VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

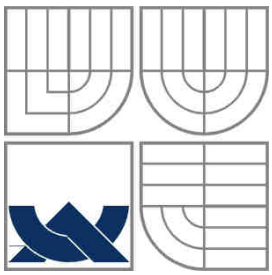
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MULTIMEDIÁLNÍ SYSTÉM PRO PODPORU VÝUKY JAZYKŮ

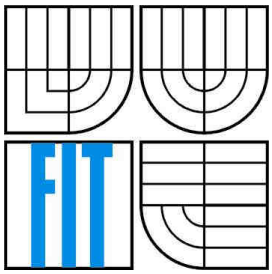
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE LIBOR DRÁPAL
AUTHOR

BRNO 2007



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

MULTIMEDIÁLNÍ SYSTÉM PRO PODPORU VÝUKY JAZYKŮ

MULTIMEDIA SYSTEM FOR SUPPORT OF LANGUAGES EDUCATION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — LIBOR DRÁPAL
AUTHOR

VEDOUCÍ PRÁCE — RNDr. JITKA KRESLÍKOVÁ, CSc.
SUPERVISOR

BRNO 2007

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2006/2007

Zadání diplomové práce

Řešitel: **Drápal Libor**
Obor: Výpočetní technika a informatika
Téma: **Multimediální systém pro podporu výuky jazyků**
Kategorie: Softwarové inženýrství

Pokyny:

1. Seznamte se s problematikou elektronického vzdělávání (e-learning). Zaměřte se na možnosti uplatnění 3D grafiky a využití audio záznamů.
2. Seznamte se s aplikačním rozhraním operačního systému Microsoft Windows.
3. Seznamte se s nástroji pro zobrazení 3D grafiky a pro přehrávání audio záznamů v operačním systému Microsoft Windows.
4. Navažte na ročníkový projekt Systém pro podporu výuky jazyků a rozšiřte specifikaci požadavků o možnosti zobrazování 3D objektů a přehrávání zvukových záznamů. Proveďte analýzu systému. Použijte vhodné modelovací nástroje.
5. Navrhněte systém pro podporu multimediální výuky jazyků.
6. Realizujte funkční prototyp navrženého systému.
7. Použitelnost systému demonstrujte na vhodně zvoleném vzorku dat.
8. Zhodnoťte dosažené výsledky a možnosti dalšího pokračování.

Literatura:

- Horton, W.: Design Web-Based Training: How to Teach Anyone Anything Anywhere Anytime, Wiley; 1 edition, ISBN: 047135614X
- Horton, W.: E-learning Tools and Technologies. John Wiley & Sons; 1st edition (January 10, 2003), ISBN: 0471444588.
- http://www.fmod.org/ - knihovna FMOD pro přehrávání audio obsahu
- http://msdn.microsoft.com/ - nápověda pro WinAPI
- http://nehe.opengl.com/ - programování 3D grafiky pod knihovnou OpenGL

Při obhajobě semestrální části diplomového projektu je požadováno:

- Bez požadavků.

Podrobné závazné pokyny pro vypracování diplomové práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva diplomové práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap, které byly vyřešeny v rámci ročníkového a semestrálního projektu (30 až 40% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním paměťovém médiu (disketa, CD-ROM), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Kreslíková Jitka, RNDr., CSc.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2006
Datum odevzdání: 22. května 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav informačních systémů
612 66 Brno, Božetěchova 2

doc. Ing. Jaroslav Zendulka, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Libor Drápal**
Id studenta: 21732
Bytem: Kotlanova 3A, 628 00 Brno
Narozen: 21. 02. 1982, Brno
(dále jen "autor")

a

2. Vysoké učení technické v Brně

Fakulta informačních technologií
se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....
(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
diplomová práce

Název VŠKP: Multimediální systém pro podporu výuky jazyků
Vedoucí/školitel VŠKP: Kreslíková Jitka, RNDr., CSc.
Ústav: Ústav informačních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

tištěné formě počet exemplářů: 1
elektronické formě počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
   - ☒ ihned po uzavření této smlouvy
   - ☐ 1 rok po uzavření této smlouvy
   - ☐ 3 roky po uzavření této smlouvy
   - ☐ 5 let po uzavření této smlouvy
   - ☐ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: .....

Drápal

....

Nabyvatel Autor

Abstrakt

Projekt se zabývá problematikou elektronického vzdělávání (e-learning). Je v něm vysvětleno, co tento pojem znamená, kde a jakým způsobem se tento druh vzdělávání využívá a jaké jsou jeho výhody a nevýhody ve srovnání s jinými výukovými metodami. Jsou zde uvedeny nástroje pro tvorbu elektronického kurzu a je nastíněno jak by měl tento kurz vypadat. Dále jsou probrány různé techniky a nástroje pro vytváření aplikačního rozhraní, zobrazování 3D grafiky a přehrávání audio záznamů pod různými operačními systémy. V tomto směru jsou pak podrobněji popsány nástroje pro operační systém Microsoft Windows. Cílem bylo navrhnout a realizovat multimediální systém pro podporu výuky jazyků. Využitím 3D grafiky a zvukových záznamů byl vytvořen výukový systém, který je pro uživatele lákavý vzhledem a pohodlný při jeho ovládání.

Klíčová slova

elektronické vzdělávání, elektronický kurz, jazyk (+angličtina, …), slovník, lekce, počítač, operační systém, grafika, zvukový záznam.

Abstract

This project is about electronic learning (e-learning). The first part of this document explains the meaning of the term e-learning, where it is used and what advantages and disadvantages has this type of learning compared to the other methods of learning. The user gets familiar with tools that create the electronic lesson and how to design the lesson. In the second part there are described different techniques and tools to create application interface, to display 3D graphic and to play audio records in different operating systems. In this part, there are described tools in more detail for the operating system Microsoft Windows. The goal of this project is to design and to realize the multimedia system for language education. With the use of 3D graphic and audio records the educational system was created with sophisticated appearance and comfortable control.

Keywords

electronic learning, electronic lesson, language (+English, …), dictionary, lesson, computer, operating system, graphic, audio record.

Citace

Libor Drápal: Multimediální systém pro podporu výuky jazyků, diplomová práce, Brno, FIT VUT v Brně, 2007

Multimediální systém pro podporu výuky jazyků

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením RNDr. Jitky Kreslíkové, CSc.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Libor Drápal
2007

Poděkování

Chtěl bych poděkovat vedoucí mé diplomové práce RNDr. Jitce Kreslíkové CSc. za vedení práce, zapůjčenou literaturu a poskytnuté konzultace.

Obsah

Úvod

První kapitola tohoto dokumentu se zabývá problematikou elektronického vzdělávání (e-learning). Je zaměřena na možnosti uplatnění 3D grafiky a využití audio záznamů. Druhá kapitola je zaměřena na různé operační systémy a jejich aplikační rozhraní. Podkapitola je věnována aplikačnímu rozhraní operačního systému Microsoft Windows. Ve třetí kapitole jsou představeny nástroje pro zobrazení 3D grafiky a pro přehrání audio záznamů. Čtvrtá kapitola navazuje na ročníkový a semestrální projekt – Systém pro podporu výuky jazyků a rozšiřuje specifikaci požadavků o možnosti zobrazování 3D objektů a přehrávání zvukových záznamů. Rozšiřuje tedy předchozí výukový systém o multimediální prvky. Jsou zde také uvedeny vhodné modelovací nástroje. V páté kapitole je návrh multimediálního výukového systému. Jsou zde shrnuty všechny poznatky a požadavky kladené na tvorbu elektronického výukového systému. Šestá kapitola se zabývá realizací prototypu. Jsou zde uvedeny použité modelovací nástroje a je nastíněna základní struktura programu na úrovni zdrojových kódů. Sedmá kapitola je určená pro uživatele aplikace. Je zde uvedeno, co vše aplikace obsahuje a jak se ovládá. V poslední kapitole je shrnutí všech poznatků a zhodnocení výsledků práce.

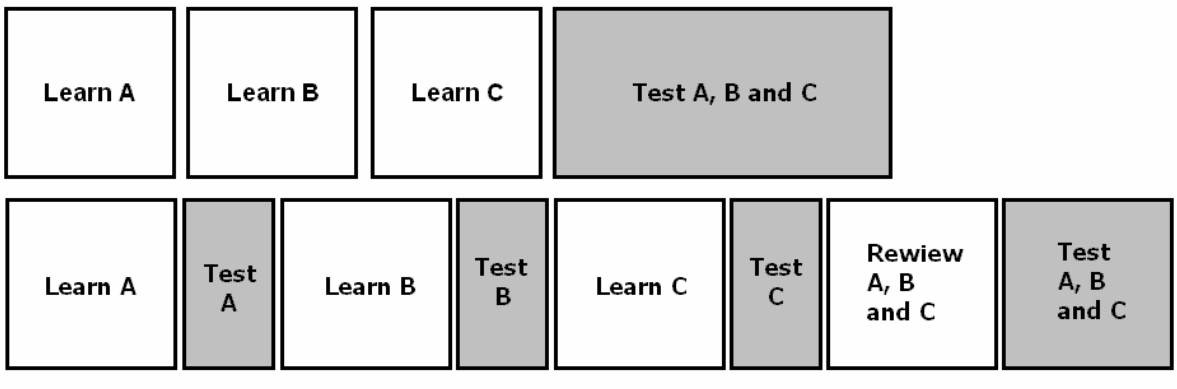
1 Co je to elektronické vzdělávání

E-learning neboli elektronické vzdělávání je proces, zabývající se poskytováním vzdělávacích a výcvikových aktivit a sdílením informací v elektronické podobě. Je určen jak pro skupiny uživatelů (výukové kurzy pro studenty, kurzy pro zaškolování zaměstnanců), tak pro jednotlivce (např. distanční vzdělávání pro samouky a zájemce, kteří mají vlastní zájem zlepšit své znalosti a dovednosti).

Klíčové slovo „elektronické“ napovídá, že e-learning je určen pro zařízení schopné prezentovat digitální obsah (počítače, mobilní telefony, televize, atd.). E - learningová aplikace umožňuje uživateli naučit se a procvičit si nové znalosti z daného oboru. Sestává ze dvou hlavních částí. První částí je obsah, tedy informace uložená v digitální podobě (texty, obrázky, animace, zvuky atd.). Druhou částí jsou metody (techniky), které pomáhají studentům pochopit a naučit se tento obsah. Metody, jsou tedy způsoby, jak co nejefektivněji prezentovat obsah e-learningového kurzu, aby byl co nejlépe pochopen a naučen.

Podle průzkumů (Clark, 1994; Dillon and Gabbard, 1998), kde byly srovnány různé výukové prostředky (video, text, učitel), se došlo k závěrům, že je důležité, jaké jsou použity výukové metody. Nelze prohlásit, že výuka pomocí počítače je lepší prostředek, než knižní podoba informace nebo kurz vedený lektorem. Je to pouze další alternativa, jak získávat informace. Počítač má unikátní vlastnosti, jako je přenos animace, videa, zvuku atd.; sdílení informací, komunikace a spolupráce mezi studenty a učiteli, které ostatní média nemohou nabídnout. Oproti tomu nemůže nabídnout přímý kontakt žáka s učitelem, který může být někdy žádoucí. Každý výukový prostředek má tedy své klady a zápory a nelze rozhodnout, který je lepší. Důležité je, jak budou tyto vlastnosti využity, aby poskytly informaci co nejefektivněji.

Každý elektronický kurz se sestává z několika lekcí. Obecně je v každé lekci seznámení s problematikou a ukázka příkladů, otestování právě naučené látky a zobrazení výsledků testů např. v podobě grafu. Obtížnost látky by měla postupně narůstat s další lekcí. Informační a testové části lekce, by se měli střídat v kratších intervalech, aby si uživatel novou problematiku hned odzkoušel viz obr. 1.1. Textová informace by měla být doplněna obrazem, videem a zvukem (zde je výhoda elektronického kurzu oproti jiným výukovým prostředkům). Vizuální vzhled a ovládací prvky elektronického kurzu by měli být navrženy co nejpřehledněji, aby uživatel nestrávil svůj čas nad ovládnutím rozhraní kurzu, ale nad kurzem samotným. Více podrobností jak navrhnout kvalitní elektronický kurz viz [1].



obr. 1.1 Střídání informačních a testových částí lekce

1.1 Historie

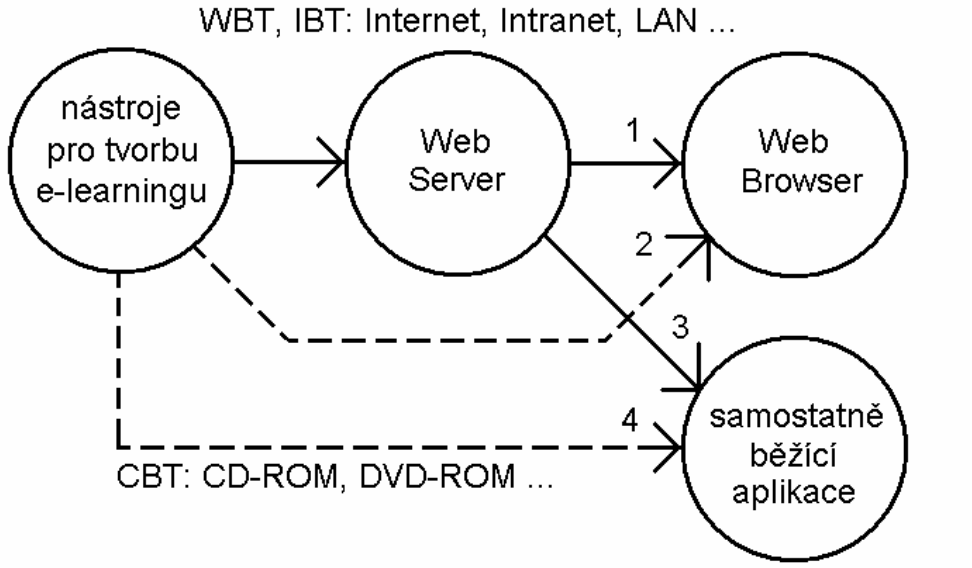
S pojmem elektronické vzdělávání jsme se mohli setkat již na konci 60. let 20. století, kdy se výpočetní technika začala používat ve vzdělávání. První výukové programy byly však pouze textové. Hlavní aplikace běžela na velkém sálovém počítači zvaném mainframe. Ke každé otázce bylo přiřazeno více odpovědí, z kterých si uživatel vybíral. Jeho odpověď pak byla porovnána s odpovědí správnou. Díky vývoji výpočetních technologií jsme znatelné rozšíření a rozvoj e-learningu mohli zaznamenat až v současnosti. Dnes jsou součástí elektronického kurzu i různé multimediální komponenty jako grafika, video, animace nebo audio, které zlepšují pochopení obsahu kurzu.

1.2 Poskytování e-learningu

Možností, jak dopravit elektronický kurz od autora k uživateli je několik. Jsou to metody určené pro přenos digitálních dat. Mezi nejčastěji používané patří: internet, intranet, CD-ROM/DVD-ROM, ale také televizní nebo mobilní přenosy, či video kazety. Každá z těchto metod má své výhody i nevýhody a pro každou aplikaci se hodí jiná metoda. V dnešní době je však výpočetní technika a internet na takové úrovni (zvyšování rychlosti připojení, zvětšování kapacity médií atd.), že nejefektivnější metoda šíření je pomocí internetu a přenosných médií, jako je CD-ROM nebo DVD-ROM (popřípadě jejich kombinací).

Podle průzkumu (Galvin, 2001) šíření e-learningu: 40% CD-ROM, 22% internet, 30% intranet. Tedy více jak 90% e-learningu je šířeno jako CBT nebo WBT (Computer-Based Training = výuka pomocí počítačů; Web-Based Training = výuka pomocí počítače připojeného do sítě, IBT = Internet-Based Training = k výuce výhradně využíván internet).

Výhody šíření informací pomocí internetu (zvaném Web Based Training WBT nebo Internet Based Training IBT) jsou snadná aktualizace obsahu e-kurzu, snadná komunikace a sdílení informací mezi žáky a tutorem, buď přímo skrze elektronický systém nebo pomocí emailu, icq atd., ale především snadná distribuce. Ne však každý uživatel má možnost připojení k internetu nebo jeho připojení nedosahuje takové rychlosti, aby mohl elektronický kurz efektivně prohlížet. V takovém případě si autor kurzu zvolí cestu tzv. Computer Based Training CBT, kde se elektronický kurz přenáší pomocí CD-ROM nebo DVD-ROM nosičů. Díky velké kapacitě média, může kurz obsahovat video nebo audio záznamy, které by se po internetu nedaly rychle přenést. Obrázek 1.2 zobrazuje distribuci elektronického kurzu.



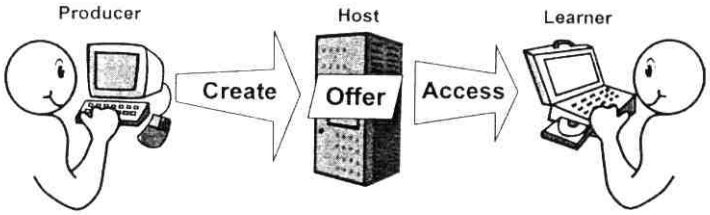
obr. 1.2 Distribuce e-learningu

1 e-learning je poskytován přes Internet, Intranet, LAN … (nutnost připojení)
2 dodání objemných dat na CD-ROM (nutnost připojení zůstává, ale objem stažených dat je výrazně menší)
3 připojení k internetu za účelem stáhnutí aktualizace aplikace dodané na CD-ROM
4 dodání aplikace na CD-ROM, DVD-ROM (vše je na dodaném médiu, není třeba připojení)

1.2.1 Hardwarové a softwarové požadavky

Zajišťování potřebného vybavení (software + hardware) na straně autora kurzu i na straně uživatele je ovlivněno způsobem, jakým je e-learning šířen. Proces elektronického vzdělávání se dělí na tři etapy viz obr. 1.3: etapa vývoje elektronického kurzu, etapa sdílení kurzu (poskytnutí kurzu ze

strany autora) a etapa zpřístupnění kurzu (prohlížení ze strany žáka – uživatele). V každé etapě se setkáme s jinou skupinou softwarového a hardwarového vybavení.



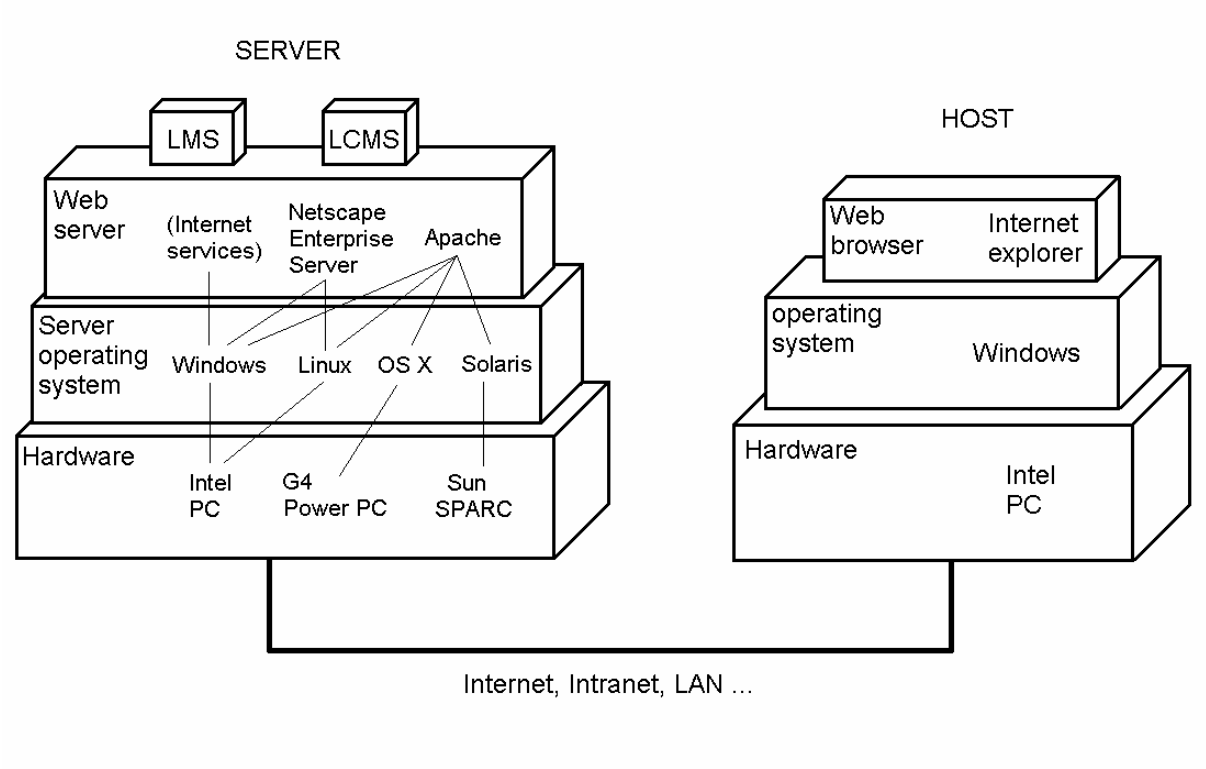
obr. 1.3 etapa vývoje, sdílení a prohlížení elektronického kurzu

Hardwarové a softwarové požadavky na straně uživatele kurzu nejsou tak náročné jako u poskytovatele. Uživateli obvykle stačí standardní počítačová sestava připojená k internetu s nainstalovaným operačním systémem a webovým prohlížečem (záleží na typu poskytování vzdělávacího systému; v jistých případech, není připojení k internetu ani webového prohlížeče zapotřebí).

Na straně autora a poskytovatele kurzu není situace už tak jednoduchá. Autor má dvě možnosti. Buď si vybere metodu šíření po internetu, a potom bude muset jeho poskytovatel umístit elektronický kurz na server, nebo se rozhodne pro off-line řešení a jeho poskytovatel – distributor vydává jeho produkt na nosičích CD-ROM, popřípadě DVD-ROM. Jistou třetí variantou je i kombinace těchto dvou zmíněných možností.

V případě, že je elektronický kurz šířen pomocí internetu, používají se tzv. LCMS - Learning Content Management Systémy viz obr. 1.4. Seznam a popis některých významných LCMS systémů viz [2]. Tyto Systémy zajišťují především správu obsahu kurzu. Umožňují správci nebo autorovi řídit vytváření, úpravy a odstraňování obsahu z webu, registraci studentů, sledování jejich studia, výsledků testů a oznamují ukončení kurzů. Autor kurzu má možnost ověřovat znalosti studentů. Tato varianta však vyžaduje vyšší hardwarové nároky na straně poskytovatele kurzu.

Autor se však může rozhodnout pro "off-line" verzi šíření, kde k vytvoření aplikace může použít různé programovací jazyky nebo nástroje určené pro tvorbu elektronických kurzů. Obvykle se k šíření takto vytvořeného softwaru používají CD-ROM nebo DVD-ROM nosiče (odpadají sice vysoké hardwarové nároky na straně poskytovatele, ale jsou zde zase náklady na straně distributora softwaru).



obr. 1.4 LMS a LCMS systém

LMS a LCMS systém a jednotlivé hardwarové vrstvy: poskytovatel elektronického kurzu – Server. Na straně uživatele – HOST je internetový prohlížeč (web browser), který zajišťuje přístup k e-learningu.

1.3 Tvorba kurzu

Než začneme vytvářet elektronický kurz, musíme si ujasnit dvě zásadní věci. Za prvé, jakým způsobem budeme chtít náš kurz šířit. Od toho se odvíjí naše softwarové i hardwarové vybavení. Za druhé, pro koho bude náš kurz určen, což ovlivňuje obsah kurzu. Pro tvorbu elektronických kurzů existuje velké množství nástrojů (softwarového vybavení). První skupinu tvoří nástroje pro tvorbu elektronického (digitálního) obsahu. Patří sem různé kreslící a animační nástroje, nástroje pro editaci videa a zvuku, textové editory atd. Další skupinou jsou nástroje pro tvorbu systémů nebo aplikací, které tento vytvořený obsah prezentují v ucelené formě vhodné pro výuku. Jsou to různé nástroje pro tvorbu webů, editory pro práci s HTML, nebo vývojové programovací studia (např. Builder C++, Delphy, Microsoft Visual Studio).

1.4 Využití 3D grafiky a audio záznamů

Kvalita programů se dříve hodnotila pouze podle funkčnosti a jisté kvality zpracování (rychlost programu, paměťové nároky atd.). Nehledělo se tolik na kvalitu grafiky nebo přítomnost multimediálního obsahu, jelikož počítače nebyly dostatečně hardwarově vybaveny, aby mohly tyto prvky obsahovat. I dnes je samozřejmě funkčnost aplikace na prvním místě, ale jelikož nároky uživatelů stále rostou jsou výrobci softwaru nuceni stále přicházet s novinkami, kterými by zaujali své zákazníky. Jelikož v posledních letech rozvoj hardwaru dosáhl velice slušné úrovně, není již žádný problém vytvářet multimediální programy se složitou grafikou. Většina výrobců toho plně využívá a předělává své dosavadní projekty z 2D prostředí na 3D. 3D grafika může aplikaci přidat nejen na přehlednosti, ale i vizuální atraktivitě. Využitím moderních technologií počítačové grafiky se programy posouvají na vyšší technickou úroveň. Společně s grafikou se na přehlednosti aplikace podílí i ostatní multimédia. Různé animace nebo doprovodný audio záznam (použitý jako zvuková nápověda), muže uživatele navigovat programem a ušetřit mu tak práci a čas, který by strávil studováním textových nápověd nebo manuálů.

Cílem je tedy využít těchto výhod a vytvořit takový systém, který by byl pro uživatele lákavý jak vzhledem, tak pohodlný při jeho ovládání. Jedině pak se může plně věnovat jeho obsahu.

2 Aplikační rozhraní operačních systémů

V této kapitole se seznámíme s různými typy operačních systémů a dozvíme se k čemu slouží jejich aplikační rozhraní. Budou nás zajímat operační systémy zařízení, která jsou nejvíce vhodná pro šíření e - learningu – tedy stolní počítače a notebooky. Existují samozřejmě i další typy zařízení, která se používají pro šíření e - learningu – PDA zařízení nebo mobilní telefony, a proto jsou pro úplnost uvedena i tato zařízení. Jejich aplikačními rozhraními se zabývat nebudeme, protože se mohou značně lišit od běžných systémů, jelikož jsou navrhnuta pro konkrétní řadu zařízení.

Operační systém je základní program, který se načte a spustí po startu počítače nebo elektronického zařízeni, které ho potřebuje ke své činnosti. Slouží ke správě všech hardwarových komponent a umožňuje instalaci a spouštění dalších aplikací. Existuje velké množství operačních systémů. Mezi nejznámější a nejpoužívanější patří: Microsoft Windows 95/98/2000/XP, Unix/Linux/FreeBSD, OS/2 a Apple Mac-OS. Tyto vyjmenované operační systémy jsou určeny převážně pro stolní počítače, notebooky nebo servery. Existují však i varianty operačních systému navržené pro mobilní telefony, PDA zařízení (personal digital assistant), a další přenosná elektronická zařízení. Mezi nejznámější operační systémy pro telefony patří Symbian OS s40/s60/s90/UIQ, SavaJe a Windows Mobile. Každý operační systém se liší svoji velikostí, funkčností, typem zabezpečení a dalšími vlastnostmi, které určuje typ zařízení, pro které je operační systém určen.

Aplikační rozhraní každého operačního systému obsahuje množinu příkazů, kterou může uživatel, programátor nebo systém samotný zasahovat do funkčnosti celého systému. Těmito příkazy je možno zasahovat jak do hardwarové vrstvy systému – konfigurovat hardwarové komponenty, měnit jejich vlastnosti a nastavení (např. změna rozlišení a barevné hloubky zobrazovacího adaptéru), tak ovládat systém na softwarové úrovni – vytvářet a vykreslovat okna aplikací, měnit registry nebo třeba měnit barvu textu v textové konzoli.

Jak bylo zmíněno, do systému mohou zasahovat tři entity – uživatel, programátor a systém samotný. Uživatel může zasahovat do systému buď přes terminál (textová konzole) nebo přes grafické rozhraní systému. Jeho pravomoc je definována množinou příkazů terminálu (popřípadě množinou skriptů, které si muže vytvořit a přes terminál spouštět). Grafické rozhraní má za úkol zjednodušit a zpřehlednit ovládání systému. V žádném případě však nepřidává na funkčnosti oproti textové konzoli, ale naopak existují operace, které přes něj nelze uskutečnit nebo jsou oproti terminálu zdlouhavé (jakákoliv akce v grafickém prostředí způsobí volání, které je ekvivalentní konzolovému příkazu). Uživatel pro ovládaní OS používá různé systémové příkazy. Programátor přistupuje k ovládání systému obdobným způsobem, ale může zasahovat do větší části systému.

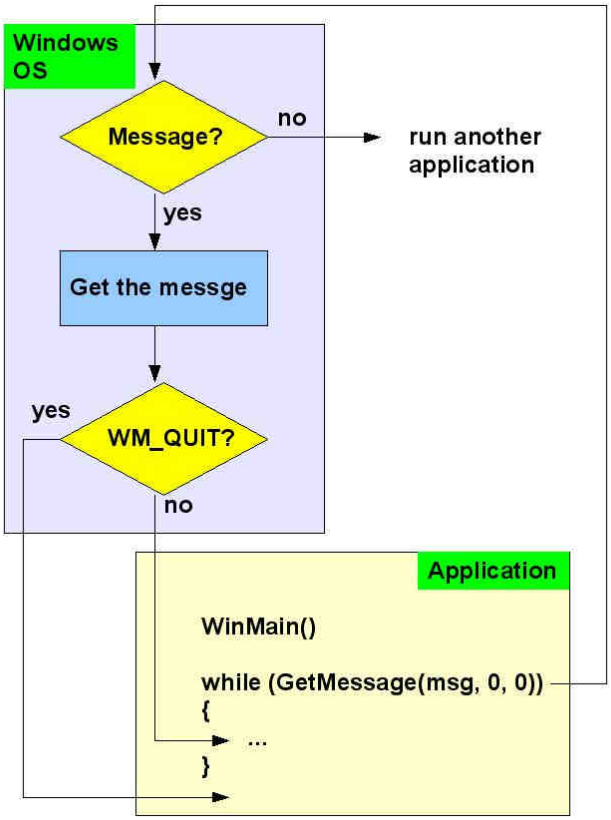
Používá ve svých programech různé knihovny, kde jsou definovány funkce, makra, struktury a objekty pro různé činnosti a množinu toolkitů, což je soubor předpřipravených nástrojů/komponent pro zjednodušení a zrychlení vývoje aplikací – naprogramovaná tlačítka, menu atd. (např. v operačním systému Microsoft Windows jsou to knihovny na rozhraní windows.h a vývojové nástroje SDK; v distribucích operačního systému Linux jsou to knihovny postaveny na knihovně Xlib a toolkity GTK+ nebo Qt). Operační systémy, které mají grafické uživatelské rozhraní, typicky používají model – řízení událostmi. Akce uživatele, jako stisk klávesy nebo pohyb myší, jsou překládány na události. Událost je datová struktura obsahující typ (např. stisk tlačítka myši) a další parametry (poloha myši a identifikátor okna, v němž je kurzor). Program přijímá události a volá pro ně obslužné funkce. Jádrem událostmi řízeného kódu je cyklus zpracování událostí. Detaily jsou v různých grafických prostředích odlišné, ale princip je vždy stejný. Zdrojem událostí není jen uživatel. Události si mohou posílat i aplikace (procesy) navzájem – to je třetí způsob, kde komunikaci obstarává systém samotný. Další informace o operačních systémech viz články na adrese [10].

2.1 Aplikační rozhraní systému Microsoft Windows

Programátorské aplikační rozhraní operačního systému Windows společnosti Microsoft se nazývá Windows API (zkráceně WinAPI; API značí application interface). WinAPI rozhraní je kompatibilní na všech verzích systému Windows a je použitelné jak pro 32-bitové, tak 64-bitové aplikace. Je navrženo pro použití s programovacími jazyky c a c++.

Proč vlastně programovat ve WinAPI? Ve vývojových prostředích Microsoft Visual C++ nebo C++ Builder fy Borland existuje sice velké množství již hotových komponent, ale velice často se stává, že potřebujeme změnit třeba činnost nebo vzhled některé z nich nebo daná komponenta vůbec neexistuje. Pomocí WinAPI ji můžeme snadno vytvořit. Dalším argumentem je efektivnost a výkonnost programu napsaného čistě ve WinAPI – lze napsat malý, relativně jednoduchý program, který je efektivní především z hlediska minimalizace nároků na paměť a další systémové zdroje. Týká se to především programů běžících většinou na pozadí po celou dobu běhu operačního systému. Nevýhodou programů napsaných ve WinAPI je závislost na operačním systému Windows (zprovoznění takové aplikace pod jiným systémem než Windows může být složitější; není ovšem nemožné, protože existuje řada emulátorů nebo implementací Windows API pro jiné systémy – například pro systém Linux existuje implementace WINE, která umožňuje spouštět programy určené pro Windows; více informací o WINE na [10]).

Systém Windows je založen na tzv. "událostmi řízeném programování". Z toho plyne, že o jakékoli události v systému jsou posílány jednotlivým oknům zprávy, které tuto událost nějak charakterizují. Takovou událostí může být stisk klávesy, akce s myší, uplynutí nějakého aplikací definovaného timeru (časová prodleva) atd. Každá událost je reprezentována zprávou, která se skládá z identifikátoru a dvou parametrů (identifikátor určuje typ zprávy – např. stisk tlačítka myši a parametry obsahují rozšiřující informaci o zprávě např. bylo stisknuto levé tlačítko myši a zároveň byla stisknuta klávesa shift). Tyto zprávy jsou pak doručovány oknům, pro která jsou určena. Okna na ně mohou, nebo nemusí reagovat. Významným rysem takového programu je, že je schopen čekat na podněty z nejrůznějších směrů.



obr. 2.1 – schéma smyčky zpráv Windows

Obrázek 2.1 znázorňuje schéma smyčky zpráv OS Windows. V systému vznikají různé události, které jsou reprezentovány zprávami. Jednotlivé zprávy se hromadí v zásobníku zpráv. Každá aplikace zjišťuje jestli přišla nová zpráva a jestli je určena pro ni. Pokud není pro konkrétní aplikaci žádná zpráva určena, systém předává řízení jiné aplikaci. Pokud se však objeví zpráva určená pro danou aplikaci, je převzata funkcí GetMessage(), přeložena a zpracována. O zpracování zprávy se může

postarat programátor nebo systém. Programátor ve svém programu zachytí zprávy na které chce reagovat a zbytek zpráv předá systému, který je automaticky vyřídí. Aplikační smyčka běží tak dlouho, dokud nedojde zpráva WM_QUIT – zpráva o tom, že uživatel požaduje ukončení programu.

2.2 Základy programování ve WinAPI

V této podkapitole si ukážeme základní strukturu programu, kterou musí obsahovat každá WinAPI aplikace. Nejprve bych se rád zmínil o dobře zpracovaných výukových lekcích o programování ve WinAPI pro začátečníky, ze kterých jsem čerpal. Nachází se na adrese [11] – článek Učíme se Win API. Dále jsem používal velmi kvalitní online nápovědu k WinAPI na adrese [7].

Struktura WinAPI aplikací se samozřejmě liší podle toho k čemu je aplikace určena. Zcela určitě bude mít jinou strukturu rezidentní program, který běží na pozadí a program, který má jedno nebo více oken s grafickým rozhraním. Jelikož se zabýváme vývojem aplikace určené pro šíření elektronického vzdělávání bude naše aplikace obsahovat minimálně jedno hlavní okno, které bude schopno zobrazovat grafické i textové komponenty. Dále by měla být aplikace schopna přehrávat zvuky, hudbu a další multimediální obsah. Proto se dále v textu budeme věnovat takové struktuře programu, která je pro tento účel určena. Než se podíváme na malé ukázkové kódy programu, probereme tři základní prvky takové WinAPI aplikace:

- zpracování smyčky zpráv
- registrace třídy okna a vytvoření okna
- procedura okna

Zpracování smyčky zpráv se odehrává v hlavním cyklu programu. Zde aplikace přijímá příchozí zprávy od systému, uživatele nebo ostatních aplikací. Každá zpráva je předána dále ke zpracování proceduře okna. Pro přijetí zprávy ze smyčky zpráv jsou k dispozici dvě funkce: GetMessage() a PeekMessage(). První jmenovaná funkce automaticky odstraňuje přijaté zprávy ze smyčky zpráv a po příchodu zprávy WM_QUIT (zpráva o ukončení programu) je její návratová hodnota FALSE. U funkce PeekMessage() můžeme blíže specifikovat, jak se zprávy budou zpracovávat. Její návratová hodnota je FALSE v případě, když není ve smyčce zpráv k dispozici žádná zpráva.

V každé aplikaci, která obsahuje alespoň jedno okno, musíme registrovat třídu okna. Třídy definují určité vlastnosti, které jsou společné všem oknům, které k této třídě náleží. Existují předdefinované systémové třídy, ke kterým můžeme ihned vytvořit okno patřící k takové systémové

třídě. Jsou to například standardní prvky Windows, jako tlačítko (buton), seznam (list-box), editační pole (edit-box) apod. Po zaregistrování třídy, můžeme vytvořit okno patřící, k této třídě.

Posledním stavebním kamenem je procedura okna. Procedura okna se stará o zpracování příchozích zpráv obdržených ve smyčce zpráv. Jelikož aplikace může mít více oken, musíme určit u každého okna, ke které proceduře okna patří a do ní je potom zpráva odeslána. K jedné proceduře okna může patřit i více oken, ale potom se všechna okna chovají na příchozí zprávy stejným způsobem.

Následuje ukázka jednoduchého programu používajícího WinAPI. První co musíme vložit do programu je hlavičkový soubor windows.h, kde se nacházejí všechny potřebné deklarace pro práci s Windows rozhraním. Každý program v jazyce c nebo c++ musí obsahovat hlavní funkci, která je spuštěna po startu aplikace – v našem případě je to WinAPI funkce WinMain. Nejdůležitější parametr je hInstance – handle instance, který nám přidělí operační systém při spuštění programu. Jde o 32-ti bitové číslo jednoznačně identifikující náš běžící program mezi ostatními programy a vlákny v systému, podle kterého lze poznat, zdali jsou příchozí zprávy určeny pro naši aplikaci. Funkce MessageBox() vykreslí dialogové okno s výpisem "Hello world!" a tlačítkem OK.

```
#include <windows.h>

int WINAPI WinMain(HINSTANCE hInstance,
                   HINSTANCE hPrevInstance,
                   LPSTR lpCmdLine,
                   int nCmdShow)
{
  MessageBox(NULL, "Hello world!","First program", MB_OK);
  return 0;
}
```

V hlavní funkci aplikace WinMain je zpravidla umístěno zpracování smyčky zpráv. Zde se přijímají a zpracovávají jednotlivé zprávy zaslané naši aplikaci. Funkce PeekMessage() vezme jednu zprávu z fronty zpráv. Pokud není aplikaci určená žádná zpráva je návratová hodnota funkce FALSE. Obdržené zprávy jsou odeslány k dalšímu zpracování. V případě, že přijde zpráva WM_QUIT je nastavena proměnná done hlavního cyklu na hodnotu TRUE a posléze dojde k ukončení celé funkce WinMain, a tedy celé aplikace. Následující program znázorňuje celou situaci. (pozn.: většina zpráv začíná WM – windows message; jsou definovány v hlavičkových souborech, ty běžné konkrétně ve winuser.h, který je vložen v souboru windows.h)

```
MSG msg;
bool done = false;

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInst,
                   LPSTR lpCmdLine, int nCmdShow)
{
  while(!done) // hlavni cyklus programu
  {
    while (PeekMessage(&msg, NULL, 0, 0, PM_REMOVE))
    {
      if (msg.message == WM_QUIT) done = TRUE;

      TranslateMessage(&msg);
      DispatchMessage(&msg);
    }
  }

  return 0;
}
```

Ve smyčce zpráv se nachází funkce TranslateMessage. Tato funkce překládá virtuální kódy zpráv klávesnice a generuje další "znakové" zprávy. Následuje funkce DispatchMessage, která zprávu odešle do příslušné procedury okna, kde je zpracována. Každé aplikaci náleží jedno nebo více oken. Při vytváření okna ve WinAPI se musí nejdříve zaregistrovat třída okna (zde se určují hlavní vlastnosti okna), po té se vytvářejí okna patřící k dané třídě okna. Ve frontě zpráv jsou zprávy náležící všem oknům aplikace, ale v proceduře okna pouze zprávy náležící oknu nebo oknům, patřícím k té třídě, která si tuto proceduru okna "zaregistrovala". Procedura okna má následující tvar:

```
LRESULT CALLBACK WindowProcMain(HWND hwnd, UINT uMsg,
                                WPARAM wParam, LPARAM lParam)
{
  switch (uMsg)
  {
    case WM_DESTROY:
      PostQuitMessage(0);
      break;
  }
  return DefWindowProc(hwnd, uMsg, wParam, lParam);
}
```

Handle okna hwnd nám identifikuje jednotlivá okna v aplikaci a díky něj můžeme rozhodnout, pro které okno je příchozí zpráva určena. Parametr uMsg je identifikátor zprávy (deklarace v souboru windows.h) – může se jednat například o zprávy WM_QUIT – ukončení aplikace nebo WM_LBUTTONDOWN – stisk levého tlačítka myši. Zbylé dva parametry nesou doplňující informace o zprávě.

Následuje kód, kde se reaguje na příslušné zprávy (v našem případě zpráva WM_DESTROY značí, že okno je právě rušeno a v okamžiku doručení zprávy je již odstraněno z obrazovky; jako reakci na tuto zprávu musíme do smyčky zpráv "ručně" vložit zprávu WM_QUIT. To realizujeme funkcí PostQuitMessage(). V hlavním programu je po příchodu zprávy WM_QUIT nastavena řídící proměnná done na hodnotu TRUE a je ukončen celý program.

Zprávy, které programátor nezpracuje musejí být předány systému pomocí funkce DefWindowProc(). Jedná se o různá překreslování a přesouvání oken a další činnosti, kterými se programátor nechce zatěžovat a jsou tedy automaticky zpracovány systémem. Pokud by programátor vyžadoval jiné chování programu než standardní, musel by příslušné zprávy zachytit a předepsat příslušné chování.

V závěru této kapitoly bych se ještě rád zmínil o univerzálních knihovnách určených pro programování aplikací, kde je kód programu v podstatě nezávislý na platformě, kde se bude překládat a spouštět (jeden zdrojový kód bez jakýchkoliv úprav přeložíme na různých platformách). Mezi používanější patří knihovna GLUT určená pro grafickou knihovnu OpenGL (Open Graphic Library viz další kapitola). Knihovna GLUT je implementována pro operační systémy Windows a Linux. Jde o množinu funkcí pomocí níž vytvoříme okna aplikace a definujeme její chování. Nezáleží pro jaký systém bude aplikace určena a vždy se použijí stejné programovací funkce. Rozdíl nastane až při

překladu, kdy musíme použít implementaci GLUTu určenou pro daný operační systém. Implementace určená pro Windows volá následně funkce WinAPI (chová se jako by byla ve WinAPI naprogramována).

Tento způsob programování se zdá velice lákavý, protože nabízí možnost napsat jeden program a spouštět ho bez úprav na různých systémech. Vždy je zde ale jisté omezení. Nelze zasahovat do jádra knihovny a proto můžeme víceméně programovat pouze vnější chování aplikace (při programování s GLUTem není vůbec vidět smyčka zpráv ani zprávy samotné, vše je před programátorem skryto). Obecně používáním knihoven, které se snaží sjednotit různé platformy, má programátor částečně svázané ruce. Je to dáno rozdílností operačních systémů. Pokud má knihovna nějak rozumě fungovat musí se vzít množina shodných vlastností systémů a ty do knihovny implementovat (tedy knihovna nikdy neposkytne úplnou funkčnost, protože funkce která existuje v jednom systému nemusí být v jiném).

Použití těchto knihoven je vhodné pro jednoduché aplikace, které nevyžadují nestandardní funkce nebo hlubší zásahy do systému. Pokud však programátor vyžaduje 100% kontrolu nad aplikací a systémem + maximální výkon a úsporu zdrojů, použije čistě aplikační rozhraní daného operačního systému.

3 Nástroje pro zobrazení 3D grafiky a přehrání audio záznamů

V této kapitole si ukážeme jak lze v aplikacích zobrazovat 3D grafiku a přehrávat audio záznamy. Pro správnou funkčnost grafického a zvukového adaptéru je nutnost mít správně nainstalované ovladače dodané od výrobce. Mezi nejznámější výrobce grafických čipů patří NVIDIA a ATI; pro zvukové karty je to CREATIVE.

Každá aplikace pro zobrazení 2D nebo 3D grafiky potřebuje běžet v grafickém režimu, tedy potřebuje od operačního systému grafickou podporu (přístup ke grafickému adaptéru). U systému Windows je grafické prostředí zabudováno přímo do jádra systému (grafický režim se spouští rovnou se systémem). Systémy Linux/Unix mají jádro od grafického prostředí odděleno. Pro zobrazení aplikací v grafickém režimu se musí spustit uživatelské rozhraní zvané X Window System.

Pro vytvoření 3D aplikace pod systémem Linux/Unix je nejvhodnější a nejrychlejší použít knihovnu GLUT, zmíněnou v minulé kapitole (nebo přímo knihovnu Xlib a nad ní jsou knihovny nástrojů Qt a GTK+) ve spojení s knihovnou OpenGL. Díky knihovně GLUT vytvoříme okna aplikace a definujeme její chování. Knihovna OpenGL zajišťuje vykreslování 2D nebo 3D grafiky do vytvořených oken aplikace. V systému Windows můžeme opět využít kombinace GLUT-OpenGL nebo WinAPI-OpenGL. Místo knihovny OpenGL můžeme použít knihovny na rozhraní DirectX určené přímo pro systém Windows vyvinuté od společnosti Microsoft. Rozhraní DirectX není určeno pouze pro zobrazování 2D/3D grafiky. Je to množina knihoven navrhnuta pro správu každého nainstalovaného hardwaru. Slouží pro obsluhu zvukových, zobrazovacích, polohovacích, síťových a dalších zařízení v systému. Každé skupině zařízení odpovídá i množina komponent s příslušným názvem: DirectSound, Direct3D, DirectDraw, DirectInput, DirectPlay. Pro zobrazení grafiky bychom tedy použili DirectDraw a Direct3D (pozn. mezi jednotlivými verzemi DirectX se funkce jednotlivých komponent mírně liší – je horší kompatibilita mezi nejnovějšími a staršími verzemi; současná verze je 9.0c).

Pro přehrání audio záznamů můžeme využít opět knihovny určené k tomuto účelu. V systému Windows je to DirectSound z rozhraní DirectX nebo funkce z rozhraní WinAPI. V systémech Linux/Unix je to zvukový ovladač OSS (Open Sound systém), který je přímo zabudovaný v jádru systému. Dnes je to ale zastaralý ovladač a přechází se na ovladač ALSA (Advanced Linux Sound Architecture) – umožňuje používat více zvukových adaptérů současně.

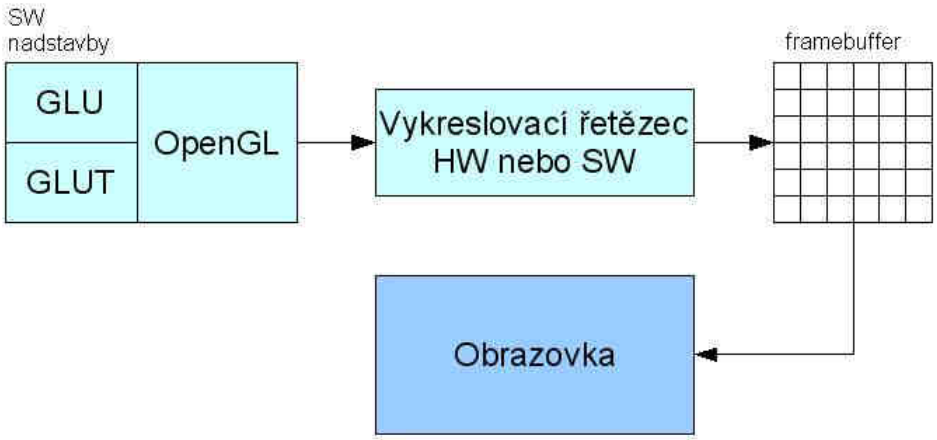
3.1 Knihovna Open GL – zobrazení 3D grafiky

Standard OpenGL vznikl na začátku devadesátých let, navržen firmou SGI (Silicon Graphics Inc.). Od počátku je koncipován jako nezávislý na hardwaru, operačním systému a programovacím jazyce. Od té doby se změnil pouze minimálně, nejnovější verze je 2.0. Knihovna OpenGL slouží k vytvoření grafického aplikačního rozhraní. Umožňuje pracovat ve 2D i 3D režimu zobrazování – využívá akcelerované (hardwarové) podpory karet. Předchůdcem této knihovny byla knihovna IRIS GL (Silicon Graphics IRIS Graphics Library). Knihovna OpenGL je navržena univerzálně, tak aby mohla poskytnout podporu pro jakýkoli typ grafického akcelerátoru. V případě, že na platformě není žádný nainstalován je použita softwarová simulace. Dnes lze knihovnu OpenGL použít na různých verzích unixových systémů (včetně Linuxu), OS/2 a na platformách Microsoft Windows. Příkazy (funkce) knihovny OpenGL lze použít v téměř jakémkoliv programovacím jazyce (nejčastěji se používá v c, c++).

3.1.1 OpenGL – základní principy

OpenGL se snaží napodobit vzhled reálného světa nebo systému ve 3D prostředí. Sestává se ze 3D scény a 3D objektů, které se v této scéně nacházejí. Objekty jsou tvořeny primitivy, což jsou základní geometrické prvky. Mezi tato primitiva patří bod, úsečka, trojúhelník, čtyřúhelník, plošný konvexní polygon, bitmapa (jednobarevný rastrový obraz) a pixmapa (barevný rastrový obraz). Na vrcholy jednotlivých grafických primitiv lze aplikovat různé transformace (otočení, změna měřítka, posun, perspektivní projekce), pomocí kterých lze vytvořit animace. Vykreslovaná primitiva mohou být osvětlena nebo pokryta texturou. Z hlediska datové reprezentace vykreslované scény poskytují funkce OpenGL pouze základní rozhraní pro přístup ke grafickým akcelerátorům. Existují však rozšiřující knihovny, které funkčnost dále zvyšují. Jednou ze základních knihoven používaných společně s OpenGL je knihovna GLU (OpenGL Utilities).

Z programátorského hlediska se OpenGL chová jako stavový automat. Během zadávání příkazů pro vykreslování lze průběžně měnit vlastnosti vykreslovaných primitiv nebo celé scény (barva, průhlednost, volba způsobu vykreslování, transformace – natočení, posunutí, ...). Výhoda tohoto přístupu spočívá především v tom, že funkce pro vykreslování mají menší počet parametrů a že jedním příkazem lze globálně změnit způsob vykreslení celé scény. Vykreslování scény se provádí procedurálně. Voláním funkcí OpenGL se vykreslí výsledný rastrový obrázek uložený v tzv. framebufferu (oblast paměti grafické karty), kde je každému pixelu přiřazena barva, hloubka, alfa složka (průhlednost) popř. i další atributy. Z framebufferu lze získat pouze barevnou informaci a tu je možné následně zobrazit na obrazovce viz následující obrázek 3.1.

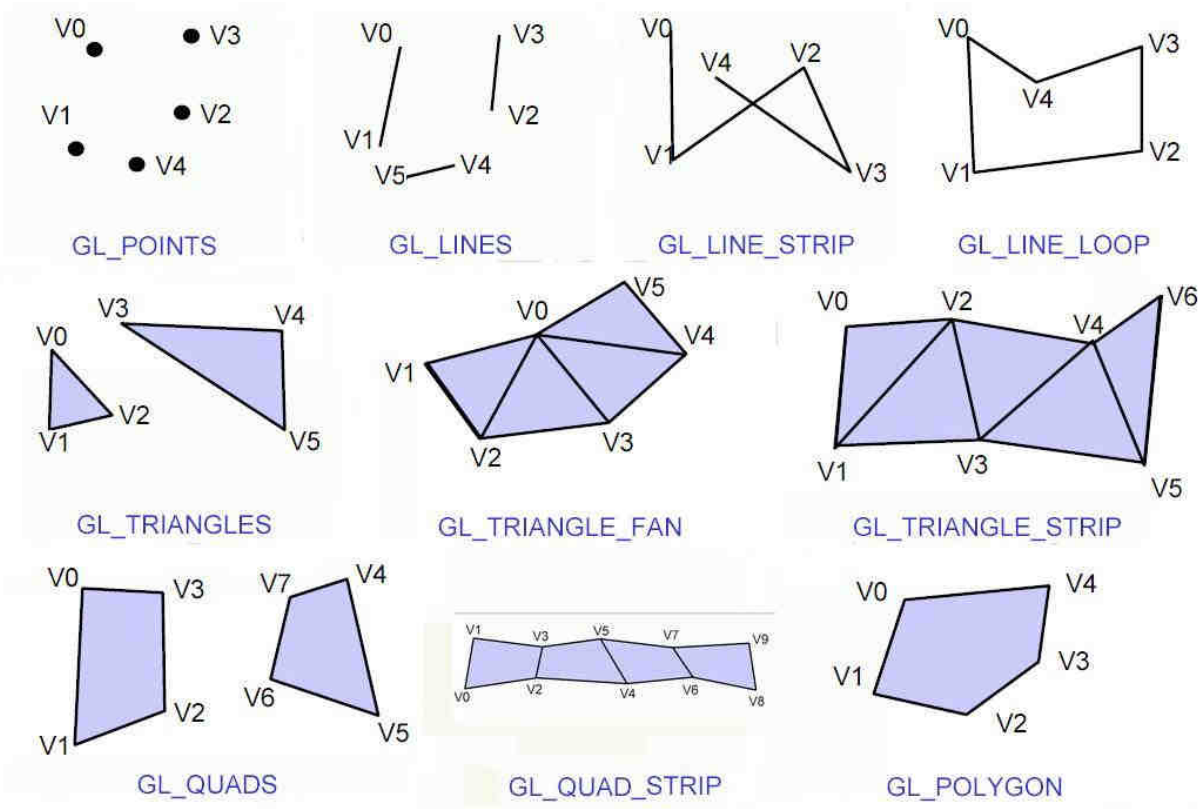


obr. 3.1 – vykreslování v OpenGL

Pro dosažení co největší nezávislosti na použité platformě zavádí knihovna OpenGL vlastní primitivní datové typy (např.: GLbyte, GLint nebo Gldouble). Programátorské rozhraní knihovny OpenGL je vytvořeno tak, aby knihovna byla použitelná v téměř libovolném programovacím jazyce.

Následující kód znázorňuje vykreslení trojúhelníku na obrazovku. Mezi příkazem glBegin() a glEnd() jsou uzavřené souřadnice vrcholů grafického primitiva. Parametr GL_TRIANGLES oznamuje OpenGL, že následující souřadnice vrcholů má použít pro vykreslení trojúhelníku. Další možná primitiva jsou body (GL_POINTS), úsečky (GL_LINES), trojúhelníky (GL_TRIANGLES), čtyřúhelníky (GL_QUADS), mnohoúhelník (GL_POLYGON) viz obr. 3.2. Nejčastěji se však k vykreslení používají trojúhelníky.

```
glBegin(GL_TRIANGLES);
   glVertex3f( 0.0f, 1.0f, 0.0f);
   glVertex3f(-1.0f,-1.0f, 0.0f);
   glVertex3f( 1.0f,-1.0f, 0.0f);
glEnd();
```



obr. 3.2 – primitiva OGL

Vlastnosti primitiv nebo cele scény se mění pomocí příkazů glEnable() a glDisable(). Jsou to příkazy k povolení nebo zakázání určité vlastnosti nebo schopnosti, kterou určíme zadaným parametrem. Můžeme tak aktivovat nebo deaktivovat průhlednost primitiv (GL_BLEND), světla ve scéně (GL_LIGHT), nanášení textur na objekty (GL_TEXTURE_2D), efekt mlhy (GL_FOG) a mnoho dalších vlastností viz [7].

V OpenGL existují tři transformační matice – ModelView (používá se pro nastavení pozice kamery a pro manipulaci s objekty), Projection (pro nastavení perspektivní projekce kamery) a Viewport (pro mapování objektů z abstraktních souřadnic do souřadnic okna). Transformační matice představují v počítačové grafice prostředek pro vyjádření lineárních transformací, mezi které patři posun, otočení, změna měřítka a zkosení. Důležitou vlastností lineárních transformací je, že je lze skládat. Tím se urychlí vlastní vykreslování.

3.2 Knihovna FMOD Ex – přehrání audio záznamů

Revoluční systém FMOD Ex je určen pro zvukovou podporu aplikací. Je určen pro vývojáře her, multimediální vývojáře a hudebníky. Vychází z předchozího systému FMOD od společnosti Firelight Technologies' viz [6]. První velkou předností tohoto systému je platformní nezávislost. Podporuje platformy: Microsoft Windows 32/64 bit, Linux 32/64 bit, Macintosh, Sony PlayStation 2, Microsoft Xbox a další konzolová zařízení. Další výhoda je možnost používat systém zcela zdarma v případě, že konečná aplikace je opět poskytnuta zdarma (pro nekomerční účely je zadarmo). Další výhodou oproti jiným zvukovým systémům jsou velmi nízké systémové požadavky (zatížení procesoru je velmi nízké).

Ke každé platformě je na [6] připravený ke stažení instalační balíček. Po nainstalování získáme sadu knihoven a hlavičkových souborů. Systém FMOD Ex je navržen pro vývoj v jazycích c, c++ a c#. Předchůdce systému FMOD Ex nebyl objektový a nabízel pouze strukturované programování. V novém systému jsou však přídavné hlavičkové soubory umožňující objektové programování.

3.2.1 FMOD EX – základní principy

Systém nabízí velké množství funkcí – od běžného přehrání zvukové stopy, až po simulaci 3D zvuku (verze 4.02 nabízí podporu až pro 7.1 zvukové systémy – 7 reproduktorů + 1 basový reproduktor). Systém může ovládat více nainstalovaných zvukových adaptérů současně. Velká je i podpora hudebních formátů. Neexistuje téměř žádný hudební soubor, který by systém neuměl přehrát. Podporované formáty (z těch nejznámějších a nejpoužívanějších): wav, raw, mp2, mp3, ogg, wma, s3m, mod, mid a další.

Pokud chceme použít ve svém programu zvukový systém FMOD Ex, stačí vložit příslušné hlavičkové soubory do hlavního modulu programu. V instalačním balíčku systému je velice kvalitní nápověda, která obsahuje přehled všech dostupných funkcí. V následující ukázce je jednoduchá struktura programu, která umožní přehrání zvukového souboru.

```
#include "fmod.hpp"
#include "fmod_errors.h"
#include <windows.h>
int main(int argc, char *argv[])
{
    FMOD::System     *system;
    FMOD::Sound      *sound;
    FMOD::Channel    *channel = 0;

    FMOD::System_Create(&system);

    system->init(32, FMOD_INIT_NORMAL, 0);

    system->createSound("data.wav", FMOD_HARDWARE, 0, &sound);
    system->playSound(FMOD_CHANNEL_FREE, sound, false,
                      &channel);

    sound->release();
    system->release();
}
```

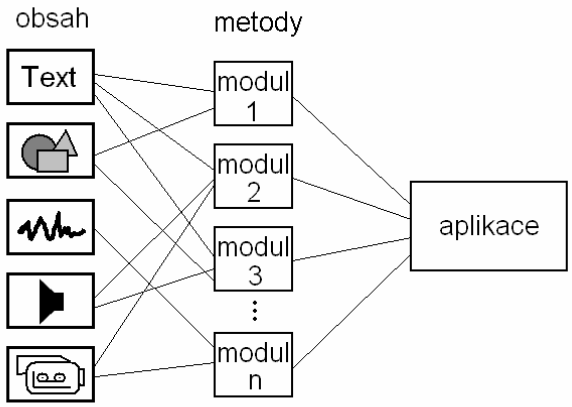
Nejdůležitější proměnnou je 'FMOD::System' – což je ukazatel na hudební systém. Takových systémů můžeme vytvořit několik. Každý pak funguje nezávisle na ostatních. Každý systém může být inicializován s jinými parametry. Uvnitř každého systému mohou byt otevřené a přehrávané jiné hudební soubory. Provedeme vytvoření a inicializaci každého systému – zde je nejdůležitější parametr počet kanálů. Toto číslo určuje, kolik se může současně přehrávat hudebních souborů v jednom systému. Příkazem createSound() se otevře příslušný hudební soubor, který se celý načte do paměti a poté se začne přehrávat. (počet načtených souborů je omezen pouze kapacitou paměti). Existuje ještě příkaz createStream(), který zahájí přehrávání souboru již během načítání do paměti (streamování). Příkaz playSound() přehraje ve zvoleném hudebním systému, zvolenou skladbu na příslušném kanálu (atribut FMOD_CHANNEL_FREE najde a vybere právě nepoužívaný kanál). Na konci programu se musí uvolnit paměť (sound – >release uvolní paměť po hudebním souboru; systém – >release uvolní paměť hudebního systému).

4 Specifikace požadavků na multimediální systém

V diplomové práci navazuji na ročníkový projekt – Systém pro podporu výuky jazyků. Jak název napovídá, zabýval jsem se vývojem aplikace určené pro výuku cizích jazyků. V této kapitole je nastíněna specifikace původního ročníkového projektu a jsou uvedená rozšíření této specifikace o možnosti zobrazování 3D objektů a přehrávání zvukových záznamů.

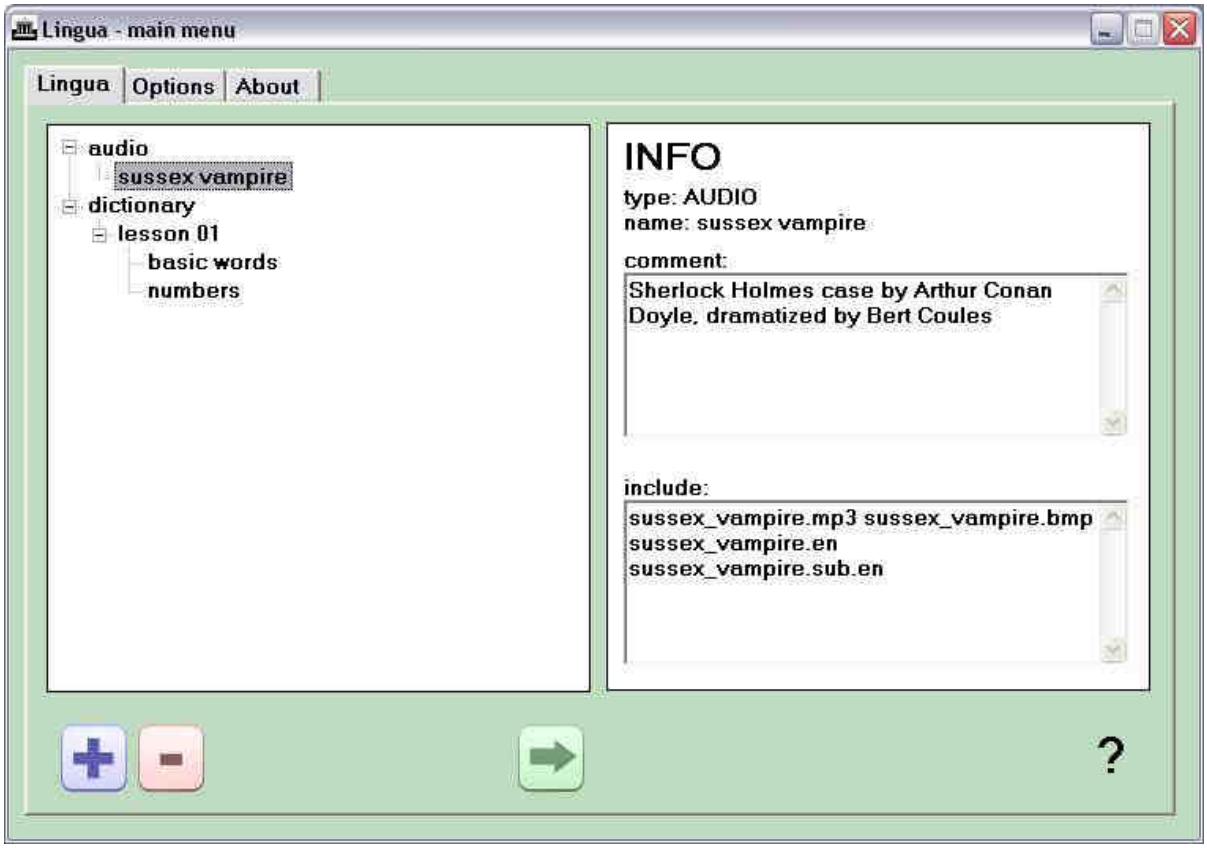
4.1 Specifikace ročníkového projektu

Název systému jsem zvolil Lingua. Podle zásad e - learningu je systém rozdělen na dvě části. Část obsahovou, což je databáze, ve které jsou uloženy všechny informace (obrázky, texty, zvuky …) a programovou část, kde je definováno, jak tyto informace prezentovat viz obr. 4.1. Program obsahuje dva moduly, které nabízí různé přístupy k výuce. Modul Dictionary slouží k naučení a procvičení slovní zásoby a nabízí možnost současné výuky několika jazyků. Modul Audio nabízí procvičení poslechu a četby. Systém je naprogramován v jazyce c++ ve vývojovém prostředí C++ Builder fy Borland (je to tedy statická aplikace). Nabízí standardní prostředí 'oken' a 2D grafiku tvořenou WinAPI objekty (tlačítka, menu, textová pole, …). Je jednouživatelský. Distribuce může probíhat např. pomocí CD-ROM nosičů.



obr. 4.1 – aplikace obsahuje několik modulů, kde jsou definovány metody jak prezentovat elektronický obsah (text, obrázky, video, audio)

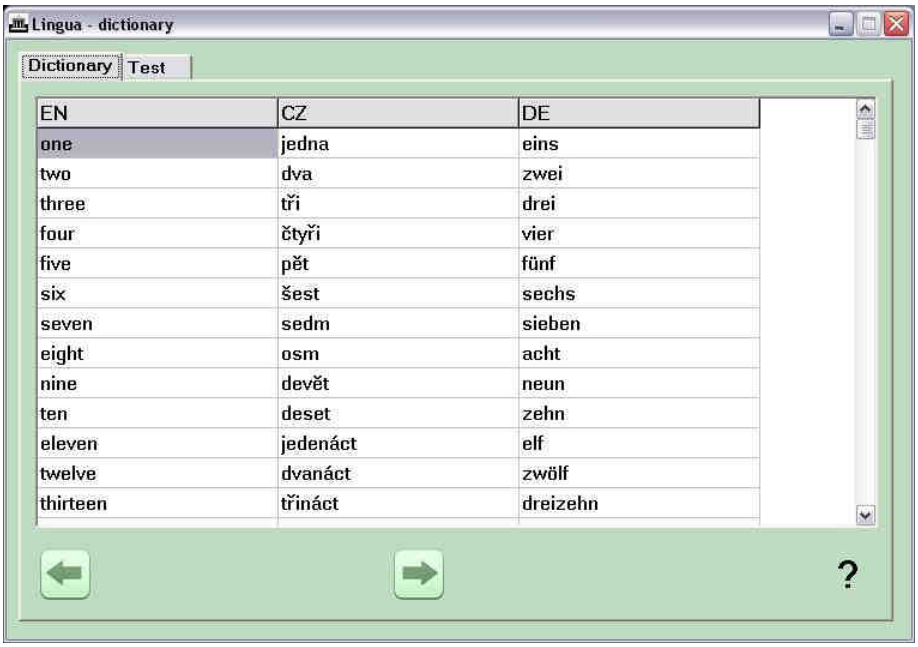
Vzhled a ovládací prvky aplikace byly navrženy co nejjednodušeji tak, aby ovládaní bylo intuitivní. Po spuštění aplikace se ocitneme v hlavním menu viz obr. 4.2a. Aplikace v každém okně nabízí přehledné záložkové menu, které se nachází v horní části (jeho obsah se mění v závislosti na tom v jaké části programu se nacházíme). Ovládací prvky jsou soustředěny do části dolní, kde se nachází i tlačítko ve tvaru otazníku, které poskytuje nápovědu k aktuálnímu oknu. Hlavní menu aplikace má tři záložky – Lingua, Options a About. Při výběru záložky Lingua se nám zobrazí dva bílé panely. V levém panelu je stromová struktura všech lekcí. Kořenem stromu jsou výukové moduly Audio a Dictionary. Stromovou strukturu lze rozbalit a zabalit pomocí modrého tlačítka plus a červeného tlačítka mínus. Kliknutím vybereme lekci. V pravém panelu se nám zobrazí informace o právě vybrané lekci. Je to název, komentář a soubory potřebné ke spuštění lekce. Typ lekce určuje, který z modulů bude spuštěn. Lekci spustíme kliknutím na zelené tlačítko se šipkou “next”.



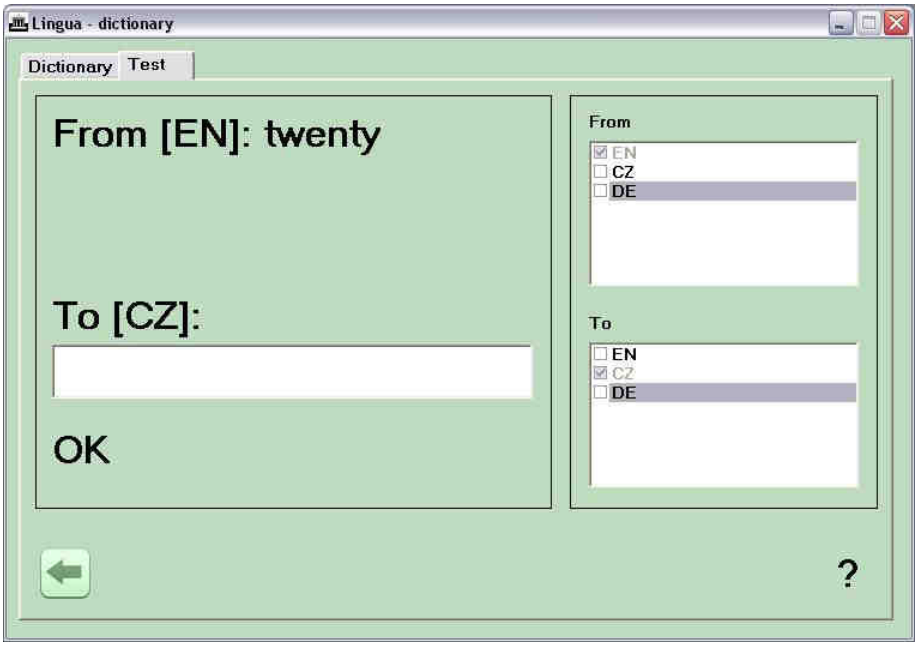
obr. 4.2a – Hlavní menu aplikace

Modul Dictionary viz obr. 4.2b, jak již název napovídá, se zabývá hlavně slovní zásobou. Slouží však k naučení a procvičení nejen slovíček, ale i různých frází a vět. Jak je z obrázku patrné program nabízí současnou výuku více jazyků. V první záložce s názvem Dictionary se nacházíme ve výukové části. Kliknutím na výraz se spustí přehrání výslovnosti (pozn.: ve verzi 1.0 je k dispozici pouze anglická výslovnost). Kliknutím na záložku s názvem Test nebo kliknutím na zelené tlačítko se

šipkou “next” se dostaneme do testové části modulu. V této části viz obr 4.2c nás program vyzkouší z právě naučené lekce. V pravé části okna vybereme mezi kterými jazyky má test probíhat. Můžeme vybrat standardní překlad z jednoho jazyka do jiného nebo vybrat skupinu jazyků, mezi kterými má test probíhat. Kliknutím do editačního pole a stiskem klávesy Enter zahájíme test. Opět je přehrána výslovnost požadovaného slovíčka, pokud se jeho audio záznam nachází v databázi. Při správném zadání požadovaného výrazu je zobrazen nápis OK. V opačném případě se nápis změní a zobrazí se správné řešení.

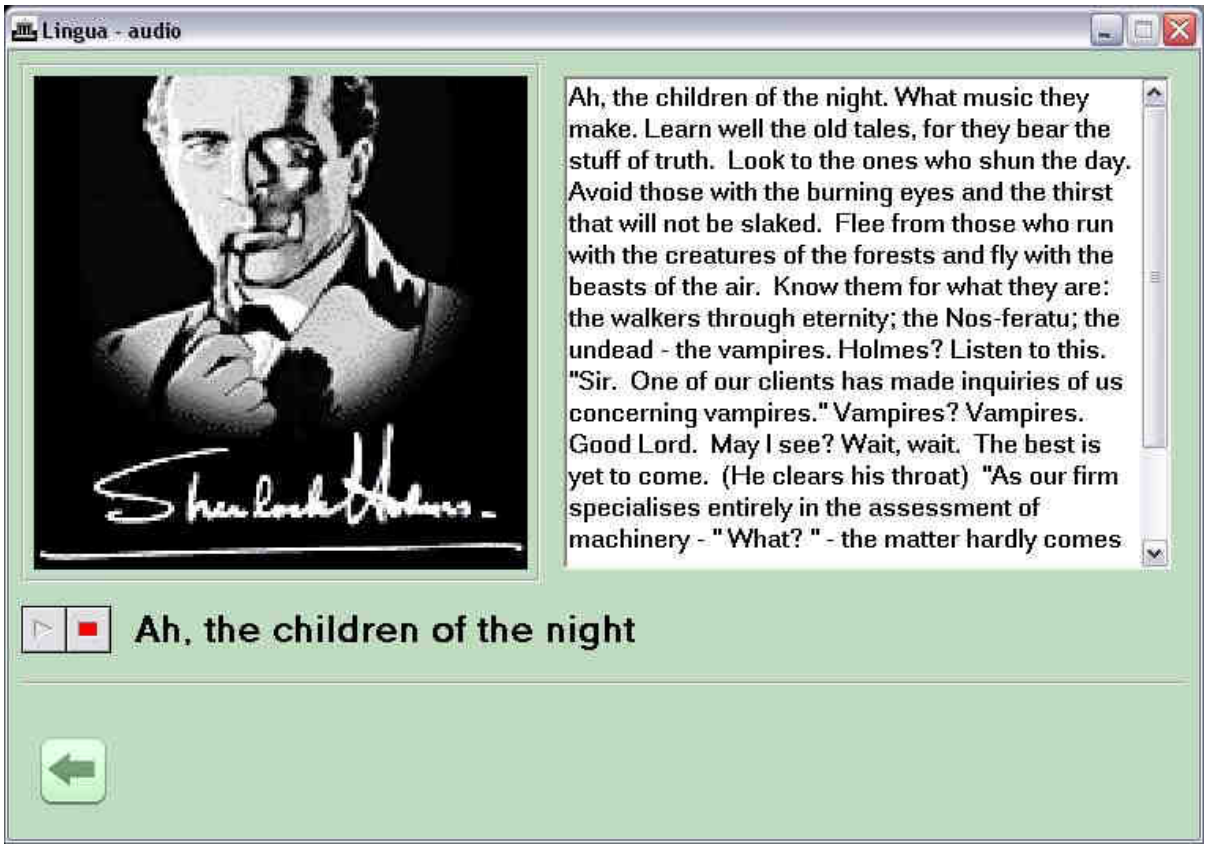


obr. 4.2b – Modul Dictionary, výuková část



obr. 4.2c – Modul Dictionary, testová část

Modul Audio viz obr. 4.2d nabízí procvičení poslechu a četby, popřípadě i výslovnosti. Obsahem bývá nějaká povídka nebo příběh. Audio záznam spustíme kliknutím na tlačítko “play”. Je doprovázen statickým textem v pravé části a titulky v části spodní. Zvukový záznam lze pozastavit pomocí červeného tlačítka stop.



obr. 4.2d – Modul Audio

4.2 Rozšíření specifikace

V nové verzi systému zůstane zachována filozofie e - learningu, což obnáší rozdělení systému na část obsahovou a prezentační. Bude také zachován modulový systém, kde každý modul nabízí jiný přístup k výuce. Rozšíření programu se projeví navenek tím, že pro zobrazení se použije 3D grafiky a 3D grafických objektů. Tato technika zobrazování by neměla přidat pouze na atraktivnosti aplikace, ale měla by i zpřehlednit a urychlit orientaci v systému. Další novinkou je maximální podpora multimédií (audio, video, grafika, …), tyto prvky jsou opět využity pro zpřehlednění systému. Další možností je doplnit systém o zvukovou nápovědu, která by uživateli ulehčovala práci se systémem. Cílem je nabídnout uživateli maximální usnadnění a intuitivní ovládání systému.

4.3 Modelovací nástroje

Původní verze systému byla naprogramována ve vývojovém prostředí C++ Builder fy Borland. Toto prostředí jsem zvolil z důvodu, abych se ho naučil používat a získal nové programátorské zkušenosti. Při implementaci jsem využil vestavěné komponenty pro přehrání zvuků, zobrazení textů a obrázků. Nemusel jsem tedy tyto komponenty vytvářet. Z programátorského hlediska jsem se nemusel zabývat vytvářením aplikačních oken nebo zpracováním smyčky zpráv. Vše je již zajištěno v knihovnách C++ Builderu a před programátorem skryto v podobě funkcí a komponent.

Při implementaci nového systému bych se chtěl touto problematikou zabývat více. Chci se naučit vytvářet aplikační okna, do nichž se bude vykreslovat pomocí OGL kontextu, zpracovávat smyčku zpráv, vytvářet vlastní komponenty pro přehrání zvuku, zobrazení obrazu a textu. Z toho plyne, že při implementaci jádra nového systému nepoužiji nic z původního projektu. Ten je totiž vázán na komponenty a knihovny vývojového prostředí C++ Builder. Tyto komponenty si naprogramuji sám za využití funkcí programovacího jazyka a funkcí aplikačního rozhraní operačního systému.

Pro vytvoření systému použiji objektový programovací jazyk c++, pro zobrazení 3D grafiky knihovnu OpenGL a pro přehrání audio obsahu knihovnu FMOD Ex. Obě knihovny jsem zvolil z důvodu přenositelnosti mezi různými platformami. Další výhodou je, že vytvořený software pomocí těchto knihoven lze při nekomerčním využití šířit zdarma. Aplikační rozhraní bude vytvořeno pomocí WinAPI, které již nezajišťuje takovou přenositelnost jako předchozí knihovny. Pro toto rozhraní jsem se rozhodl z toho důvodu, že výukový systém bude určen pro operační systém Microsoft Windows a je lepší použít aplikační rozhraní systému, pro který je aplikace určena. Při přenosu aplikace do jiného operačního systému bude potřeba přepsat pouze ty části programu, obsahující tuto závislost na operačním systému.

5 Návrh systému

Jak již bylo zmíněno v předchozí kapitole, projekt se zabývá vývojem aplikace (multimediálního systému) pro výuku jazyků. Při návrhu jsem se řídil získanými zkušenostmi o elektronických kurzech. Položil jsem si několik důležitých otázek, jejichž zodpovězení ovlivnilo návrh systému a výběr programového vybavení pro jeho vytvoření.

- pro jakou skupinu uživatel bude kurz určen
- kolik uživatelů bude současně užívat jednu instalaci systému
- v jaké formě bude systém vytvořen (internetová / statická aplikace)
- jakým způsobem bude systém distribuován
- jaký bude grafický vzhled systému
- v jakém jazyce bude systém lokalizován
- typ elektronického obsahu (obrázky, audio, …)
- jakým způsobem bude elektronický obsah prezentován
- jak budou uložena data el. obsahu

Systém bude určen pro každého uživatele, který se chce naučit nový jazyk nebo si jen zlepšit své již dosažené znalosti. Jelikož by systém měl být vhodný pro co největší skupinu uživatel, a to bez ohledu na jejích věk nebo zkušenosti, měl by být kladen důraz na jeho jednoduchost a přehlednost. Tyto atributy jsou velice důležité hlavně v případě, kdy systém budou užívat uživatelé s menšími nebo žádnými zkušenostmi s prácí na počítači. Urychlí se tím hlavně naučení ovládání systému a pochopení jeho struktury. Uživatel se pak může plně věnovat jeho obsahu.

Systém bude jednouživatelský. Nebude obsahovat žádnou databází uživatelů. Odpadá tedy registrace a přihlašování uživatele. Jednu instalaci bude moci samozřejmě využívat více uživatelů, ale systém je nebude nijak rozlišovat.

Systém bude navržen jako statická aplikace spustitelná pod systémem Microsoft Windows XP. K distribuci bude použit CD-ROM nebo DVD-ROM nosič. Všechna potřebná data budou umístěna na tomto nosiči, a proto uživatel nemusí mít počítač připojen k internetu.

Nejvíce bude kladen důraz na vzhled aplikace. Použije se 3D grafický režim, který přidá na atraktivnosti systému. Pro přehlednost budou použity velké grafické komponenty – velká tlačítka, editační a textové boxy. Pro zobrazování textů budou použity větší fonty. Zlepší se tím čitelnost všech textů. Aplikace bude lokalizovaná v českém jazyce.

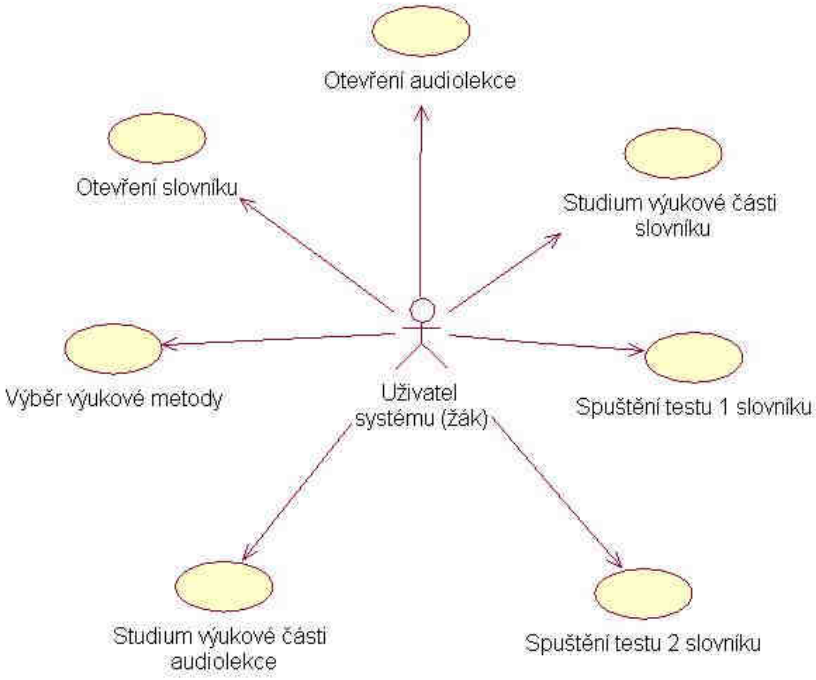
Aplikace bude obsahovat dva výukové moduly zaměřené na výuku slovní zásoby, gramatiky, poslechu, výslovnosti a četby. V každém modulu bude několik lekcí, ve kterých se uživatel nejprve

seznámí s novou látkou ve výukové časti a po té si ji procvičí v testové části. V systému bude použit k prezentaci elektronický obsah ve formě textů, obrázků a audio záznamů.

Elektronická data budou uložena formou souborů (nebude použit databázový systém). Použity budou standardní souborové formáty. Pro textový obsah bude použit formát *.txt, pro zobrazení obrázků formát *.bmp a pro audio záznamy *.wav nebo *.mp3. Soubory budou uloženy v adresářové struktuře systému.

5.1 Struktura systému

Aplikace bude obsahovat dva výukové moduly. První výukový modul bude zaměřen na výuku gramatiky, slovní zásoby a výslovnosti. Je označen jako modul SLOVNÍKY. Tento modul bude obsahovat několik ukázkových lekcí. Každá lekce bude rozdělena na dvě části – na část výukovou a část testovou. Ve výukové části bude uživateli prezentována určitá část gramatiky (např. použití přítomného času prostého v anglickém jazyce) a to formou textu, kde bude vysvětleno v jakých případech se gramatika používá a její pravidla. V lekcích zaměřených na slovní zásobu bude ve výukové časti zobrazen překladový slovníček s novými výrazy (např. anglicko – český). Po výukové části bude následovat ihned část testová. V případě procvičení slovní zásoby půjde pouze o překlad výrazů z jednoho jazyka do jiného. Gramatický test zaměřený např. na testování přítomného času prostého bude obsahovat věty, kde se tento čas používá (opět půjde o překlad vět). Velice důležité je, aby výukové části neobsahovaly mnoho nového učiva a aby se po každé výukové časti spustil test, kde si uživatel nově probranou látku procvičí. Druhý výukový modul bude zaměřen na četbu a poslech. Je označen jako modul AUDIO. Bude opět obsahovat několik výukových lekcí. Každá bude obsahovat článek, povídku, nebo jiný textový dokument, který bude určen k procvičení četby. Součástí bude namluvený audio záznam, který bude sloužit k procvičení poslechu. Na následujícím obr 5.1 vidíme diagram použití (use case) aplikace. Do systému přistupuje pouze uživatel (žák). Po výběru výukové metody (modulu) SLOVNÍKY nebo AUDIO následuje otevření příslušné lekce. Poté se spustí výuková a následně testová část lekce. Pro každou činnost zobrazenou v diagramu použití bude vytvořena jedna grafická scéna. Scéna výběru výukové metody bude hlavní menu aplikace. Následují dvě scény pro otevření audio nebo slovníkových lekcí. Dvě scény výukové a dvě testové. Aplikace bude obsahovat tedy 7 grafických scén.



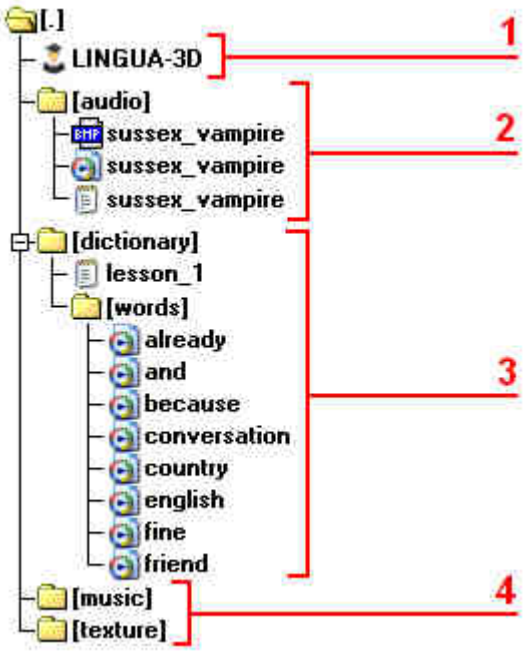
obr. 5.1 diagram použití

5.2 Uložení dat

Data aplikace budou uložena na nosiči CD-ROM. V kořenovém adresáři bude umístěn podadresář s názvem aplikace. Zde se bude nacházet spustitelný soubor aplikace a potřebné dynamické knihovny. Další soubory určené pro funkci aplikace jako textury a zvukové stopy budou uloženy v podadresářích texture a music. Jelikož bude aplikace obsahovat dva výukové moduly bude elektronický obsah uložen ve dvou adresářích. Modul SLOVNÍKY se bude nacházet v podadresáři dictionary a modul AUDIO v podadresáři audio.

Adresář dictionary bude obsahovat textové soubory. Každý soubor představuje jednu lekci modulu SLOVNÍKY. Součástí tohoto modulu bude i procvičení výslovnosti. Soubory s audio nahrávkami výslovnosti budou uloženy v podadresáři words. V adresáři audio se nacházejí soubory pro modul AUDIO. Každá lekce se skládá ze tří souborů (obrázek, zvukový záznam a text) viz obr. 5.2.

Po spuštění programu se načte obsah obou adresářů audio a dictionary, který je zobrazován ve scéně otevření slovníku a ve scéně otevření audio lekce.



obr. 5.2 adresářová struktura

1) spustitelný soubor systému
2) modul AUDIO: každá lekce se skládá ze tří souborů (obrázek, zvukový záznam a text). Všechny tři soubory každé lekce mají vždy stejný název (liší se pouze příponou). Po výběru lekce se otevřou všechny tři soubory.
3) modul SLOVNÍKY: každá lekce je uložena v samostatném textovém souboru. Adresář words obsahuje zvukové záznamy výslovností výrazů všech lekcí.
4) adresáře s texturami a zvuky potřebnými pro funkci aplikace.

Jelikož je každý soubor ve standardním formátu je možné jej editovat klasickými nástroji pro editaci textu, obrazu a zvuku. Uživatel tak může měnit obsah jednotlivých lekcí nebo přidávat lekce nové. V operačním systému Windows je pro editaci textových souborů k dispozici program Notepad (Poznámkový blok), pro editaci obrázků je to program Paint (Malování) a pro editaci zvuku Sound recorder (Záznam zvuku). Existují i alternativní editory viz (www.studna.cz).

Jinou možností, jak uložit elektronický materiál, je použití databázového systému. Data databáze jsou však přístupná pouze přes tento systém. Rozhodl jsem se uložit data do adresářové struktury v podobě souborů se standardním formátem. Jedině pak je možné data snadno prohlížet a editovat pomocí různých editorů textu, zvuku a obrazu. Data v adresářové struktuře jsou pro uživatele lépe přístupná oproti datům uloženým v databázi.

5.2.1 Slovníky

Každá lekce modulu Slovníky bude uložená v samostatném textovém, souboru. Po spuštění aplikace se načte obsah adresáře dictionary. Všechny nalezené soubory budou zobrazeny ve scéně Otevření slovníku viz obr. 5.1. Každý soubor bude obsahovat popisek lekce, data výukové části, data testové části 1 a data testové části 2. Pro zpřehlednění se popisek lekce bude zobrazovat i ve scéně Otevření slovníku spolu s názvem souboru. Testová část 1 bude formou testu – vybírá se jedná

správná odpověď z více možností. Testová část 2 bude formou doplňování – musí se zadat přesně požadovaný výraz.

Struktura textového souboru jedné lekce modulu Slovníky

soubor	význam
…	popisek / název lekce
#1	oddělovač
…	data výukové části slovníku
#2	oddělovač
…	data testové části 1
#3	oddělovač
…	data testové části 2

* zobrazované tečky (…) značí uložená data v textovém formátu viz ukázková lekce systému

5.2.2 Audio

Každá lekce modulu Audio bude uložená ve třech souborech – soubor obrázku, zvukového záznamu a textového obsahu. Po spuštění aplikace se načte obsah adresáře audio (načtou se pouze textové soubory). Všechny nalezené soubory budou zobrazeny ve scéně Otevření audio lekce viz obr. 5.1. Každý textový soubor obsahuje popisek lekce, data výukové části a data slovní zásoby. Pro zpřehlednění se popisek lekce bude zobrazovat i ve scéně Otevření audio lekce spolu s názvem souboru. Po výběru audio lekce se otevřou i zbylé dva soubory, které obsahují obrázek a audio záznam (propojení je dáno názvem souborů – všechny tři soubory každé audio lekce mají stejný název, liší se pouze příponou).

Struktura textového souboru jedné lekce modulu Audio

soubor	význam
…	popisek / název lekce
#1	oddělovač
…	data výukové části
#2	oddělovač
…	data použité slovní zásoby

* zobrazované tečky (…) značí uložená data v textovém formátu viz ukázková lekce systému

6 Realizace prototypu

Aplikace je určená pro operační systém Windows XP. Pro realizaci prototypu jsem použil aplikace: MinGW – Minimalist GNU for Windows – kolekce hlavičkových souborů a knihoven pro Windows včetně překladače c, c++ nachází se na adrese [12], ConTEXT – textový editor určený pro programátory – umožňuje zvýraznění syntaxe několika programovacích jazyků s možností rozšíření pomocí pluginů na adrese [13] a FMOD – sada knihoven a nástrojů pro ovládání audio výstupu různých formátů viz [6].

Aplikace je naprogramovaná v objektovém programovacím jazyce c++. Interface aplikace, který je závislý na použitém operačním systému (Windows) je naprogramován ve WinAPI (Windows Aplication Interface). Grafický výstup je zajištěn pomocí knihovny OGL – Open Graphic Library.

Při realizaci prototypu jsem si aplikaci rozdělil na několik částí (modulů), kde každý vykonává určitou činnost (vytvoření aplikačního rozhraní, zobrazování grafiky, přehrávání zvuku). Zdrojové texty aplikace jsou tedy rozděleny do několika souborů. Ke každému zdrojovému textu *.cpp přísluší hlavičkový soubor se stejným jménem a příponou *.hpp. Všechny moduly jsou navrženy univerzálně a proto je možné je beze změny použít při vytváření dalších aplikací, kde je třeba zvukového výstupu a zobrazování 3D grafiky.

main.cpp	hlavní zdrojové texty aplikace – určují chování a reakce aplikace na různé podněty od uživatele. Zde je umístěn kód specifický pouze pro aplikaci Lingua-3D (ostatní moduly jsou navrhnuty univerzálně a jsou znovupoužitelné pro vývoj dalších aplikací)
TAudioPlayer.cpp	modul, který zajišťuje přehrávání zvukových nahrávek různých formátů
TInterface.cpp	aplikační interface – vytváření oken, zpracování smyčky zpráv atd. Zde je umístěn všechen kód, který je závislý na operačním systému Windows
TOGL.cpp	modul pro zobrazování 3D grafiky pomoci OGL

6.1 Aplikační rozhraní

Při realizaci aplikačního rozhraní jsem používal online nápovědu společnosti Microsoft na [7] a přednášky předmětu SCS viz přílohy na CD-ROM.

Nejdůležitější částí aplikace bylo vytvoření jejího rozhraní. Aplikační rozhraní je prostředníkem mezi operačním systémem, uživatelem a samotnou aplikací. Tato část programu je závislá na použitém operačním systému a je tedy velice vhodné, pro přenositelnost aplikace na jiný operační systém, oddělit zdrojové kódy aplikačního rozhraní od ostatních částí aplikace. Při přenosu aplikace na jiný systém stačí přeprogramovat pouze tuto část aplikace. Zdrojové texty jsou uloženy v souboru TInterface.cpp

Aplikační rozhraní zajišťuje vytváření oken, změnu nastavení zobrazovacího adapteru, vytváření fontů, vytváření časovačů, přijímání zpráv od uživatele (stisk klávesy, operace s myší), přijímání zpráv systému (zpráva od časovače, změně velikosti okna, atd.) a přijímání zpráv aplikace (zprávy generované v aplikaci – např. ukončení aplikace). Všechny části aplikačního rozhraní jsou naprogramovány objektově (lze vytvářet několik instancí dané třídy – např. vytvoření několika oken aplikace s různými atributy).

6.1.1 Vytváření oken

Před vytvořením okna je nutné zaregistrovat třídu okna. Proto jsem vytvořil třídu TWindowClass s následujícími atributy a metodami:

Třída okna

_string	className	unikátní jméno pro každou třídu
HBRUSH	background	určuje pozadí klientské části okna
HICON	hIcon	ikona aplikace – definována v ressource souboru: resource.rc
HCURSOR	hCursor	kurzor – definován v souboru resource.rc
LPCTSTR	lpszMenuName	menu aplikace – definováno v resource.rc
HINSTANCE	hInstance	unikátní identifikátor aplikace (číslo integer)
bool	registered	určuje, zda je třída oken v systému zaregistrována
int	windowCount	počet oken zaregistrovaných ke třídě
metody:		
int	Register()	zaregistruje třídu
int	UnRegister()	zruší registraci třídy

Díky objektovému návrhu můžeme zaregistrovat několik tříd s různými parametry. Ke třídě okna lze zaregistrovat jedno nebo více oken, které budou mít vlastnosti definované touto třídou. Důležitou součástí třídy okna je funkce nazývající se Procedura Okna WindowProc(). Tato funkce je

volána při každém výskytu zprávy, která náleží oknu naší aplikace. V této funkci je definováno, jak se bude na danou zprávu reagovat. Zprávy, které nebyli nijak zpracovány jsou předány zpět systému, který je zpracuje automaticky.

Nejčastěji používané zprávy:

WM_CREATE	vytváření okna
WM_CLOSE	zavírání okna
WM_PAINT	překreslení okna
WM_TIMER	zpráva od časovače
WM_KEYDOWN	stisk klávesy
WM_KEYUP	uvolnění klávesy
WM_LBUTTONDOWN	stisk levého tl. myši
WM_MOUSEMOVE	pohyb myší

Pro vytvoření okna jsem vytvořil třídu TWindow, která má následující atributy a metody. Opět je možno vytvořit jedno nebo více oken díky objektovému návrhu aplikace.

Okno

_string	caption	popisek okna umístěn v horní liště aplikace
int	left, top	umístění levého horního rohu okna vzhledem k pracovní ploše
int	width, height	šířka a výška okna v pixelech
int	bits	barevná hloubka použité palety barev – 16, 24, 32 (v závislosti na použité grafické kartě)
HWND	hWnd	unikátní identifikátor každého okna
HDC	hDC	device kontext – kontext zařízení – plocha v paměti kam se vykresluje obsah okna
HGLRC	hRC	rendering kontext – renderovací kontext OGL
TWindowClass	windowClass	ukazatel na třídu, kam je okno zaregistrováno
DWORD	windowStyle	styl okna – určuje, jaké bude mít okno okraje, zda bude zobrazeno nad ostatními okny atd.
metody:		
int	Create()	vytvoří okno se zadanými atributy
void	Destroy()	zruší okno
void	Show()	zobrazí okno

Okno

metody:		
void	Hide()	schová okno
void	SetShape()	nastaví tvar okna (obdélník, elipsa, ovál)
void	SetCaption()	nastavení popisku okna
void	SetLeft()	nastavení vzdálenosti od levého okraje obrazovky
void	SetTop()	nastavení vzdálenosti od horního okraje obrazovky
void	SetWidth()	nastavení šířka okna
void	SetHeight()	nastavení výšky okna
void	SetCentered()	vycentrování okna vzhledem k obrazovce

Třída Twindow dále obsahuje ukazatel na strukturu klávesnice a myši, kde se ukládají stisknuté klávesy a akce provedené myší. Při stisku klávesy systém pošle zprávu o stisku klávesy (WM_KEYDOWN) do fronty zpráv. Aplikace, pro níž je zpráva určená, vyzvedne zprávu z fronty a předá ji proceduře okna. Zde je zpráva zpracována a jsou provedeny předem definované úkony. V našem případě je zaznamenáno, která klávesa byla stisknuta nebo uvolněna; v případě myši je zaznamenáno, které tlačítko bylo stisknuto / uvolněno a jaká je pozice kurzoru.

Struktura klávesnice pro uložení informací o stisknuté klávese

```
typedef struct {
    bool            key[256];
    char            code;
    bool            extended;
} KEYBOARD;
```

Po stisku klávesy se ve smyčce zpráv objeví zpráva WM_KEYDOWN. Zpráva obsahuje 2 parametry, z nichž jeden udává kód stisknuté klávesy a druhý zdali byla stisknuta standardní nebo funkční klávesa (kurzorové šipky, F1, F2, …).

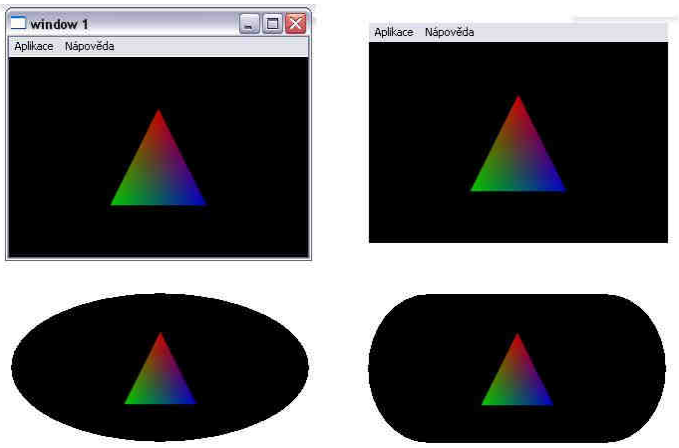
V poli key se nastaví na hodnotu TRUE všechny pozice, které odpovídají právě stisknutým klávesám (je možno zaznamenat stisk několika kláves současně), proměnná code obsahuje kód naposled stisknuté klávesy a proměnná extended určuje, zda je stisknuta standardní nebo funkční klávesa.

Struktura myši

```
typedef struct {
    int             x;
    int             y;
    int             button;
} MOUSE;
```

Do proměnných x a y se ukládá pozice kurzoru vzhledem ke klientské části okna. Proměnná button uchovává informaci o stavu tlačítek myši.

Třída okna TWindow má k dispozici metody pro nastavení velikosti, tvaru a pozice okna. Okno může byt vytvořeno s ohraničením, popiskem a ovládacími prvky pro zavření, minimalizaci a maximalizaci okna nebo může být vytvořena pouze klientská část bez ostatních komponent viz obr. 6.1. Při vytváření okna je aktivováno vyhlazování obrazu 4x (anti-aliasing). Pokud grafická karta nedisponuje tímto rozšířením je okno vytvořeno standardně bez aktivace vyhlazování obrazu.



obr. 6.1 – tvary a ohraničení okna

6.1.1.1 Časovač

Každé okno aplikace obsahuje jeden časovač (TTimer). Je určen pro události, které se opakují v daném časovém intervalu (překreslování okna, animace, nastavení atributů, …).

Funkce časovače spočívá v posílání zprávy WM_TIMER do smyčky zpráv, ve chvíli, kdy vyprší jeho časový interval. Časovač vždy přísluší k některému oknu aplikace a je identifikován unikátním číslem. Při vytváření časovače nastavujeme časový interval v milisekundách, okno ke kterému časovač přísluší a identifikátor unikátní pro každý časovač.

Obecně je možno vytvořit několik časovačů současně. Velké množství časovačů s malým časovým intervalem však může způsobit zaplnění smyčky zpráv. Z toho důvodu jsem se v aplikaci omezil pouze na jeden časovač, který je nastaven na konstantní hodnotu 20ms (zopakuje se tedy 50-krát za sekundu; je to dostatečně malý interval pro vytváření animací a menší interval by zbytečně plýtval výkonem počítače). Pro události, které se opakují v delších časových intervalech se musí spočítat, kolik cyklů časovače musí proběhnout, aby se daná událost provedla (např. pro časový interval 2s proběhne 100 cyklů časovače). Výhodou jednoho časovače je malá zátěž smyčky zpráv. Nevýhodou je přesnost vypočtu delších intervalů, které nejsou násobkem 20ms (jelikož přesnost časovače není v naší aplikaci potřebná, můžeme tuto nevýhodu zanedbat).

6.1.1.2 Fonty

S vytvářením oken je spojena i tvorba fontů. Vykreslování grafiky sice obstarává OGL, ale generování znakové sady je spojeno se systémem a je zařízeno pomocí WinAPI. Standard OGL obsahuje základní 2D fonty bez znaků s českou diakritikou (obsahuje pouze základní znaky první poloviny ASCII tabulky bez speciálních znaků) – nelze s nimi rotovat nebo pohybovat v prostoru. Proto jsem v aplikaci vytvořil vlastní fonty. Každé okno aplikace generuje vlastní znakovou sadu (ve vektorovém formátu). Je vytvořena z fontů, které jsou součástí operačního systému Windows (Times New Roman, Arial, …). Všechny systémové fonty najdeme ve složce 'Fonts' (START→Control Panel→Fonts). Tyto fonty posloužily jako šablona. Byly z nich vygenerovány trojrozměrné fonty – můžeme s nimi tedy pohybovat v prostoru, rotovat a nastavovat hloubku. Aplikace používá fonty Courier New, Times New Roman, Arial a Webdings.

6.2 Vykreslování grafiky

Při vytváření modulu pro vykreslování grafiky jsem čerpal z online tutoriálu zabývajícím se 3D OGL grafikou na adrese [9]. Všechny zdrojové texty vykreslující 3D grafiku jsou umístěny v souboru TOGL.cpp.

Prostředí aplikace obsahuje 3D grafické prvky (editační pole, textová pole, tlačítka, nápisy, obrázky a další grafické prvky). Modul TOGL.cpp se nezabývá pouze vykreslováním 3D grafiky, ale i chováním a reakcí grafických prvků na různé události (kliknutí myší, stisknutí tlačítka klávesnice, vypršení časového intervalu, …). Každý grafický prvek je vlastně komponenta s atributy (pozice ve scéně, natočení, …) a metodami, které určují typ reakce komponenty na různé události. Všechny komponenty spravuje grafický manažér, který přijímá zprávy od okna aplikace (aplikačního rozhraní). Má tedy přehled o tom, jaký je stav klávesnice, myši, atd.

Komponenta

float	x, y, z	pozice komponenty ve scéně
float	a, b, c	natočení komponenty ve scéně
metody:		
virtual void	Behaviour()	zde je pro každou komponentu zvlášť definováno chování / reakce na různé události – stisk definované klávesy, činnost s myší, …

V komponentě je definováno pouze chování na různé události. Každá komponenta se skládá z jednoho nebo více objektů. Například komponenta tlačítko se skládá ze dvou objektů. Je to tělo tlačítka a jeho popisek.

Objekt

int	id	unikátní identifikátor objektu
float	x, y, z, a, b, c	pozice a natočení objektu vzhledem ke komponentě
bool	selected	určuje, zda je nad objektem kurzor myši
bool	aktive	určuje zdali je objekt aktivní
bool	visible	určuje, zda se objekt zobrazuje
GLuint	texture	ukazatel na texturu
int	transparency	průhlednost objektu
metody:		
virtual void	onPaint()	vykreslovací funkce
virtual void	BuildList()	sestavení display-listů

V každém objektu je definován jeho vzhled (tvar, barva, textura, průhlednost), pozice a natočení vzhledem ke komponentě. Atribut selected určuje, zda se nad objektem nachází kurzor myši. Dalším atributem je aktivita objektu. Při kliknutí kurzorem myši na objekt, se stává objekt aktivním a přijímá vstup z klávesnice. Ostatní objekty ve scéně jsou pasivní a vstup z klávesnice ignorují. Důležitou metodou každého objektu je sestavovací funkce BuildList(), kde je definován vzhled objektu a probíhá načtení textur. Textury jsou obrázky, které se umísťují na povrch objektů. Jejich načtení se provádí pomocí funkce LoadGLTextures(). Jelikož se ve scéně může vyskytovat několik stejných komponent skládajících se ze stejných objektu (tlačítka, editační boxy, …) je každý objekt (jeho vizuální část) sestavena v paměti pouze jednou pomocí funkce BuildList() a všechny objekty stejného typu se potom odkazují při vykreslování na stejné místo v paměti. Urychlí se tím značně vykreslování a sníží požadavky na paměť.

Grafický manažér

TWindow	window	ukazatel na okno, do kterého se vykresluje
metody:		
void	Select()	funkce pro detekci polohy kurzoru

Grafický manažér (ToglScene) má za úkol správu všech komponent a jejich objektů ve scéně. Obsahuje ukazatel na okno, ve kterém se scéna vykresluje. Je prostředníkem mezi oknem aplikace a grafickou scénou. Může reagovat na některé zprávy určené pro okno. Obsahuje metodu Select(), která má za úkol detekci grafických objektů kurzorem myši. Metoda funguje tak, že nejprve přepne OpenGL do "selection" módu pomocí příkazu glRenderMode(GL_SELECT). V tomto režimu se nic nevykresluje na obrazovku, ale informace o renderovaných objektech se uloží do "selection bufferu" (vymezené místo v paměti). Potom voláním funkce glInitNames() inicializujeme "name stack" (paměť jmen), kde je každý grafický objekt identifikován unikátním číslem (jménem). Dalším krokem je omezení velikosti oblasti, do které se vykresluje. Omezíme ji na malou oblast pod kurzorem myši. Zjistíme, které objekty se nacházejí v této oblasti a získané informace uložíme do "selection bufferu". V "selection bufferu" jsou tedy uložena jména všech detekovaných objektů a hloubka v jaké se nacházejí ve scéně. Najdeme objekt, který je nejblíže k pozorovateli (nejmenší hloubka) a nastavíme jeho proměnnou selected na hodnotu TRUE. Posledním příkazem glRenderMode(GL_RENDER) přepneme OpenGL zpět do renderovacího módu, takže se opět budou grafické objekty vykreslovat na scénu.

Množina zpráv na které reaguje grafický manažér

onPaint()	požadavek na překreslení
onTimer()	vypršel časový limit časovače
onActivate()	při změně aktivity okna
onKeyDown()	stisk tlačítka
onKeyUp()	uvolnění tlačítka
onMouseWheel()	pohyb s kolečkem myši

6.2.1 Komponenty

V aplikaci Lingua-3D jsem si vytvořil několik komponent pro různé činnosti. Nejdůležitější komponenty jsou tlačítko (ToglButton), textové pole (ToglLabel), editační pole (ToglEdit) a obrázek (ToglPicture). Další komponentou jsou například hodiny v hlavním menu aplikace viz obr. 6.2. Všechny komponenty mají označení "ogl", protože jsou vykreslovány pomocí OpenGL. Reagují na

různé podněty – např. tlačítko se vysune do strany, když na něj najedeme kurzorem myši. Mají různé vlastnosti (parametry) – např. u textového pole můžeme definovat zarovnání textu do bloku, na střed, doleva nebo doprava. Každá komponenta je navržena objektově, tudíž můžeme vytvářet několik instancí dané komponenty.

obr. 6.2 – hodiny

Hodiny slouží v aplikaci spíše jen jako dekorativní prvek. Ukážeme si na nich jak pracují všechny komponenty v systému. Každá komponenta se skládá z jednoho nebo více objektů. Komponenta hodin se skládá ze čtyř objektů. Je to sekundová ručička, minutová ručička, hodinová ručička a tělo hodin. Dále komponenta obsahuje jeden časovač a s ním spojenou metodu onTimer() a metodu pro definování chování komponenty Behaviour().

Časovač hodin slouží k opakovanému zjišťování systémového času. Ten zjistíme pomocí funkce GetLocalTime(). Existuje ještě jedna podobná funkce GetSystemTime(), ale ta vrací čas světový (Greenwich Mean Time – GMT) a tudíž v našich zeměpisných šířkách o hodinu menší v zimě (zimní čas u nás je GMT+1) a o dvě hodiny menší v létě (letní čas u nás je GMT+2). Funkce GetLocalTime() vrací čas pásma, ve kterém se nacházíme s ohledem na letní i zimní čas.

Následující metoda onTimer() je pravidelně volána při vypršení časového limitu časovače a nastavuje ručičky hodin podle zjištěného systémového času. Přesněji řečeno se nastavuje úhel ručiček hodin, pod kterým se budou ve scéně vykreslovat.

```
void ToglAnalogClock::onTimer()
{
  if (timer->Elapse()) {
    GetLocalTime(time);
    if (time->wHour > 12) time->wHour = time->wHour - 12;
    second->c = -(time->wSecond * 6.0);
    minute->c = -(time->wMinute * 6.0);
    hour->c = -(time->wHour * 30.0 + time->wMinute * 0.5);
  }
}
```

Metoda pro definování chování komponenty obsahuje obvykle akce, které se mají provézt na základě nějakého podnětu. U hodin nejsou definovány žádné další akce, ale např. komponenta tlačítka obsahuje v metodě Behaviour() akci vysunutí tlačítka do strany a přehrání zvukového efektu při najetí kurzoru myši na tělo tlačítka.

```
void ToglButtonRight::Behaviour()
{
  if (board->selected)
  {
    if (x > -4.0) x -= 0.5; if (z > -22.0) z -= 0.25;
    if (play)
      scene->window->audioPlayer->Play("button_selected");
    play = false;
  }
  else
  {
    x = 0.0; z = -20.0; play = true;
  }

  if (board->active)
  {
    scene->window->audioPlayer->Play("button_active");
  }
}
```

6.3 Přehrávání zvukových nahrávek

Součástí aplikace je modul pro přehrávání zvukových nahrávek. Použil jsem zvukovou knihovnu FMOD [6]. Je to množina nástrojů (hlavičkové soubory, knihovny) pro detekci zvukové karty a ovládání jejího výstupu. Zdrojové kódy jsou v souboru TAudioPlayer.cpp.

Vytvořil jsem zvukový přehrávač, který je schopen otevřít a přehrát zvukové nahrávky několika formátů. Nejčastěji používané formáty jsou *.wav, *.mp3 a *.midi. Přehrávač má velice jednoduché rozhraní. Pomocí funkce Open() otevřeme všechny požadované zvukové soubory.

Ukládají se do tzv. play-listu, což je seznam všech otevřených skladeb. Pomocí ostatních funkcí můžeme skladby spouštět, zastavovat, měnit jejich hlasitost nebo zjišťovat o nich podrobné informace. Přehrávač je naprogramován objektově a tak můžeme vytvářet několik jeho instancí.

Přehrávač obsahuje vyhledávání skladeb podle indexu nebo názvu souboru. Vyhledávání podle názvu souboru se využívá v aplikaci Lingua-3D při přehrávání výslovnosti výrazů. Audio nahrávky jsou uloženy v adresáři /dictionary/words. Název každého souboru odpovídá nahranému výrazu. Při spuštění aplikace se načtou všechny výrazy do play-listu, odkud jsou pak při vybrání výrazu přehrány.

Zvukový přehrávač

int	NowPlaying	index právě přehrávaného souboru
metody:		
bool	Open()	otevře hudební soubor
bool	Close()	zavře hudební soubor
bool	Play()	přehraje hudební soubor
bool	Stop()	zastaví přehrávání
bool	Pause()	pozastaví / spustí přehrávání
void	Prev()	přehrání předchozí skladby
void	Next()	přehrání následující skladby
bool	Rewind()	skok ve skladbě vpřed / zpět
int	GetLength()	vrátí délku skladby v ms
int	GetSongCount()	vrátí počet otevřených skladeb
TSongInfo	GetSongInfo()	vrátí informace o souboru (v případě *.mp3 načte informace z ID3 tagu)
bool	SetSoundPos()	nastaví pozici zvuku ve 3D prostoru
void	SetVolume()	nastaví úroveň hlasitosti právě přehrávané skladby

6.4 Hlavní program

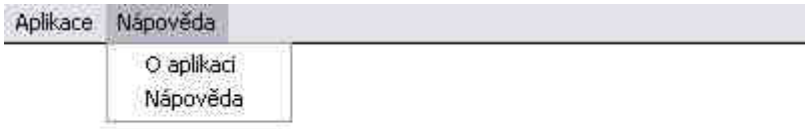
Jádro aplikace je soustředěno v modulu s názvem main.cpp. Zde je část programu specifická pouze pro aplikaci Lingua-3D (kód ostatních modulů je možno beze změny použít při vytváření dalších aplikací). Při vytváření modulu jsem čerpal z Microsoft online nápovědy [7] a publikací [3], [4] a [5].

Aplikace vytvoří jedno okno přes celou obrazovku bez ohraničení do kterého se vykreslují všechny OGL scény. Je vytvořeno sedm 3D grafických scén (hlavní menu, načítání slovníků, výuková část slovníků, testovací část slovníků 1, testovací část slovníků 2, načítání audio lekcí a výuková část audio lekcí). Hlavní program funguje jako automat a vždy může být pouze v jednom stavu. Počáteční stav je scéna hlavní menu.

Okno hlavního programu

TMainMenuScene	scéna hlavní menu
TDictionaryLoadScene	scéna načítání slovníků
TDictionaryEx1Scene	scéna výuková část slovníků
TDictionaryEx2Scene	scéna testovací část 1
TDictionaryEx3Scene	scéna testovací část 2
TAudioLoadScene	scéna načítání audio lekcí
TAudioScene	výuková část audio lekcí
metody:	
onPaint()	překreslení okna
onClose()	zavření okna
onTimer()	zpráva časovače
onCommand()	zpráva o výběru položky v menu
onMouseMove()	pohyb myši
onMouseDown()	stisk tlačítka myši
onMouseWheel()	otočení kolečka myši
onKeyDown()	stisk tlačítka klávesnice

V záhlaví aplikace je umístěn úzký pruh s nabídkou pro ukončení aplikace a nápovědou (obr. 6.1 Menu Aplikace). Toto menu je dostupné ve všech scénách. Jeho struktura je uložena v souboru zdrojů ressource.rc a hlavičkovém souboru ressource.h (v těchto souborech je definována i ikona aplikace a kurzorová šipka)



obr. 6.1 Menu aplikace

6.5 Zkušenosti

Při realizaci výukového systému jsem získal nové zkušenosti s objektovým programovacím jazykem c++ a zlepšil své dovednosti při programování s grafickou knihovnou OpenGL. Naučil jsem se programovat aplikační rozhraní systému Microsoft Windows pomocí funkcí WinAPI. Seznámil jsem se s funkcemi knihovny FMOD Ex pro ovládání audio výstupu. Aplikace, které využívají tuto knihovnu lze při nekomerčním využití distribuovat zdarma. Při programování jsem nepoužil žádné vývojové prostředí z důvodů, abych si vyzkoušel realizaci aplikace na systémové úrovni. Všechny komponenty jsem sám navrhl a vytvořil.

Aplikační rozhraní WinAPI oproti knihovně GLUT (platformě nezávislá knihovna pro tvoření aplikačního rozhraní s podporou OGL viz kapitola 2) nabízí úplnou kontrolu nad systémem a aplikací. Pomocí knihovny GLUT nelze přistupovat ke smyčce zpráv. Reagovat na události můžeme pouze pomocí tzv. ”call back” funkcí (funkce, které volá knihovna GLUT při výskytu definovaných zpráv). Při použití knihovny GLUT ve svém projektu bych nemohl reagovat např. na příchozí zprávu od kolečka myši (při rolování stránky s textem), protože neexistuje žádná “call back” funkce pro tuto činnost. Při použití knihovny GLUT není k dispozici “handle” okna (ukazatel na okno) nebo kontext okna do kterého se vykresluje. Neměl bych pak možnost realizovat některé funkce systému jako vytváření 3D písma z True Type fontů systému Windows nebo aktivovat vyhlazování obrazu (multi-sampling).

Pro načítání dat výukových lekcí jsem chtěl použít funkce z minulé verze aplikace ročníkového projektu. Zjistil jsem však, že jsou již zastaralé a byly implementovány pouze ve vývojovém prostředí C++ Builder z důvodu zpětné kompatibility pro starší programy. Jedná se o funkce, které mají načíst obsah adresáře. Jsou to funkce findfirst() a findnext(). Jejich parametry jsou cesta k požadovanému adresáři, datová struktura, kam se ukládají informace o nalezených souborech a parametr, kterým se určuje, jaký typ souboru se má hledat (zahrnuje i typ “adresář“). Nové verze těchto funkcí jsou findfirst () a findnext(). Mají pouze dva parametry – cestu k požadovanému adresáři a datovou strukturu s informacemi o nalezených souborech. Datová struktura je oproti minulé verzi vylepšená a obsahuje navíc položku, která určuje typ nalezeného souboru. Z toho plyne, že funkce vyhledá všechny typy souborů i s adresáři. Třídění podle typu probíhá až následně. V následující části programu vypíšeme obsah požadovaného adresáře na obrazovku. Vynechávají se položky, u kterých je atribut typu adresář.

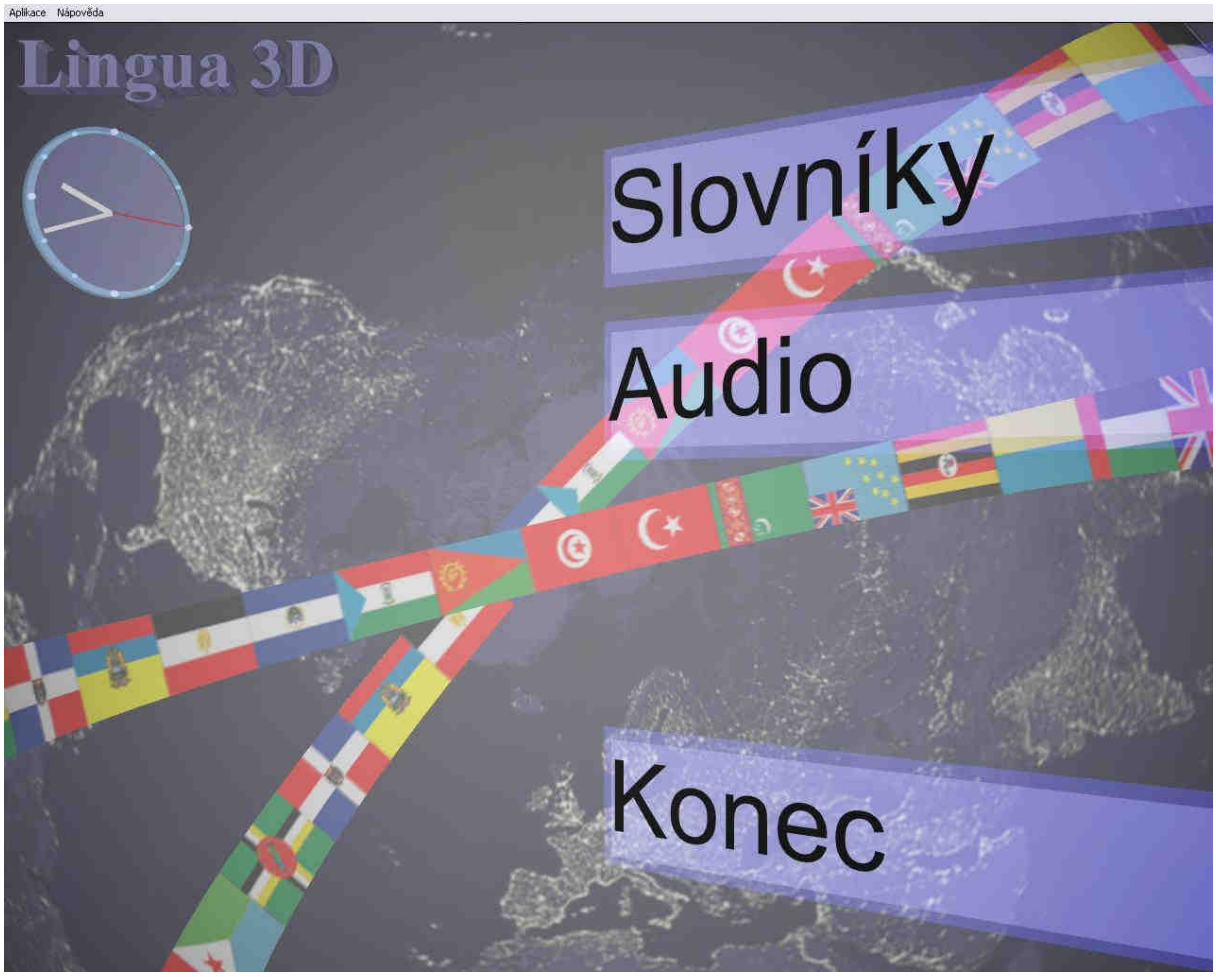
```
int done;
  struct _finddata_t data;
  char fullpath[];

  done = _findfirst (fullpath, &data);
  if (done == -1) return;

  do {
    if (data.attrib != _A_SUBDIR)
    {
      printf("%s\n", data.name);
    }
  } while (_findnext(done, &data) == 0);

  _findclose(done);
}
```

Při testování systému jsem zjistil, že aplikace zabírá velké množství operační paměti. Konkrétně je to 115 MB paměti RAM. Je to způsobeno neoptimalizovaným vytvářením textur z obrázků typu *.bmp. Při deaktivaci načítání textur klesla hodnota využití paměti RAM na 25 MB. Tato hodnota je již přijatelná. Tento problém by se dal řešit optimalizací funkcí zabývajícími se otevřením obrázků a vytvářením textur. V paměti by se měla vytvářet vždy pouze jedna instance každé textury. Textury by se měly tvořit až ve chvíli, kdy jsou potřebné. Pro zrychlení reakcí systému vytvářím všechny textury při spuštění aplikace a paměť uvolňuji až při ukončení (to způsobuje velké vytížení paměti). Další možnou optimalizací je vytváření textur z obrázků jiného formátu než je *.bmp, např. nahráváním komprimovaných obrázků formátu *.tga viz [9].

7 Demonstrace systému

Aplikace Lingua-3D má poutavý trojrozměrný graficky vzhled. Důraz byl kladen na jednoduchost a přehlednost systému, a proto aplikace obsahuje malé množství ovládacích prvků a všechny grafické objekty mají velké rozměry. Pro zlepšení přehlednosti a čitelnosti je použita větší velikost textů.

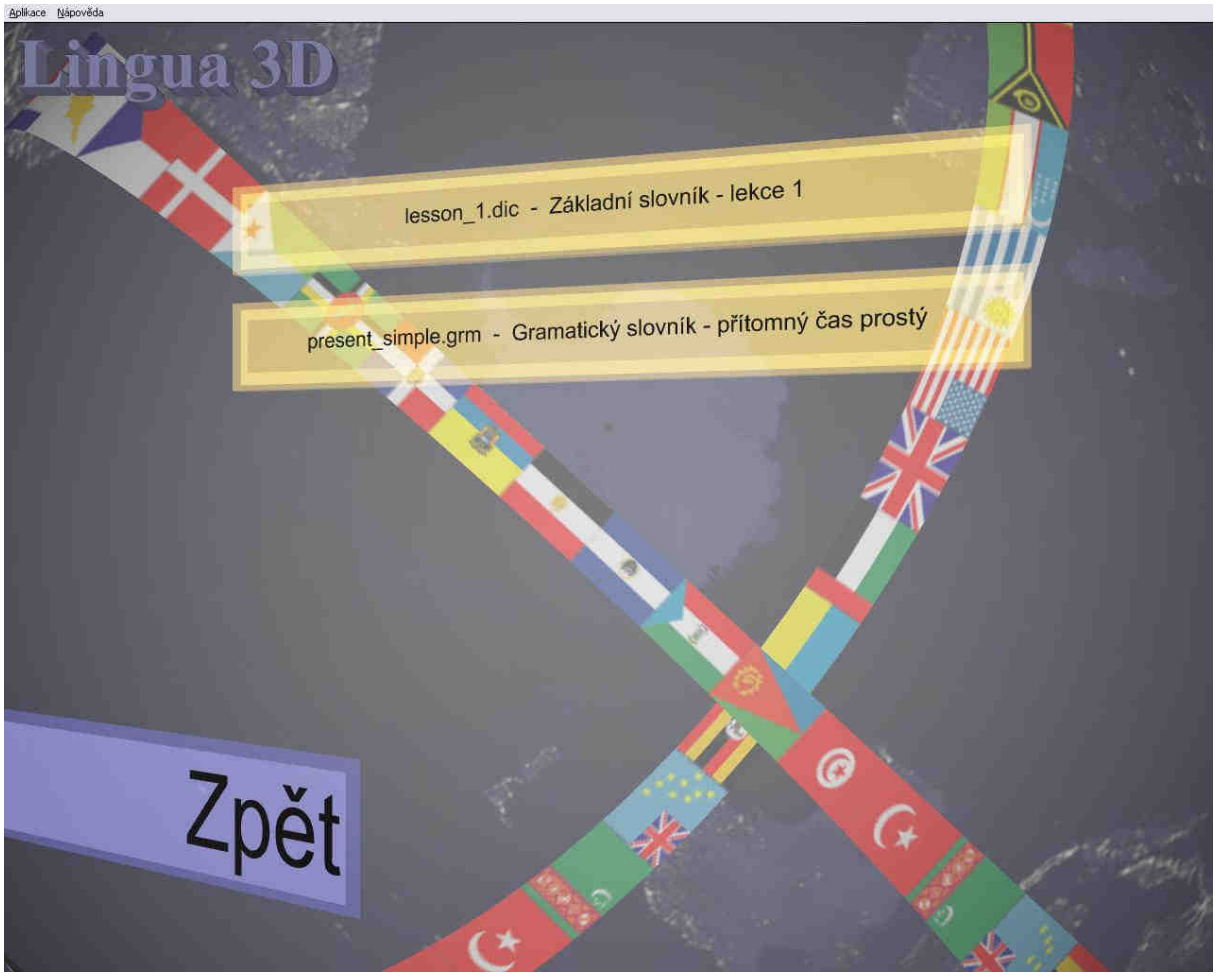
Po spuštění aplikace se nacházíme v hlavním menu viz obr. 7.1 Hlavní menu. Zde je nabídka výukových modulu. Jsou to moduly **Slovníky** a **Audio**. Kliknutím myší na modul se nám zobrazí všechny dostupné lekce daného modulu. Tlačítkem **Konec** se zobrazí dialog pro ukončení aplikace viz obr. 7.2 Konec (aplikaci lze ukončit kdykoli klávesou Esc).



obr. 7.1 Hlavní menu

obr. 7.2 Konec

Po výběru modulu **Slovníky** se nám zobrazí všechny dostupné lekce tohoto modulu viz obr. 7.3 Načítání slovníku. U každé lekce je zobrazen název souboru, ve kterém je obsah lekce uložen (např. lesson_1.dic) a název lekce (např. Základní slovník – lekce 1). Slovníky, které obsahují pouze slovní zásobu mají příponu *.dic. Slovníky s příponou *.grm jsou gramatické slovníky – obsahují výuku gramatiky. Při kliknutí na tlačítko **Zpět** se dostaneme do hlavního menu. Po výběru modulu **Audio** se nám zobrazí všechny dostupné audio lekce. Zobrazení a výběr lekcí probíhá stejně jako u modulu Slovníky.

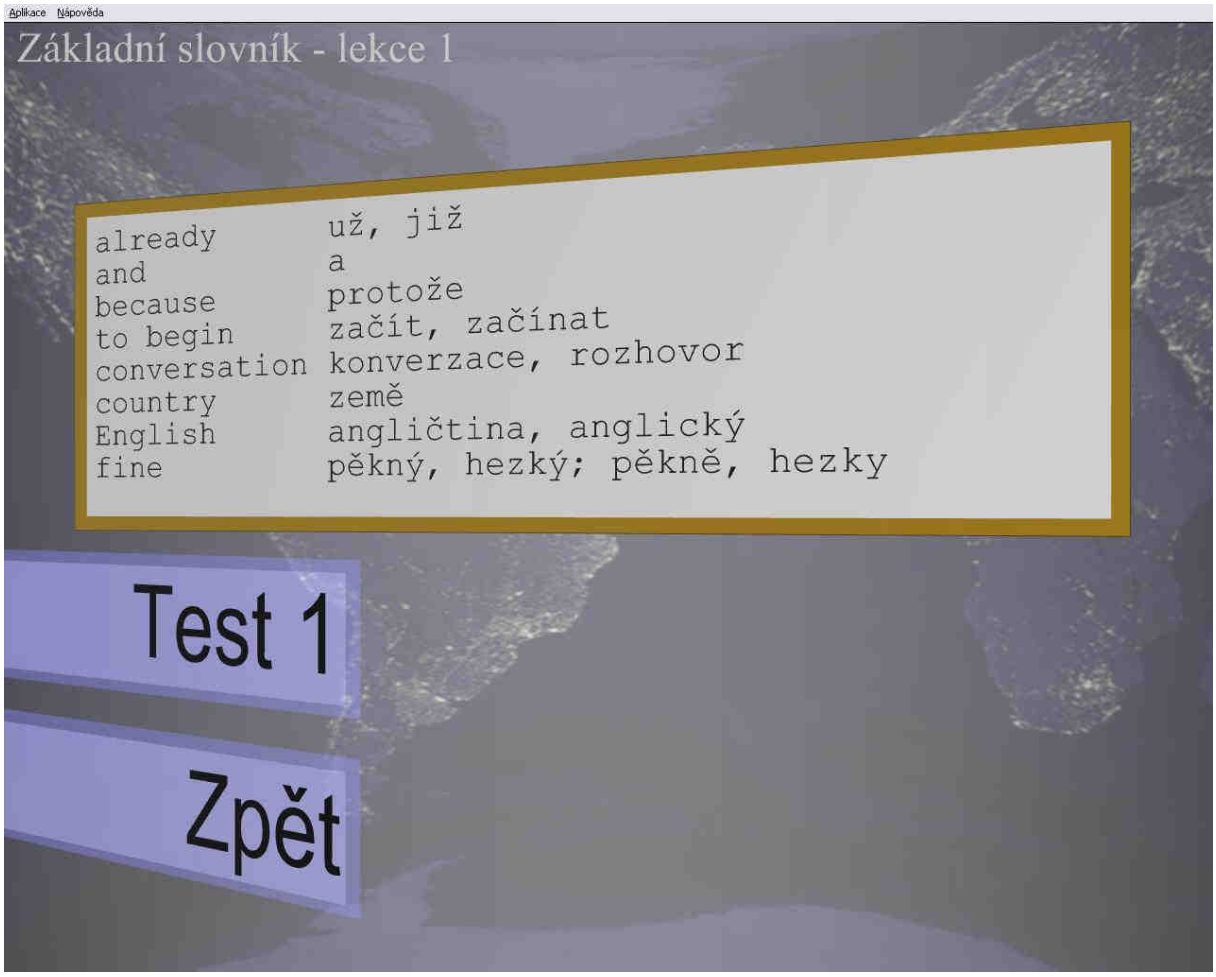


obr. 7.3 Načítání slovníků

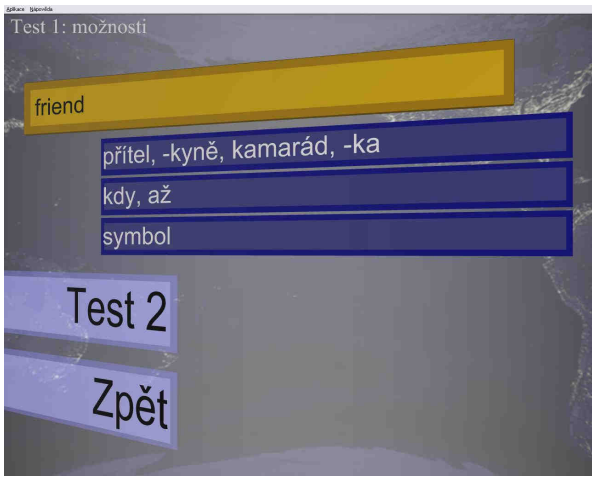
7.1 Slovníky

Po výběru slovníku se spustí výuková část. V případě slovníku na výuku slovní zásoby se zobrazí slovníček viz obr. 7.4 Výuková část slovníku. Zde se uživatel seznámí se všemi slovními výrazy dané lekce. Procvičování probíhá v následujících dvou testech. První test obr. 7.5a má lehkou obtížnost a je zaměřen na výslovnost. Ve žlutém poli je vždy zobrazen překládaný výraz. Při kliknutí do žlutého pole je přehrána jeho výslovnost. Níže jsou zobrazena tři modrá pole. Každé obsahuje možný překlad daného výrazu, ale pouze jedna možnost je vždy správná. Při výběru špatného překladu je přehrán výstražný zvuk. Po výběru správné možnosti je náhodně vybrán další slovní výraz. Test není nijak omezen a neustále se opakuje. Uživatel si tak sám může určit, kdy je vhodné přistoupit k druhému testu viz obr. 7.5b. Obtížnost druhého testu je vyšší. Ve žlutém poli je překládaný výraz. Jeho překlad se doplňuje do zeleného editačního pole. Při správném zadání je náhodně vybrán další výraz. V opačném případě je správná odpověď zobrazena a je přehrán výstražný zvuk. Test se opět neomezeně opakuje a tak záleží pouze na uživateli, kdy uzná za vhodné s testem skončit.

Gramatické slovníky mají stejnou skladbu jako slovníky na výuku slovní zásoby. Opět je zde část výuková a dvě části testové. Ve výukové části viz obr. 7.6 je vysvětlena určitá část gramatiky (např. použití přítomného času prostého v anglickém jazyce). Následující testové časti obsahují slovní výrazy nebo věty k procvičení viz obr. 7.7a, 7.7b.

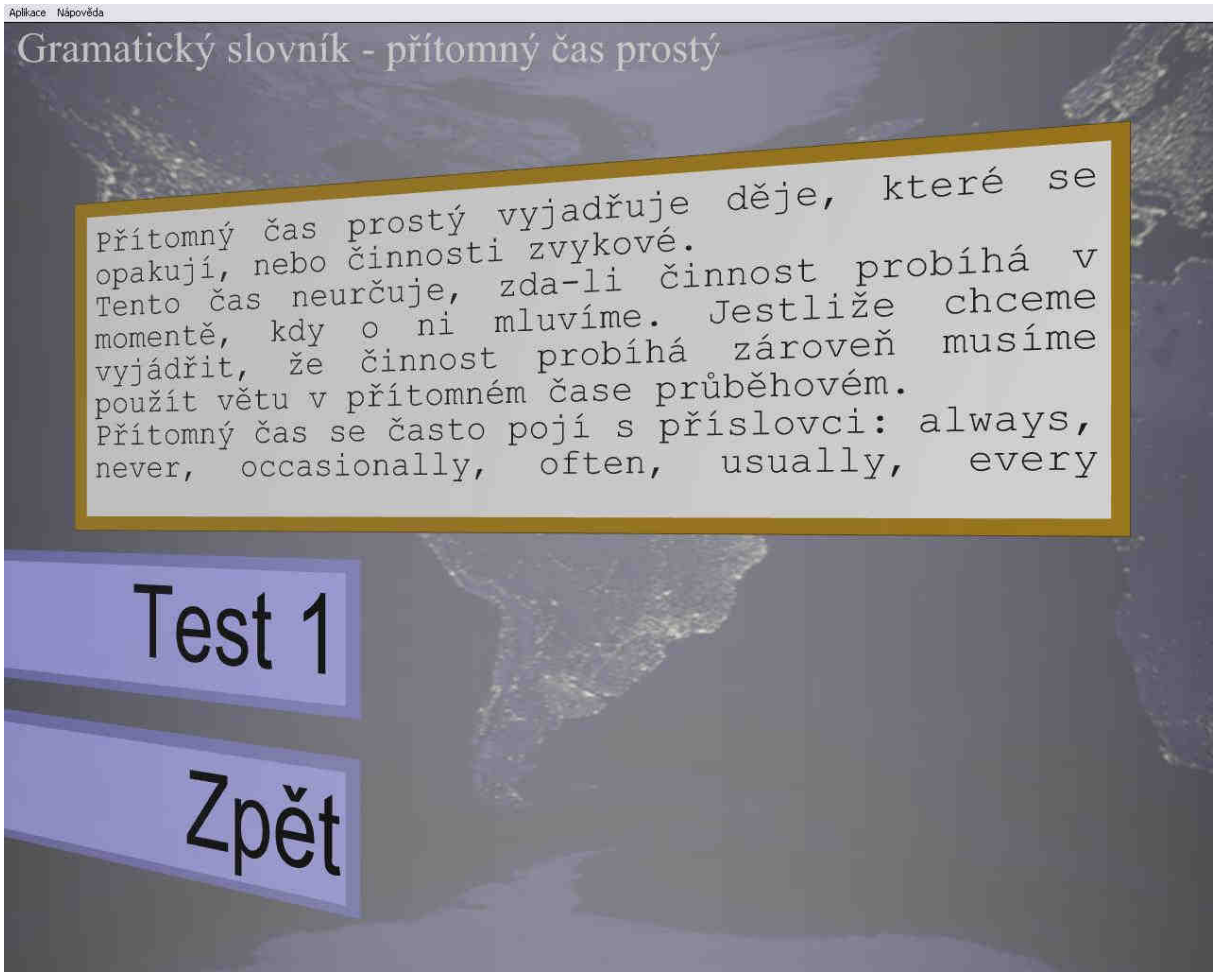


obr. 7.4 Výuková část slovníku – slovní zásoba

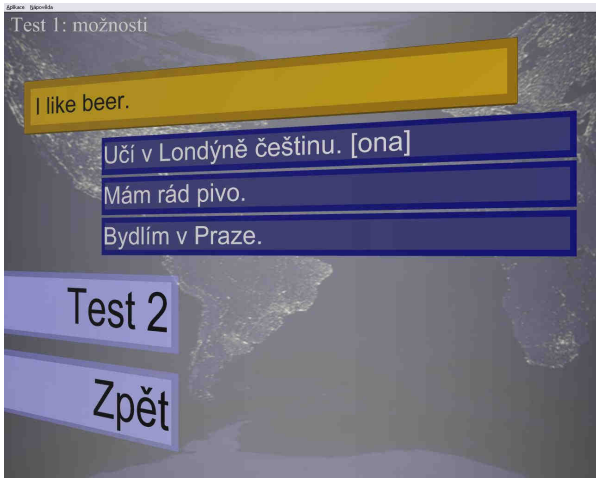


obr. 7.5a Testovací část 1

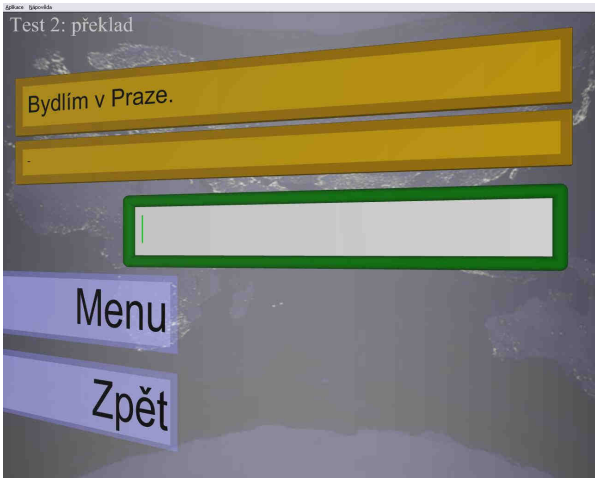
obr. 7.5b Testovací část 2



obr. 7.6 Výuková část slovníku – gramatický slovník



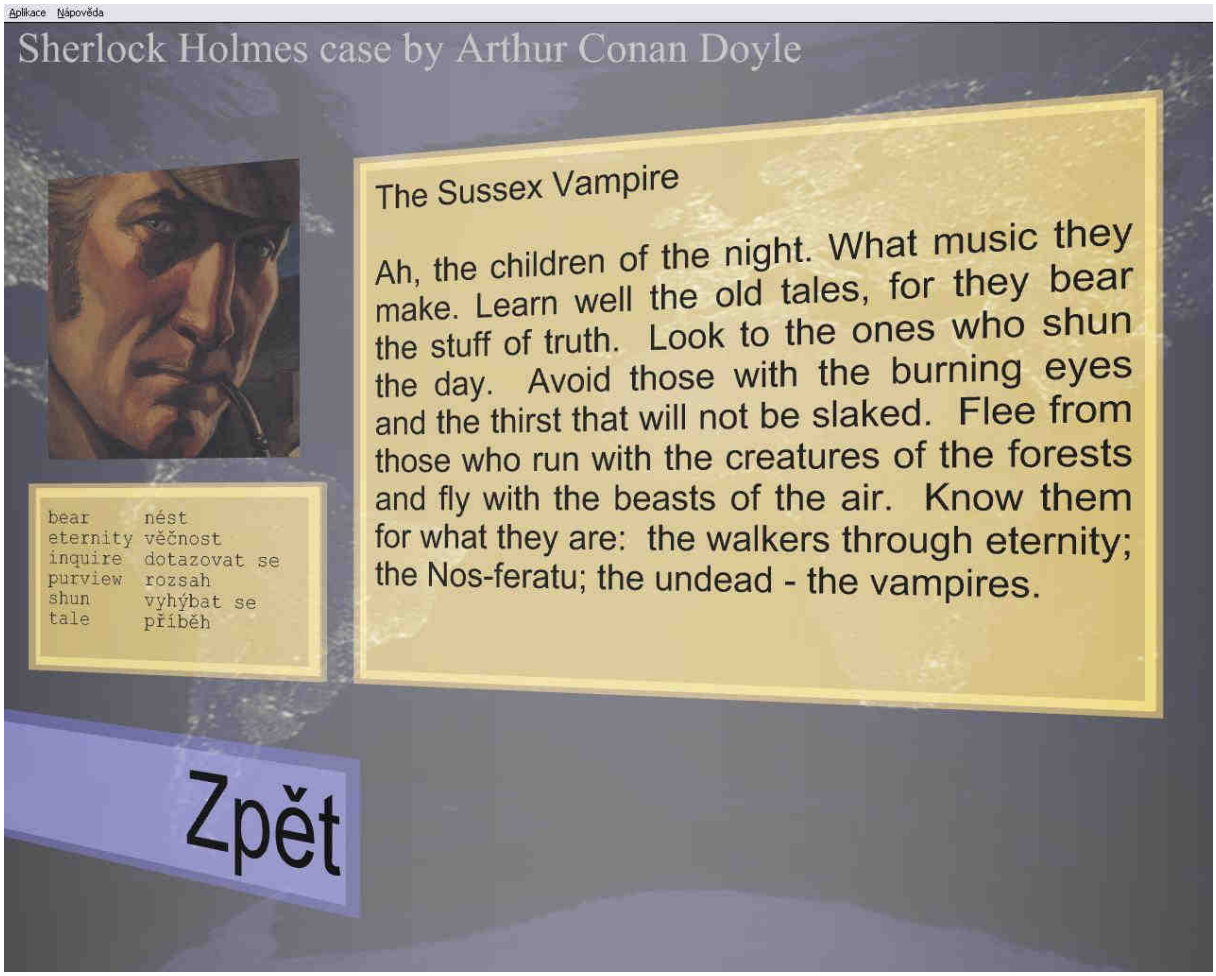
obr. 7.7a Testovací část 1



obr. 7.7b Testovací část 2

7.2 Audio

Modul Audio obsahuje lekce zaměřené na procvičení poslechu a četby. Každá lekce obsahuje povídku, příběh nebo článek. K dispozici je text i audio záznam. Audio modul obsahuje pouze výukovou část viz obr. 7.8 Audio lekce. Po spuštění lekce se nám zobrazí dvě žlutá textová pole a spustí se přehrávání audio záznamu. Větší textové pole obsahuje text příběhu nebo povídky. Menší textové pole obsahuje slovní zásobu použitou v článku. Tlačítkem **Zpět** se dostaneme do nabídky audio lekcí.



obr. 7.8 Výuková část Audio lekce

7.3 Editování

Program Lingua-3D nepodporuje editování lekcí nebo vytváření nových lekcí přímo (nelze ovlivnit v programu Lingua-3D). Aplikace při každém spuštění prohledává pevný disk a načítá seznam všech dostupných lekcí, je tedy možné přidávat nebo editovat lekce manuálně v adresáři programu (k editaci postačí běžné nástroje operačního systému Windows). Nástroje k editaci dat a jejich struktura uložení jsou popsány v kapitole 5.2 Uložení dat.

7.4 Demonstrovaná data

Demonstrovaná data jsem získal z elektronického kurzu na adrese www.e-academy.cz a z internetového rádia http://www.bbc.co.uk. Tento materiál je použit se souhlasem autorů pouze pro účely tohoto projektu.

7.5 Instalace

Na počítači, kde se bude aplikace provozovat musí být nainstalován operační systém Microsoft Windows XP. Dále by měly být správně nainstalovány a nastaveny ovladače grafického a zvukového adapteru. Ovladač grafického adaptéru by měl být nastaven tak, aby zajistil podporu standardu OGL.

Instalace probíhá pouze nakopírováním obsahu adresáře [./aplikace] z CD-ROM do požadovaného adresáře na počítači. Program lze spustit i přímo z přiloženého CD-ROM.

7.5.1 Systémové požadavky

- operační systém: Microsoft Windows XP
- potřebné místo v paměti RAM: 115 MB
- potřebné místo na HD: 8 MB
- standardní graficky adaptér pro zobrazení 3D grafiky
- standardní audio adaptér

8 Závěr

V diplomové práci jsem se zabýval problematikou elektronického vzdělávání. Je to jedna z mnoha výukových metod jak lze poskytovat nové znalosti. Jedná se o velice sofistikovanou metodu využívající dnešní moderní výpočetní techniku.

Cílem teoretické části práce bylo objasnění pojmu e-learning a důkladné prostudování problematiky elektronického vzdělávání. Došel jsem k závěru, že elektronické vzdělávání nenahrazuje současné vzdělávací prostředky. Je to pouze další alternativa vzdělávání vhodná jak pro samouky, tak pro skupiny lidí. Jeho hlavní síla je ve schopnosti přenášet a prezentovat multimediální obsah. Nevýhoda spočívá v izolování žáka od učitele. Součástí teoretické přípravy bylo zaměření se na uplatnění 3D grafiky, využití audio záznamů v elektronickém vzdělávání a seznámení s aplikačním rozhraním operačního systému Microsoft Windows. Tato problematika byla již částečně řešena v ročníkovém projektu. Získané informace jsem zahrnul i do diplomové práce, protože jsou nezbytné pro další části projektu.

Cílem projektu bylo navrhnout a realizovat systém pro podporu multimediální výuky jazyků. Vytvořil jsem prototyp systému s názvem Lingua-3D. Jedná se o statickou aplikaci vytvořenou v programovacím jazyce c++. Aplikace je navrhnuta pro operační systém Microsoft Windows.

Systém obsahuje dva výukové moduly. Modul Slovníky je zaměřen na výuku slovní zásoby, gramatiky a výslovnosti. Modul Audio je zaměřen na poslech a četbu. Oba moduly obsahují lekce pro výuku anglického jazyka. Jako další rozšíření aplikace by bylo možné implementovat editor lekcí, grafické zobrazení výsledků testů formou grafů, přidat databázi uživatelů nebo další výukové moduly (např. modul kde by výuka probíhala formou zábavné hry). Po programátorské stránce by bylo vhodné provézt optimalizaci systému s ohledem na snížení využití operační paměti RAM.

V této diplomové práci jsem získal nové znalosti v oblasti návrhu, realizace a distribuce elektronických kurzů. Zlepšil jsem své programátorské schopnosti v objektovém jazyce c++ a grafické knihovně OpenGL. Seznámil jsem se s aplikačním rozhraním Microsoft Windows.

Literatura

[1] Horton, W.: Design Web-Based Training: How to Teach Anyone Anything Anywhere Anytime, Wiley; 1st edition, ISBN: 047135614X

[2] Horton, W.: E-learning Tools and Technologies, John Wiley & Sons; 1st edition (January 10, 2003), ISBN: 0471444588

[3] Eckel, Bruce: Myslíme v C++, Grada; Knihovna programátora, první vydání, ISBN: 80-247-9009-2

[4] Herout, Pavel: Učebnice jazyka C, Kopp; Třetí upravené vydání, ISBN: 80-85828-21-9

[5] Virius, Miroslav: Od C k C++, Kopp; ISBN: 80-7232-110-2

[6] knihovna FMOD/FMOD Ex pro přehrávání audio obsahu, 11.04.2007, http://www.fmod.org/

[7] nápověda pro Windows API, 11.04.2007, http://msdn2.microsoft.com/en-us/default.aspx

[8] programování 3D grafiky pod knihovnou OpenGL, 11.04.2007 http://www.opengl.org/

[9] OpenGL tutoriál, 11.04.2007, http://nehe.ceskehry.cz/

[10] články (operační systémy, grafika), 11.04.2007, http://www.root.cz/

[11] články (programování ve Windows API), 11.04.2007, http://www.builder.cz/

[12] překladač c++, 11.04.2007, http://www.mingw.org/

[13] textový editor se zvýrazněním syntaxe, 11.04.2007, http://www.context.cx/

Seznam příloh

Příloha 1. CD – R obsahuje:

- Aplikaci Lingua-3D
- Kompletní zdrojové texty aplikace Lingua-3D
- Informační soubory install.txt a readme.txt
- Dokumentaci diplomové práce ve formátech *.doc a *.pdf
- Prezentaci diplomové práce ve formátech *.pdf a *.ppt
- Přednášky předmětu SCS (Styk člověk-stroj) – tvorba uživatelských rozhraní

Adresářová struktura přiloženého CD:

[\]
    install.txt
    readme.txt
[\aplikace] - aplikace Lingua-3D
[\aplikace\source] - soubory obsahující zdrojové kódy aplikace
[\dokumentace] - dokumentace a prezentace diplomové práce
[\prilohy] - ostatní přílohy