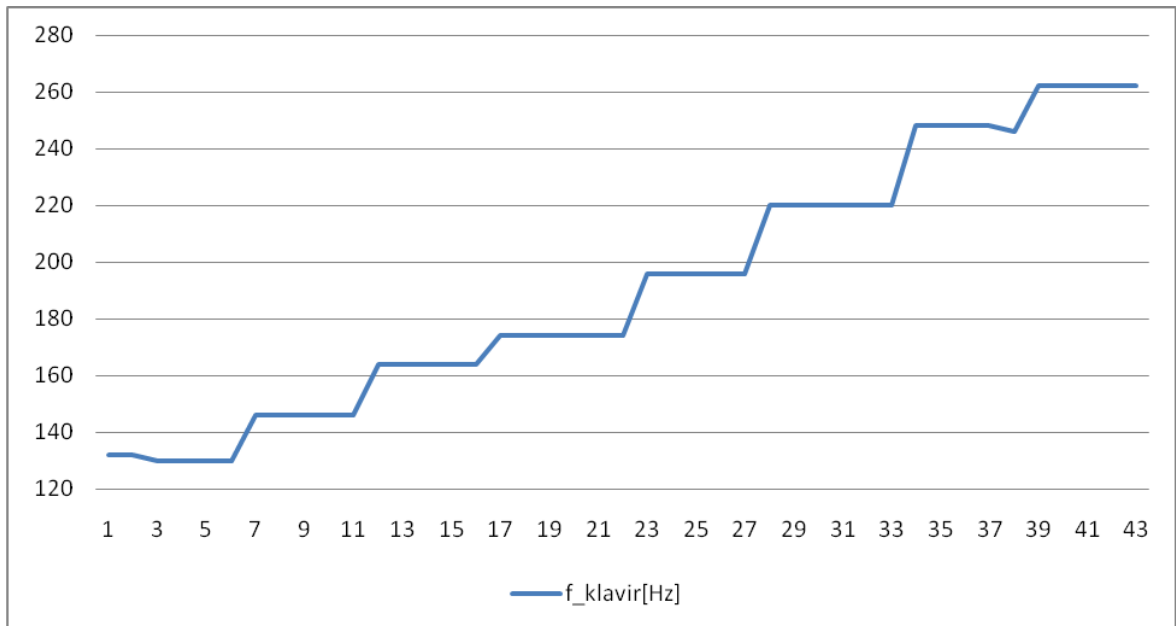
Všechny testovací soubory wav mají parametry:

- Vzorkovací frekvence: 44100Hz
- Počet kanálů: 1 (Mono)
- Počet bitů na vzorek: 16

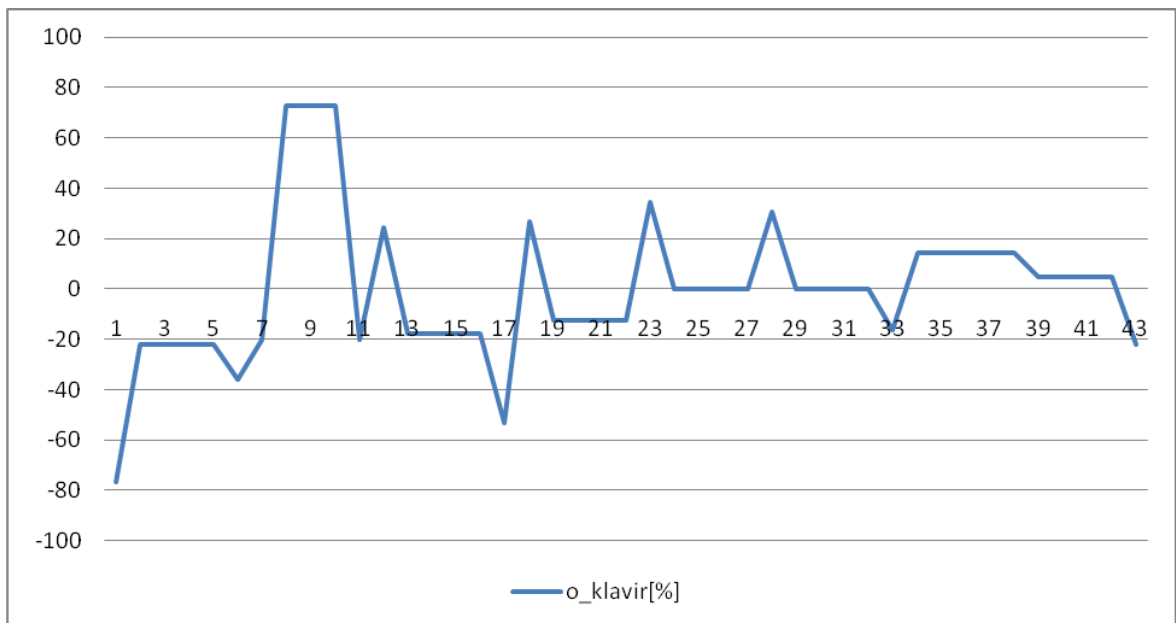
Testovací soubory:

- *cdur_midi_klavir03.wav*
- *cdur_midi_viola01.wav*
- *cdur_zpev001.wav*
- *cdur_zpev002.wav*
- *cdur_zpev003.wav*

6.1.1 Analýza C-DUR MIDI klavír

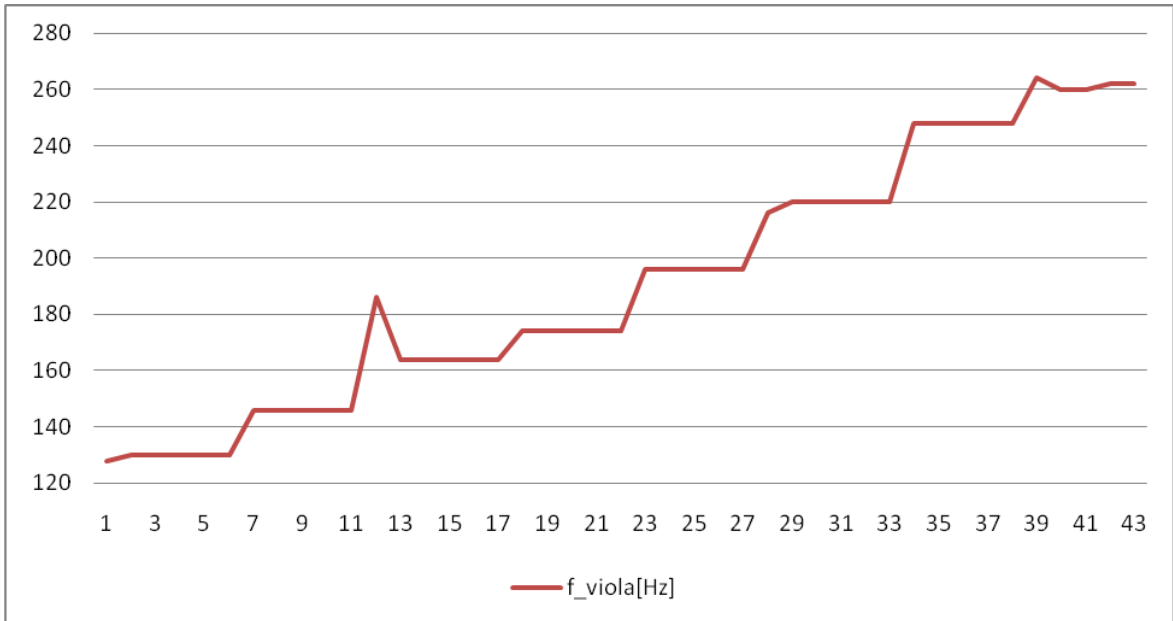


Obr. 15. Grafické zobrazení analyzovaných frekvencí MIDI klavíru

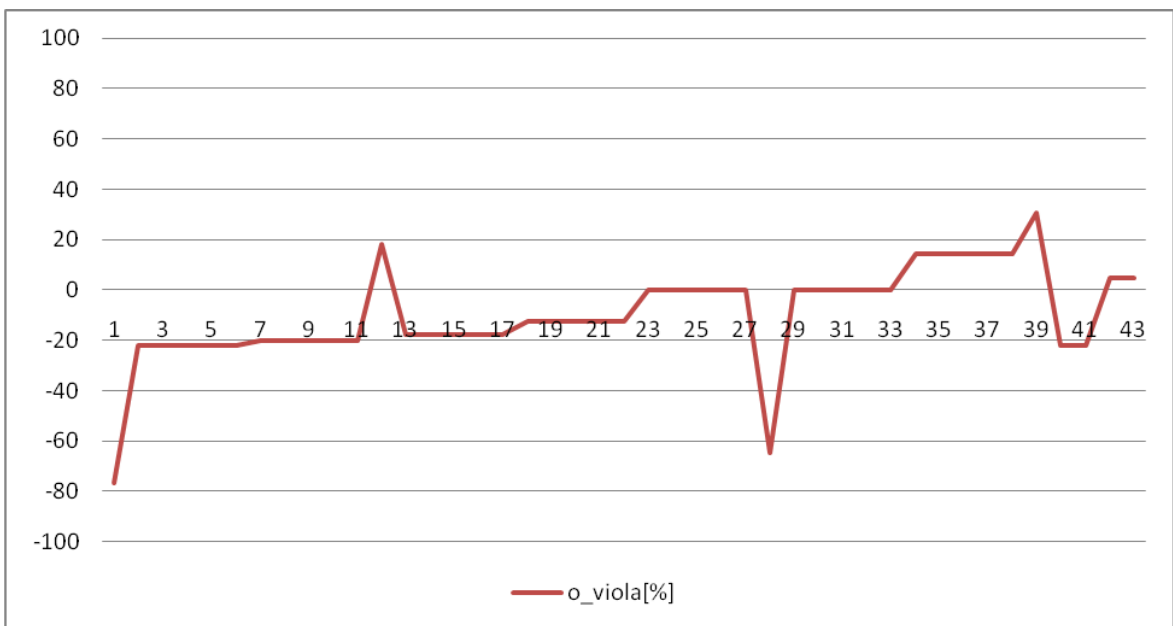


Obr. 16. Grafické zobrazení odchylek od nejbližších tónů MIDI klavíru

6.1.2 Analýza C-DUR MIDI viola

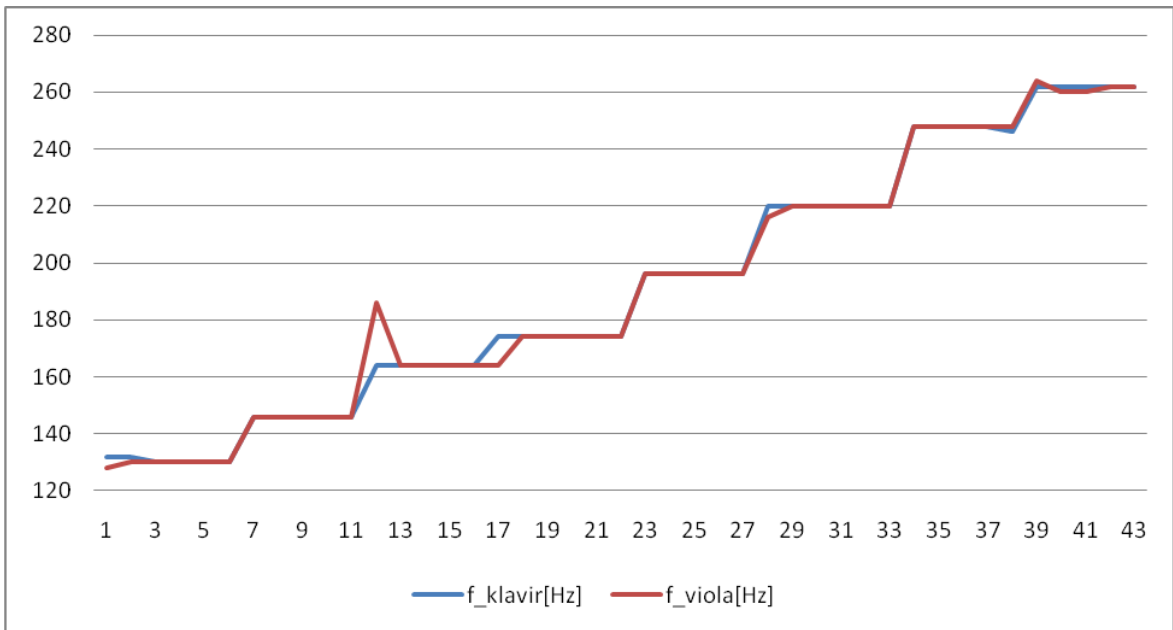


Obr. 17. Grafické zobrazení analyzovaných frekvencí MIDI violy

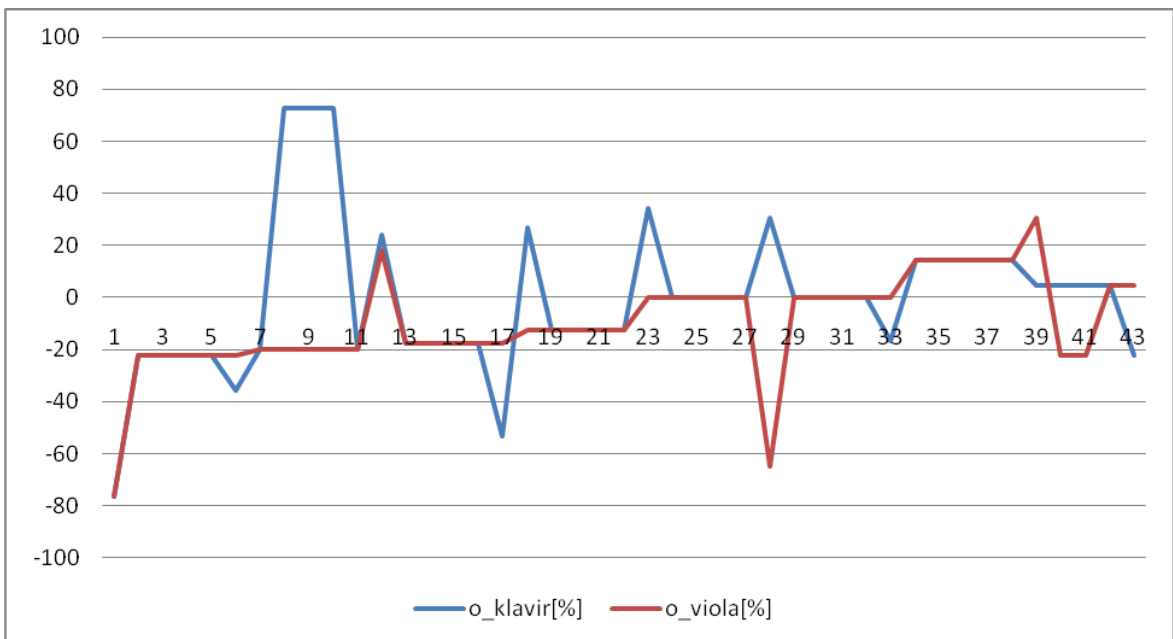


Obr. 18. Grafické zobrazení odchylek od nejbližších tónů MIDI violy

6.1.3 Srovnání analýz C-DUR MIDI nástrojů

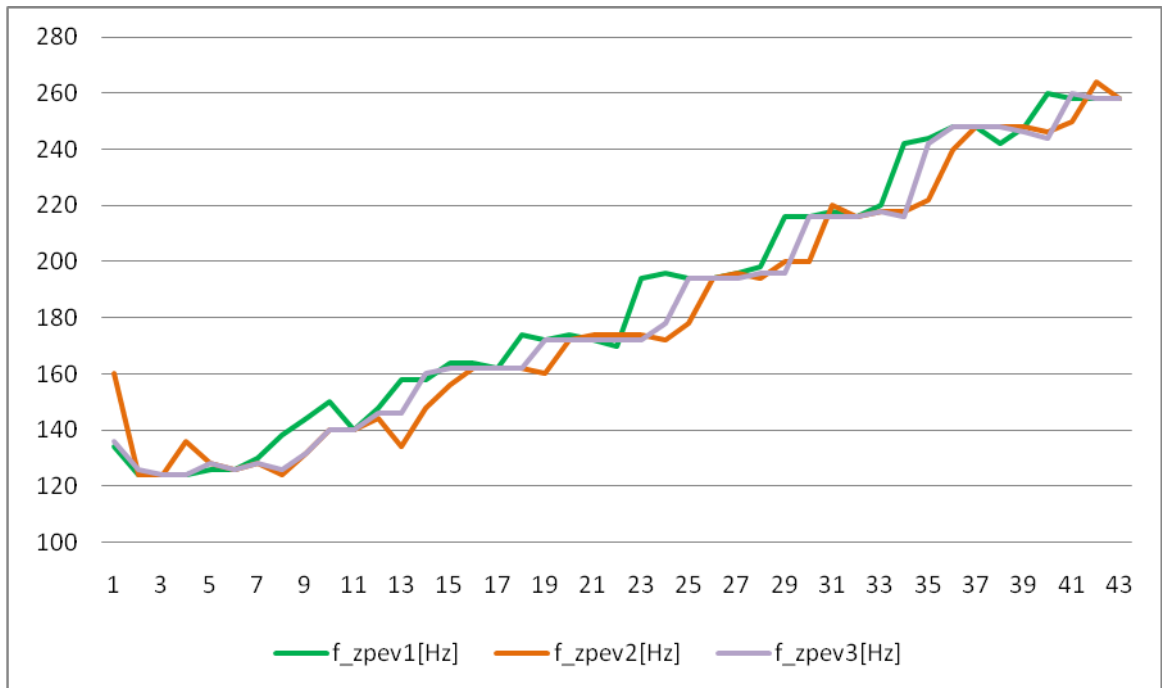


Obr. 19. Grafické srovnání analyzovaných frekvencí MIDI nástrojů

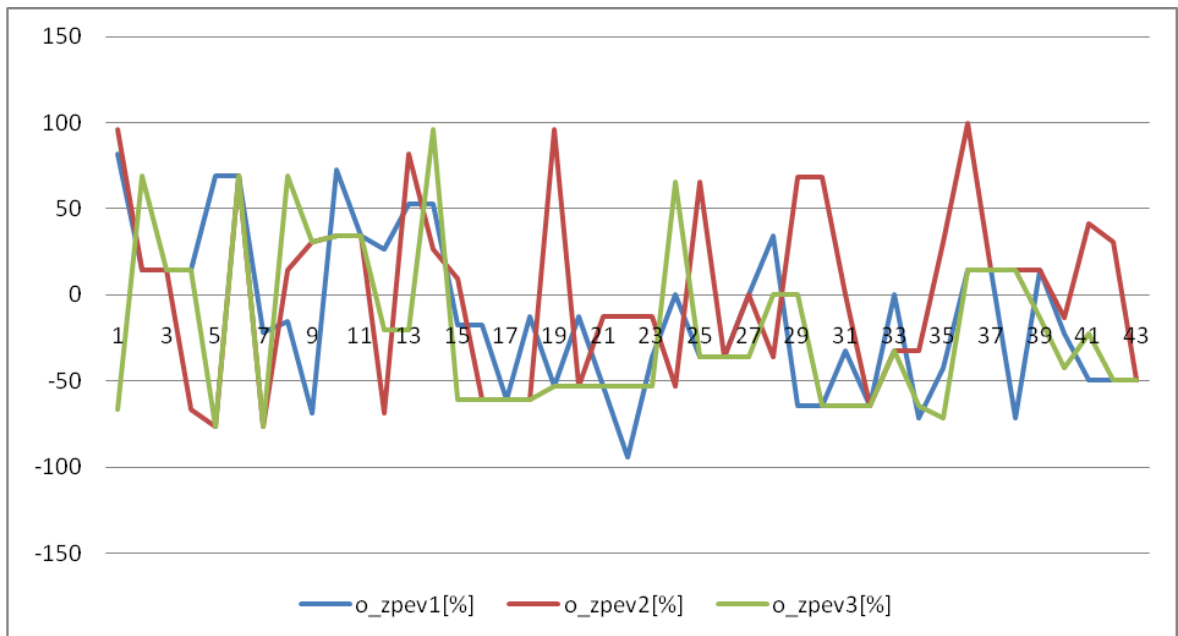


Obr. 20. Grafické srovnání odchylek od nejbližších tónů MIDI nástrojů

6.1.4 Analýza C-DUR zpěv

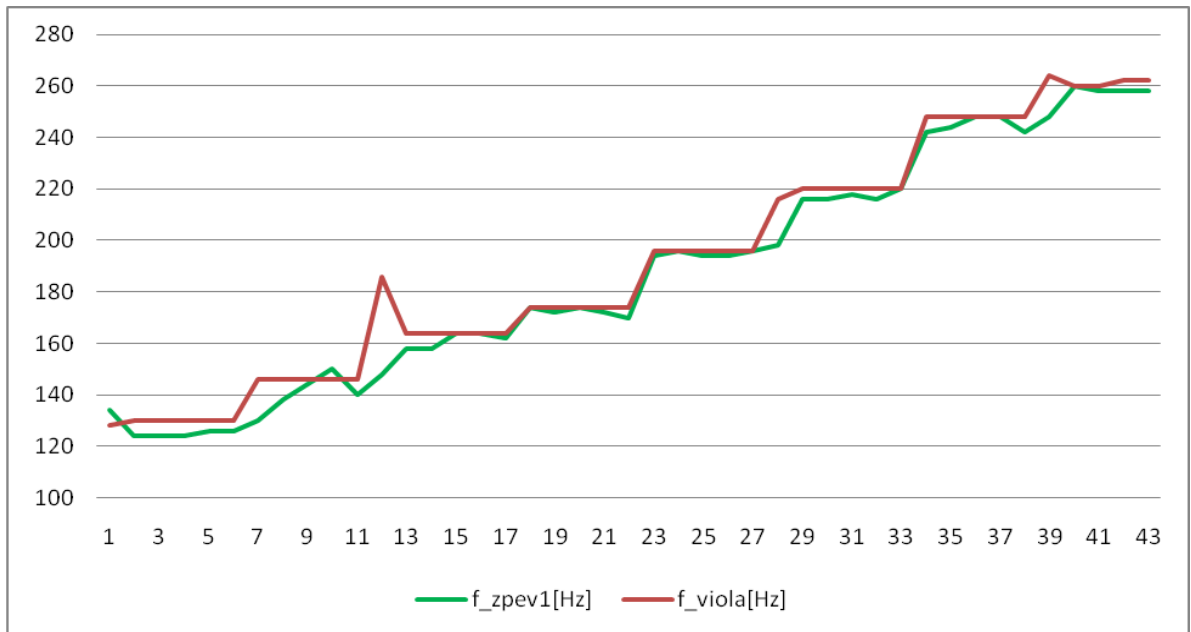


Obr. 21. Grafické zobrazení analyzovaných frekvencí 3 pokusy zpěvu

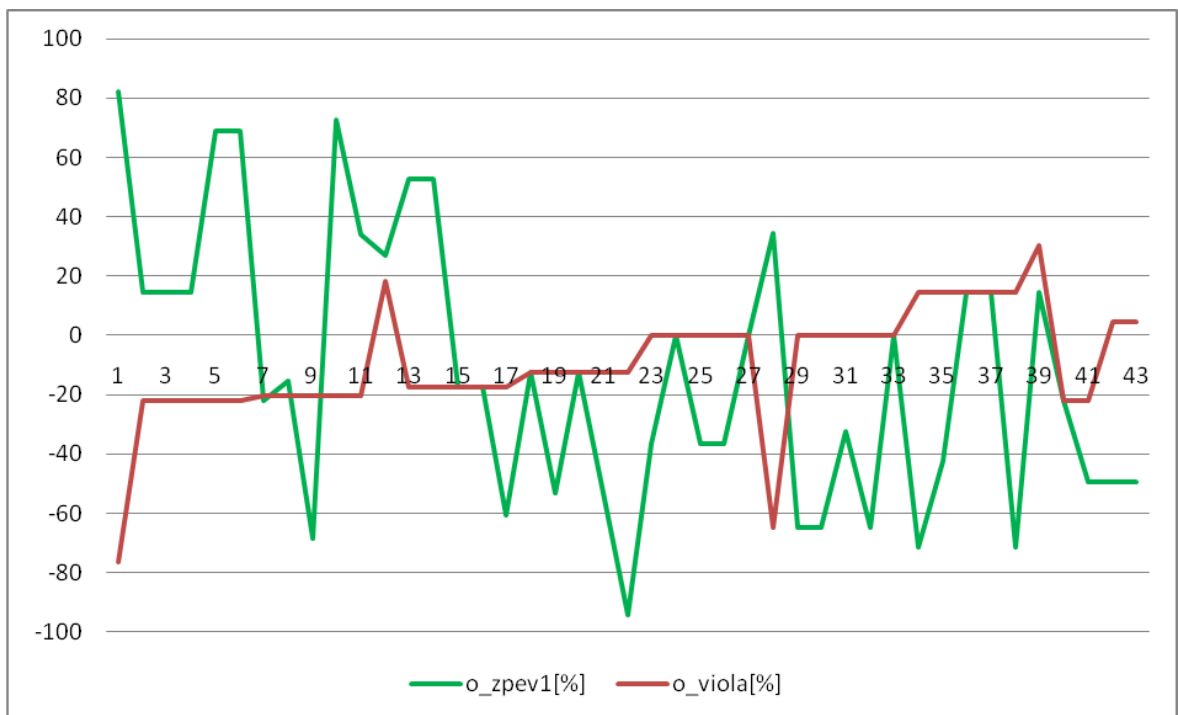


Obr. 22. Grafické srovnání odchylek od nejbližších tónů 3 pokusy zpěvu

6.1.5 Srovnání C-DUR MIDI viola a zpěv



Obr. 23. Grafické zobrazení srovnání MIDI violy a zpěvu



Obr. 24. Grafické srovnání odchylek od nejbližších tónů MIDI violy a zpěvu

6.2 Testování rytmiky

Testování proběhlo na počtu dob: 10 a tempu 120bpm.

Tab. 3. Výsledek rytmická analýza

i	t[s]
1	1,046875
2	1,03125
3	1
4	1
5	1,0625
6	1,0625
7	1
8	1,015625
9	1,046875
10	1,03125

7 UŽIVATELSKÁ PŘÍRUČKA

- **registrační klíč web:** „*utb*“
- **předdefinovaný účet:**
 - přihlašovací jméno: *utbl*
 - heslo: *utbfai*
- **stáhnutí spustitelného exe souboru:** <http://gold-voice.com/index.php?id=5>
- **stáhnutí zdrojových kódů:** <http://gold-voice.com/index.php?id=5>

7.1 Ovládání programu

7.1.1 Práce s notovým základem

Vkládání notového záznamu se provádí klikem na příslušnou klaviaturu. Ovšem před samotným klikem na vybraný tón v klaviatuře, je třeba vybrat druh noty (půlová, čtvrt'ová, osminová), který se provádí v nabídce v levé části layoutu. Implicitně je vybrána nota půlová. Tedy po těchto krocích se vybraná nota zobrazí v notové osnově vykreslovacího plátna. Přidání další noty lze posunutím kurzoru doprava šipkou doprava na klávesnici počítače. Zadaný notový záznam lze přehrát v záložce přehrání notového záznamu - tlačítkem pro přehrání.

Notový záznam se uloží za pomoci klávesové zkratky *Ctrl+S* nebo kliknutím na *menu > soubor > uložit*. V dialogovém oknu se vybere umístění souboru a jeho název a potvrdí. Po tomto kroku se soubor uloží.

Otevírání souboru se děje obdobným způsobem jako jeho ukládání, tedy buď klávesovou zkratkou *Ctrl+O* nebo pomocí *menu > soubor > otevřít*. Otevírání probíhá opět standartním způsobem Windows před dialogovým oknem.

7.1.2 Analýza souboru WAV

Analýzu externího souboru wav lze provést kliknutím na *menu > soubor > import wav* nebo klávesovou zkratkou *Ctrl+I*. V záložce detekce wav souboru se nyní načtou informace o souboru. Pak stačí kliknout na tlačítko analyzovat a data se analyzují a zobrazí v novém formulářovém okně. Data je možné exportovat do txt souboru.

7.1.3 Práce s rytmickými cvičeními

Vybráním záložky pro rytmická cvičení se zobrazí panel s možností nastavení počtu dob, pro to jak má celá analýza trvat. Dále je zde na výběr comboBox sloužící pro určení rytmiky tohoto cvičení. Jakmile dojde ke kliknutí na tlačítko start, odstartuje se blikání zeleného čtverce. V této chvíli uživatel začne klikat na neaktivní tlačítko pro klik, aby do sebe dostal tento rytmus a jakmile se blikání obarví na červeno časovač se spustí a analýza začne probíhat, po třech bliknutí červeného čtverce.

ZÁVĚR

Cílem této diplomové práce byl návrh a implementace software určený k analýze intonačních a rytmických dovedností uživatele. Dalším nemalým cílem bylo vytvořit dynamickou webovou prezentaci s možností personalizace uživatelů programu a sdílení jednotlivých cvičení.

V teoretické části byla rozebrána akustika, vznik kmitání či vlnění, fyziologická podstata zvuku a také metody zvukové analýzy. V druhé kapitole bylo vysvětleno několik základních pojmů z hudební teorie využitých hojně v praktické části. Teoretická část končí rozborem jednotlivých technologií potřebných k implementaci software.

V praktické části byly popsány nejdůležitější funkce programu a následně provedena analýza hudebních souborů wav, které jsou přiloženy v příloze diplomové práce.

Z výsledků testování programu na zvukových souborech je zřejmé, že aplikace analyzuje zvuková data správně. Nejprve bylo vytvořeno několik zvukových souborů z midi nástrojů, kde je jistota, že jednotlivé tóny jsou intonačně správné. Ovšem i tak výsledek programu vykazoval malé nepřesnosti v řádech jednotek centů. Tento nedostatek byl nejspíše zapříčiněn převodem z midi formátu do wav. Po analýze zvuků MIDI nástrojů, následovala analýza zvukových nahrávek lidského zpěvu. Zde již program vykazoval ze zvukových záznamů vyšší nepřesnosti. Ovšem tento fakt již byl dopředu zcela zřejmý, jelikož lidské hlasivky nejsou stroj. Tři nahrávky lidského zpěvu byly analyzovány na technickém cvičení stupnice C-DUR, kde jako intonačně nejčistší vyšla nahrávka cdur_zpev001.wav. Velmi zajímavé je srovnání analýzy zpěvu a MIDI nástroje, z níž lze vyčíst nedostatky testované osoby. Dále byla provedena analýza rytmického cítění.

Byla vyvinuta aplikace, která by mohla nalézt uplatnění nejen při odborné analýze zvukových záznamů, ale také i u širší veřejnosti zabývající se zpěvem. K tomu také slouží dynamická webová prezentace, kde si lidé mohou stáhnout od ostatních registrovaných uživatelů pěvecká cvičení a na nich zdokonalovat své hudební dovednosti.

Vývoj této aplikace bude probíhat i nadále. V další verzi si je kladeno za cíl spojit zvukovou analýzu s vyšší zábavností. To bude docíleno implementací karaoke a MIDI souborů a také ve zdokonalení intuitivnějšího ovládání i laické nehuděbní veřejnosti.

CONCLUSION

The main aim of this diploma thesis is to suggest and implement a software made for analysis of the intonational and rhythmical abilities of its users. Second important aim, is to create a dynamical web presentation with the possibility of personalisation for every single user and to share single exercises of this software.

Theoretical part is particularly about the acoustics, occurrence of an oscillation and wave motion, physiological matter of sound and about the sound-analysing methods as well. In second chapter, a number of basic musical theory concepts used in practical part is explained. Theoretical part ends with an analysis of single technologies needed for a software implementation.

Analytical part shows the most important program's functions and analysis of audio wav files, included among the appendixes of this diploma thesis.

The results of testing procedure with audio files show the right and correct application's analysis of the audio data. First step was a creation of several audio files by midi tools, because in this case we can be sure the single sounds are absolutely correct from the intonational point of view. Although this fact, the programme has shown some irregularities, but in a very inconsiderable way, which is most probably caused by conversion from the midi into the wav format. After the analysis the midi sounds, the analysis of the human singing followed and during this step the programme already shows some bigger irregularities. This fact is quite clear and it's also simple to predicate it because of the reason that human voices are not machines at all. Three records of human singing were analyzed during a technical exercise based on the C-DUR scale; from which the clearest record was created – "cdur_zpev001.wav". Comparing the human singing to the appropriated midi sound is very interesting, because we can find embarrassments of a tested person here. Another step is the analysis of the rhythmical feeling.

This application may be helpful not for only making analysis of audio records, but generally for many people interested in singing as well. In this case, the dynamic web presentation may be also very useful, because it's possible to download exercises from other users, which could be really helpful in trying to make the musical skills better.

Developing of this application continues. One of the main aims for next version is to make analyses more enjoyable and get it closer to uninitiated non-musical community. This

is going to be reached by implementing a karaoke, midi files and also by making the operating more intuitive and easy.

SEZNAM POUŽITÉ LITERATURY

- [1] NAGEL, Christian, et al. C 2008 : Programujeme profesionálně. David Dirga. 1. vydání. Brno : Computer Press, a.s., 2009. 1898 s. ISBN 978-80-251-2401-7.
- [2] MCCONNELL, Steve. Dokonalý Kód : Umění programování a techniky tvorby software. Bogdan Kiszka. Brno : Computer Press, a.s., 2006. 894 s. ISBN 80-251-0849-X.
- [3] PETZOLD, Charles. Mistrovství ve Windows Presentation Foundation. Jakub Mikulaščík, Jiří Fadrný. Brno : Computer Press, a.s., 2008. 928 s. ISBN 978-80-251-2141-2.
- [4] GUTMANS, Andi; BAKKEN, Saether Stig; RETHANS, Derick. Mistrovství v PHP 5. Ivo Magera; Bogdan Kiszka. 2. vydání. Brno : Computer Press, a.s., 2007. 655 s. ISBN 978-80-251-1519-0.
- [5] KOFLER, Michael. Mistrovství v MySQL 5 : Kompletní průvodce webového vývojáře. Martin Domes; Jan Svoboda, Ondřej Baše, Jaroslav Černý. 1. vydání. Brno : Computer Press, a.s., 2007. 805 s. ISBN 978-80-251-1502-2.
- [6] WILLIAMS, E. Hugh; LANE, David. Programujeme webové aplikace pomocí PHP a MySQL. Ivo Magera; David Krásenský. 1. vydání. Praha : Computer Press, a.s., 2002. 530 s. ISBN 80-7226-760-4.
- [7] CROFT, Jeff; LLOYD, Ian; RUBIN, Dan. Mistrovství v CSS : Pokročilé techniky pro webové designéry a vývojáře. Martin Domes; Josef Bábík. 1. vydání. Brno : Computer Press, a.s., 2007. 409 s. ISBN 978-80-251-1705-7.
- [8] SYROVÝ, Václav. Hudební akustika. Vyd. 2. 2008 : Akademie múzických umění v Praze, 2008. 440 s. ISBN 978-80-7331-127-8.
- [9] MAJTÁN, Petr. Web implementace strategické hry. Zlín, 2009. 45 s. Bakalářská práce. Univerzita Tomáše Bati ve Zlíně.
- [10] MAJTÁN, Petr. Bezpečnost dynamické webové prezentace. Zlín, 2010. 9 s. Seminární práce. Univerzita Tomáše Bati ve Zlíně.

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

- FFT Rychlá fourierová transformace.
- C# Programovací jazyk Csharp.
- MySQL Databázový systém s komunikací probíhající pomocí jazyka SQL.
- PHP Skriptovací jazyk určený zejména k programování dynamických webových stránek
- GVO Souborový typ gold-voice
- C-DUR Stupnice začínající od tónu C

SEZNAM OBRÁZKŮ

<i>Obr. 1. Odvození kmitavého harmonického pohybu.....</i>	<i>11</i>
<i>Obr. 2. Sinusoida s vyznačenými parametry.....</i>	<i>13</i>
<i>Obr. 3. Tlumené kmity s vyznačenými parametry</i>	<i>14</i>
<i>Obr. 4. Skládání fázově posunutých kmitů.....</i>	<i>15</i>
<i>Obr. 5. Stavba lidského ucha [8].....</i>	<i>18</i>
<i>Obr. 6. Logaritmické vnímání výšky tónu [8]</i>	<i>19</i>
<i>Obr. 7. Náhled programu Gold-voice.....</i>	<i>28</i>
<i>Obr. 8. Rozložení oblastí ve formuláři 1.....</i>	<i>30</i>
<i>Obr. 9. Rozložení oblastí formulář 2</i>	<i>30</i>
<i>Obr. 10. Zobrazovací displej a klaviatura.....</i>	<i>31</i>
<i>Obr. 11. Zobrazení dat na vykreslovacím plátně.....</i>	<i>32</i>
<i>Obr. 12. Schéma volání metod při ukládání souboru GVO.....</i>	<i>34</i>
<i>Obr. 13. Výsledek analýzy WAV souboru</i>	<i>39</i>
<i>Obr. 14. Náhled webové prezentace</i>	<i>41</i>
<i>Obr. 15. Grafické zobrazení analyzovaných frekvencí MIDI klavíru.....</i>	<i>46</i>
<i>Obr. 16. Grafické zobrazení odchylek od nejbližších tónů MIDI klavíru.....</i>	<i>46</i>
<i>Obr. 17. Grafické zobrazení analyzovaných frekvencí MIDI violy.....</i>	<i>47</i>
<i>Obr. 18. Grafické zobrazení odchylek od nejbližších tónů MIDI violy</i>	<i>47</i>
<i>Obr. 19. Grafické srovnání analyzovaných frekvencí MIDI nástrojů.....</i>	<i>48</i>
<i>Obr. 20. Grafické srovnání odchylek od nejbližších tónů MIDI nástrojů</i>	<i>48</i>
<i>Obr. 21. Grafické zobrazení analyzovaných frekvencí 3 pokusy zpěvu.....</i>	<i>49</i>
<i>Obr. 22. Grafické srovnání odchylek od nejbližších tónů 3 pokusy zpěvu</i>	<i>49</i>
<i>Obr. 23. Grafické zobrazení srovnání MIDI violy a zpěvu.....</i>	<i>50</i>
<i>Obr. 24. Grafické srovnání odchylek od nejbližších tónů MIDI violy a zpěvu.....</i>	<i>50</i>

SEZNAM TABULEK

<i>Tab. 1. Struktura souboru GVO.....</i>	<i>33</i>
<i>Tab. 2. Struktura tabulky gvo_registration.....</i>	<i>42</i>
<i>Tab. 3. Výsledek rytmičká analýza</i>	<i>50</i>

SEZNAM PŘÍLOH

CD-ROM – Obsahuje text diplomové práce, zdrojové kódy k praktické části, spustitelný exe soubor a všechny testované soubory, které byly potřebné v analýze těchto dat.

PŘÍLOHA P I: CD-ROM