

# **Pokročilé techniky v ORACLE – úložiště multimediálních dat**

Advanced techniques in ORACLE – multimedia data store

Bc. Michal Trúnek

---

Diplomová práce  
2013



Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky

---

Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky  
akademický rok: 2012/2013

## **ZADÁNÍ DIPLOMOVÉ PRÁCE**

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Michal Trúnek**  
Osobní číslo: **A11449**  
Studijní program: **N3902 Inženýrská informatika**  
Studijní obor: **Počítačové a komunikační systémy**  
Forma studia: **prezenční**

Téma práce: **Pokročilé techniky v ORACLE – úložiště  
multimediálních dat**

Zásady pro vypracování:

- 1. Analyzujte problematiku a zpracujte literární rešerši na dané téma.**
- 2. Navrhněte a vytvořte UML model aplikace.**
- 3. Popište možnosti využití dostupných nástrojů.**
- 4. Vytvořte aplikaci dle navrženého modelu.**
- 5. Popište proces tvorby aplikace a následné možnosti její využití.**
- 6. Věnujte pozornost zabezpečení aplikace.**

Rozsah diplomové práce:

Rozsah příloh:

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

Seznam odborné literatury:

1. RIORDAN, R. M. Vytváříme relační databázové aplikace. Praha : Computer Press, 2000. 280 s. ISBN 80-7226-360-9.
2. ZEMEK, Lukáš. Bezpečnost webových aplikací. Praha, 2012. bakalářská práce (Bc.). Univerzita Tomáše Bati ve Zlíně. Fakulta aplikované informatiky.
3. MACH, Jakub. PHP pro úplné začátečníky. 2., přeprac. a rozš. vyd. Brno: Computer Press, 2003, 167 s. ISBN 8072268341.
4. PROCHÁZKA, David. Oracle: průvodce správou, využitím a programováním nad databázovým systémem. 1. vyd. Praha: Grada, 2009, 168 s. ISBN 978-80-247-2762-2.
5. WELLING, Luke a Laura THOMSON. PHP a MySQL: rozvoj webových aplikací. 2. vyd. Praha: SoftPress, c2004, 910 s. ISBN 8086497607.
6. DAWSON, Alexander. Výjimečný webdesign: jak tvořit osobité, přitažlivé, použitelné weby. 1. vyd. Brno: Computer Press, 2012, 344 s. ISBN 978-80-251-3719-2.
7. PROCHÁZKA, David. SEO: cesta k propagaci vlastního webu. 1. vyd. Praha: Grada, 2012, 144 s. ISBN 978-80-247-4222-9.

Vedoucí diplomové práce:

**doc. Ing. Zdenka Prokopová, CSc.**

Ústav počítačových a komunikačních systémů

Datum zadání diplomové práce:

**26. února 2013**

Termín odevzdání diplomové práce:

**31. května 2013**

Ve Zlíně dne 26. února 2013

prof. Ing. Vladimír Vašek, CSc.  
*děkan*



prof. Ing. Karel Vlček, CSc.  
*ředitel ústavu*

## ABSTRAKT

Cieľom tejto diplomovej práce bolo vytvoriť webovú aplikáciu, ktorá pracuje s pokročilými funkciami databázového systému Oracle a využíva jej špeciálne nástroje pre spracovanie obrazových a zvukových súborov. Zvukové súbory sú prezentované vo forme audio prehrávača, implementovaného priamo vo webovej aplikácii. Pre tvorbu aplikácie bol využitý skriptovací jazyk PHP. Jednotlivé procedúry pre spracovanie obrazových a zvukových dát boli vytvorené v jazyku PL/SQL. Na projekte spolupracovali dvaja ľudia, pričom každý z nich mal určené špecifické zadanie, ktoré na seba priamo naväzovali.

Klíčová slova:

Oracle, InterMedia, PHP, PL/SQL, audio, databáza, webová aplikácia, prehrávač

## ABSTRACT

The aim of this thesis was to create a web application, which works well with the advanced features of Oracle database and uses this special tools for processing video and audio files. Audio files are presented in the form of audio player, which is embedded directly in a web application. There has been used scripting language PHP for programming every script in this web application. Individual procedures for processing audio and video data were created in the language PL/SQL. Two people were cooperated on this project, each of whom had specific assignment, which directly followed up on each other.

Keywords:

Oracle, InterMedia, PHP, PL/SQL, audio, database, web application, audio player

Týmto by som chcel poďakovať doc. Ing. Zdenke Prokopovej, CSc. za cenné rady, pripomienky a čas, ktorý mi venovala pri odbornom vedení tejto práce.

**Prohlašuji, že**

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

**Prohlašuji,**

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

Ve Zlíně

.....  
podpis diplomanta

**OBSAH**

<b>ÚVOD</b> .....	<b>10</b>
<b>I TEORETICKÁ ČÁST</b> .....	<b>11</b>
<b>1 VÝVOJ SOFTVÉRU</b> .....	<b>12</b>
1.1 SCHÉMATICKÉ VYJADRENIE.....	12
1.2 SOFTVÉROVÝ PROCES .....	14
1.2.1 Základné typy .....	14
1.2.2 Unified Process (UP).....	15
1.2.2.1 Pracovné postupy iterácie .....	16
<b>2 UML</b> .....	<b>17</b>
2.1 ŠTRUKTÚRA JAZYKA UML.....	17
2.1.1 Stavebné bloky .....	17
2.1.2 Spoločné mechanizmy.....	18
2.1.3 Architektúra.....	18
2.2 POŽIADAVKY .....	18
2.2.1 Definícia požiadavkov.....	19
2.3 USE CASE - MODEL PRÍPADOV POUŽITIA .....	20
2.3.1 Modelovanie prípadov použitia.....	20
2.3.2 Komplexné prípady použitia – Scenár .....	20
2.4 ANALYTICKÉ TRIEDY .....	21
2.4.1 Analýza podstatných mien a slovies.....	21
2.4.2 Metóda CRC.....	22
2.5 SEKVENČNÉ DIAGRAMY.....	22
2.5.1 Iterácia a vetvenie.....	22
2.6 AKTIVITNÉ DIAGRAMY .....	23
<b>3 ZABEZPEČENIE WEBOVÝCH APLIKÁCIÍ</b> .....	<b>24</b>
3.1 SQL INJECTION .....	25
3.1.1 Priebeh SQL injection útoku .....	25
3.1.1.1 Prieskum aplikácie .....	25
3.1.1.2 Testovanie náchylnosti na chyby .....	25
3.1.1.3 Útok .....	25
3.1.2 Ako sa brániť.....	26
3.1.2.1 Escapovanie vstupných dát .....	26
3.1.2.2 Kontrola dátového typu.....	26
3.1.2.3 Framework .....	26
3.2 CROSS-SITE SCRIPTING .....	27
3.2.1 Ako sa brániť.....	28
3.2.1.1 Escapovanie .....	28
3.2.1.2 Filtrovanie .....	29
<b>4 ORACLE</b> .....	<b>30</b>

4.1	ÚLOŽISKO DÁT .....	30
4.2	ORACLE INTERMEDIA .....	30
4.2.1	Architektúra.....	31
4.2.2	Oracle Call Interface (OCI).....	32
4.2.2.1	Výhody.....	32
4.3	ŠPECIÁLNE MULTIMEDIÁLNE NÁSTROJE.....	33
4.3.1	Objektový typ – ORDImage.....	33
4.3.1.1	Komponenty.....	33
4.3.1.2	Metadáta.....	34
4.3.1.3	Úprava obrazových dát .....	34
4.3.2	Objektový typ – ORDVideo.....	34
4.3.2.1	Komponenty.....	34
4.3.3	Objektový typ – ORDAudio.....	35
4.3.3.1	Komponenty.....	35
4.3.4	Streamovanie obsahu z Oracle databázy .....	35
4.3.4.1	Oracle Multimedia Plug-in pre RealNetworks streamovací server .....	35
<b>II</b>	<b>PRAKTICKÁ ČÁST .....</b>	<b>36</b>
<b>5</b>	<b>ZÁSADY PRI TVORBE WEBOVEJ APLIKÁCIE .....</b>	<b>37</b>
5.1	VÝVOJOVÉ NÁSTROJE.....	37
5.1.1	SQL a PL/SQL .....	37
5.1.2	HTML.....	38
5.2	TVORBA DESIGNU WEBOVÝCH STRÁNOK .....	38
5.2.1	CSS.....	38
5.2.2	Adobe Photoshop .....	38
<b>6</b>	<b>DOTAZNÍKOVÝ PRIESKUM.....</b>	<b>39</b>
<b>7</b>	<b>MODELOVANIE ŠTRUKTÚRY APLIKÁCIE V UML.....</b>	<b>40</b>
7.1	ÚVODNÁ ŠTÚDIA .....	41
7.2	POŽIADAVKY .....	42
7.2.1	Funkčné požiadavky.....	42
7.2.2	Nefunkčné požiadavky.....	45
7.3	USE CASE DIAGRAM .....	45
7.3.1	Aktéri systému.....	45
7.3.2	Hranice systému .....	46
7.3.3	Prípady užitia (Use Case).....	46
7.4	DIAGRAM AKTIVÍT .....	48
7.5	DIAGRAM TRIED .....	50
7.6	SEKVENČNÉ DIAGRAMY.....	51
<b>8</b>	<b>WEBOVÁ APLIKÁCIA.....</b>	<b>52</b>
8.1	SPRACOVANIE AUDIO SÚBOROV .....	52
8.1.1	Príprava databázy .....	53
8.1.1.1	Špeciálne procedúry a funkcie .....	54
8.1.2	Upload audio súborov .....	55

8.1.2.1	Konfigurácia PHP .....	56
8.1.3	Spracovanie vlastností zvukových súborov .....	56
8.2	AUDIO PREHRÁVAČ .....	57
8.2.1	Načítanie demo nahrávok.....	57
8.2.2	Ovládacie prvky .....	58
8.3	ORGANIZÁCIA KONCERTOV .....	59
<b>9</b>	<b>OŠETRENIE A ZABEZPEČENIE ZDROJOVÉHO KÓDU WEBOVEJ APLIKÁCIE .....</b>	<b>60</b>
9.1	KONFIGURÁCIA PHP.INI.....	60
9.2	KONFIGURÁCIA WEBOVÉHO SERVERA APACHE (.HTACCESS).....	61
9.2.1	SEO a clean URL .....	61
9.3	CROSS-SITE SCRIPTING (XSS).....	62
9.3.1	Zabezpečenie z pohľadu užívateľa.....	64
9.4	SQL INJECTION .....	64
	<b>ZÁVER .....</b>	<b>66</b>
	<b>ZÁVER V ANGLIČTINE.....</b>	<b>67</b>
	<b>ZOZNAM POUŽITEJ LITERATÚRY .....</b>	<b>68</b>
	<b>ZOZNAM POUŽITÝCH SYMBOLOV A ZKRATIEK.....</b>	<b>69</b>
	<b>ZOZNAM OBRÁZKOV .....</b>	<b>70</b>
	<b>ZOZNAM TABULIEK .....</b>	<b>72</b>
	<b>ZOZNAM PRÍLOH.....</b>	<b>73</b>

## ÚVOD

Databázové systémy ako také existujú už niekoľko desiatok rokov a postupne od ich vzniku si získavajú čoraz väčší vplyv v rôznych odvetviach. V dnešnej dobe sa do popredia dostávajú databázy, ktoré sú schopné spracovávať a ukladať nielen klasické dáta ale aj multimediálne súbory. Hlavnou databázou, ktorá umožňuje spracovávať multimediálne súbory, či už sú to obrazové, zvukové alebo video dáta, je databáza Oracle. Pomocou špeciálneho vývojového prostredia interMedia dokáže databáza ukladať, spravovať, a získavať multimediálne dáta a informácie, ktoré obsahujú.

V teoretickej časti obsahuje práca popis vývojových prostriedkov. V prvej kapitole je popísaný všeobecný postup pri tvorbe softvéru, schématické vyjadrenie a jednotlivé pracovné postupy a diagramy. Druhá kapitola obsahuje popis jazyka UML, pri ktorom sú popísané jednotlivé prístupy k analýze, návrhu a popisu tvorby jednotlivých funkcií softvéru. Tretia kapitola je venovaná zabezpečeniu webovej aplikácie, kde sú popísané základné typy útokov a spôsoby ochrany proti nim. Posledná časť je zameraná na databázový systém Oracle. Popísané sú špeciálne nástroje interMedia ale aj rozširujúce funkcie a možnosti, ktoré databáza Oracle ponúka.

Praktická časť je venovaná samotnému vývoju a modelovaniu štruktúry webovej aplikácie pomocou jazyka UML. Popísané sú všetky využité diagramy a postupy, ktoré boli použité pri návrhu. V ďalšej časti je popísaná tvorba a využitie vybraných špeciálnych funkcií webovej aplikácie. Podstatnou časťou je spracovanie audio súborov databázou Oracle a ich následné použitie vo webovej aplikácii. Audio súbory sú prezentované vo forme audio prehrávača, ktorý je umiestnený priamo v aplikácii. Záverečná časť práce je venovaná zabezpečeniu zdrojového kódu a obsahuje taktiež popísané vybrané časti ošetrovaného kódu.

## **I. TEORETICKÁ ČÁST**

## 1 VÝVOJ SOFTVÉRU

Vývoj softvérového systému zahrňuje celú škálu faktorov a bodov, ktoré vedú k úspešnému vytvoreniu navrhovaného projektu.

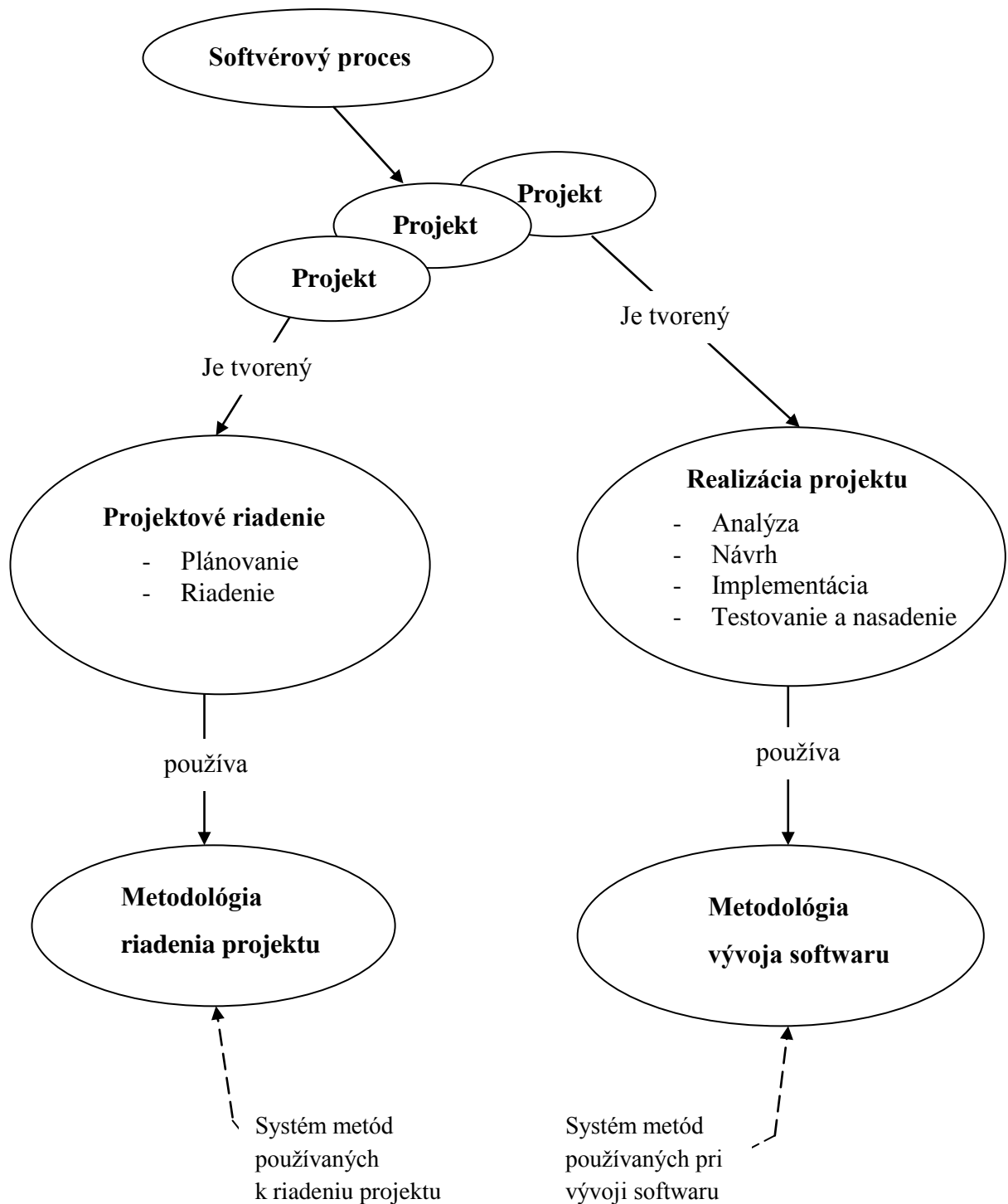
Technické aspekty patria medzi hlavné faktory, ktoré je pri navrhovaní softvéru potrebné zohľadniť. Zahrňujú hlavne počítačovú infraštruktúru a vybavenie softvéru. Naopak netechnické aspekty vyvíjajú daný produkt a zaoberajú sa viac menej ekonomickými možnosťami, ktoré sú taktiež nevyhnutné pri navrhovaní projektu.

Medzi ďalšie faktory, patrí taktiež odborná znalosť špecifických požiadavkov na daný produkt a požiadavky na daný softvér v očiach verejnosti alebo úzkej skupiny zákazníkov, pre ktorých bude tento produkt určený a ktorí budú tento produkt využívať či už pri svojej každodennej činnosti alebo prídu s ním do styku počas svojej práce. Jedná sa hlavne o produkty, ktoré poskytujú svoje služby širokej verejnosti. Verejný prieskum je jednou z foriem vyriešenia takéhoto problému. Väčšinou prebieha vo forme dotazníku, kedy sú širokej verejnosti poskytnuté otázky a navrhnuté spôsoby riešenia a prezentácie produktu. Účastník má taktiež možnosť navrhnúť vlastný spôsob alebo funkcionality, ktorú po preštudovaní a zistení dostupných prostriedkov na jeho realizáciu, programátor buď zahrnie alebo vylúči z možných riešení daného problému. Vyhodnotením dotazníku, získava programátor alebo navrhujúci projektu objektívny pohľad na daný problém a tým aj výhodu pri vytváraní a navrhovaní danej aplikácie.

### 1.1 Schématické vyjadrenie

Ako základ je stanovený softvérový proces, teda presne daný postup činností, ktoré vedú k vytvoreniu výsledného produktu. Podľa týchto presne stanovených postupov sa následne defunujú projekty (inštanciácia procesu). Každý projekt je tvorený svojou realizáciou, činnosťami viazanými na vývoj vlastného produktu a celkovým riadením celého projektu.

V prípade riadenia ide o metodológiu projektového riadenia, ktorá je daná systémom metód používaných v projektovom riadení. Na druhej strane metodológia vývoja softvéru je tvorená systémom metód používaných pri vývoji. [1]



Obrázok 1. Sémantický diagram vývoja softvérového produktu

## 1.2 Softvérový proces

„Softvérový proces je po častiach usporiadaná množina krokov smerujúcich k vytvoreniu alebo úprave softvérového diela.“ [1]

Krokom môže byť nejaká aktivita alebo samotný podproces. Aktivity alebo podprocesy môžu bežať nezávisle na sebe, takže je požadovaná koordinácia medzi týmito aktivitami. Dôležitá je opakovateľnosť, alebo znovupoužitie procesu, vo vzťahu k jednotlivým projektom. Cieľom je dosiahnutie vysokej úrovne kvality daného produktu.

Jednotlivé činnosti sú vykonávané ľuďmi, ktorí majú potrebné znalosti v danom obore a sú oboznámení s problematikou daného produktu. Softvérový produkt je realizovaný podľa nejakej osnovy s ekonomickými možnosťami a samotnou organizačnou štruktúrou.

### 1.2.1 Základné typy

Medzi základné typy softvérového procesu môžeme zaradiť **vodopádový model**. Tento model môžeme označiť ako základný, pretože sa jeho štruktúra vyskytuje vo viacerých modifikáciách a rozšíreniach, súčasť postupov a prístupov k riešeniu daného problému. Tento model vychádza z rozdelenia životného cyklu do štyroch hlavných fáz:

1. Analýza požiadavkov a ich špecifikácie
2. Samotný návrh softvérového systému
3. Implementácia návrhu do samotného kódu
4. Testovanie a udržovanie vytvoreného produktu.

Samotná filozofia tohto modelu vychádza z presne daných postupov jednotlivých bodov. Zjednodušene povedané ďalší bod môže nastať len po úspešnom dokončení predchádzajúcej fázy. Alebo inak, výstupy z predchádzajúcej fázy vývoja „vtekajú“ ako vstupy do nasledujúcej fázy vývoja.

Nedostatky, ktoré sa vyskytovali pri tomto modeli boli v priebehu niekoľkých rokov nahradzované a vylepšované aby tak mohli vzniknúť ďalšie modifikácie tohto typu. Medzi takéto typy môžeme zaradiť *inkrementálny model* postavený na princípe postupného vytvárania verzií systému zahrňujúcich stále širšiu škálu funkcií definovaných postupne

v priebehu celého procesu jeho vytvárania. Medzi ďalšie typy patrí napríklad aj špirálovitý model, ktorý zahrňuje vo svojom životnom cykle ďalšie fázy vývoja ako je vytvorenie a hodnotenie prototypu overujúceho funkcionality cieľového systému, pričom každý nanovo započatý cyklus priberá ďalšie požiadavky na cieľový produkt, ktoré stanovuje sám zadávateľ projektu.

### 1.2.2 Unified Process (UP)

Samotnou kapitolou pri tvorbe softvéru je metodika UP. Metodika UP je založená na metódach Ericsson (Ericsson approach, 1967), Rational (Rational Objectory Process, 1996-1997) a na ďalších zdrojoch, ktoré vychádzajú z najlepších postupov. Táto metóda zahňuje najlepšie overené postupy pri tvorbe softvéru. [2, s. 22]

Metodika UP je založená na návrhu a postupnom vývoji architektúry daného systému. Architektúra popisuje vzájomné vzťahy medzi jednotlivými komponentami a taktiež strategické aspekty ako sa môže daný systém rozdeliť na jednotlivé menšie komponenty. Metodika je iteratívna a prírastková (inkrementácia). Základná myšlienka je veľmi jednoduchá. Ako nám už história nespočetne krát ukázala, človeku sa lepšie a ľahšie riešia malé problémy ako problémy veľké. Inak povedané snažíme sa rozdeliť celý projekt na tzv. miniprojekty, ktoré riešime postupne a tým dotiahneme svoj veľký projekt do úspešného konca. Každý **miniprojekt je považovaný za iteráciu**.

Kľúčovým východiskom je skutočnosť že iterácia obsahuje **rovnaké prvky** ako softvérový projekt:

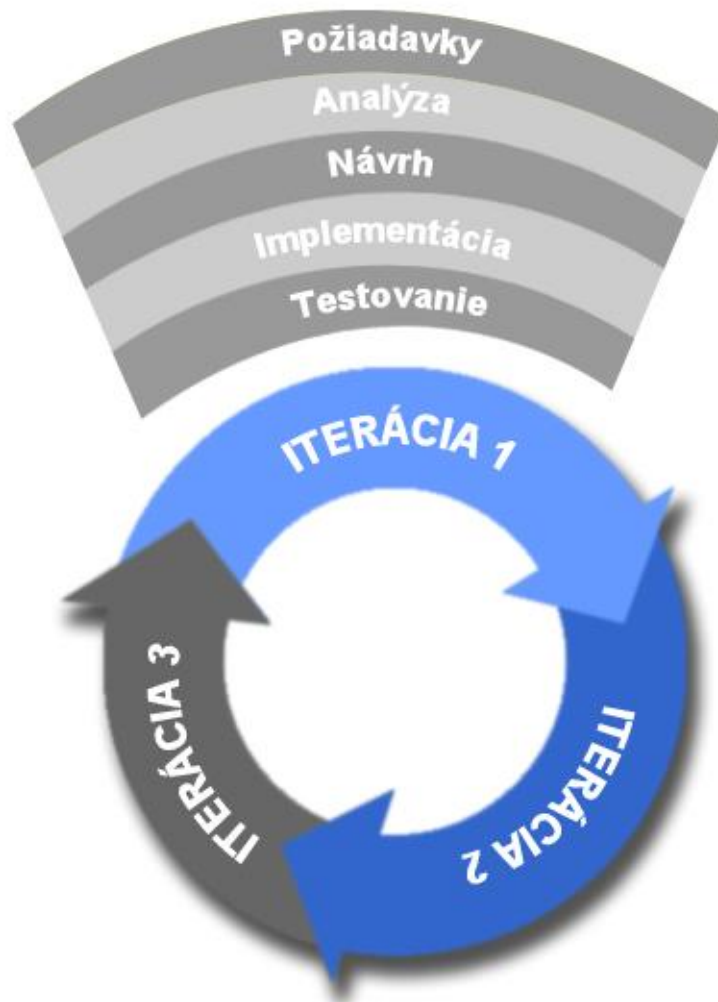
- Plánovanie
- Analýza a návrh
- Tvorba
- Integrácia a testovanie
- Interné alebo externé uvedenie

Každá iterácia generuje tzv. baseline alebo základnú líniu, ktorá sa skladá z čiastočne kompletnej verzie finálneho systému a z príslušnej dokumentácie daného čiastočného problému. Tieto línie sú postupne vrstvené až do finálnej podoby produktu. [2, s. 29]

### 1.2.2.1 Pracovné postupy iterácie

Každá iterácia obsahuje päť základných pracovných postupov tzv. workflows, ktoré určujú čo treba spraviť a akým spôsobom to dosiahneme.

- **Požiadavky** – určujú čoho by mal byť daný systém schopný
- **Analýza** – štrukturovanie požiadavkov a ich vylepšenie
- **Návrh** – Samotná realizácia požiadavkov v architektúre
- **Implementácia** – Tvorba software
- **Testovanie** – Overenie funkčnosti a správnosti implementácie



Obrázok 2. Jednotlivé pracovné postupy iterácie zobrazené do koláčového grafu

## 2 UML

Jazyk UML (unifikovaný modelovací jazyk) je univerzálnym jazykom pre vizuálne modelovanie systémov. Zložitý softvérový dizajn popísaný len samotným textom, môže byť veľmi zložitý na pochopenie a predstavenie, preto bol navrhnutý modelovací jazyk UML aby boli vaše programy / časti programov ľahko čitateľné pomocou diagramov.

Modelovanie ponúka **dve hlavné výhody**:

- Vizualizácia
- Jednoznačná komunikácia

Jazyk UML je navrhnutý tak, aby ho mohli implementovať všetky nástroje CASE (computer-aided software engineering). Diagramy vytvorené v jazyku UML sú zrozumiteľné pre ľudí, ale čo je hlavné, veľmi ľahko ich môžu interpretovať aj programy CASE. [2, s. 4]

### 2.1 Štruktúra jazyka UML

Štruktúra najlepšie odzrkadľuje funkciu jazyka ako jazyka vizuálneho. Skladá sa z týchto častí:

- Stavebné bloky
- Spoločné mechanizmy
- Architektúra

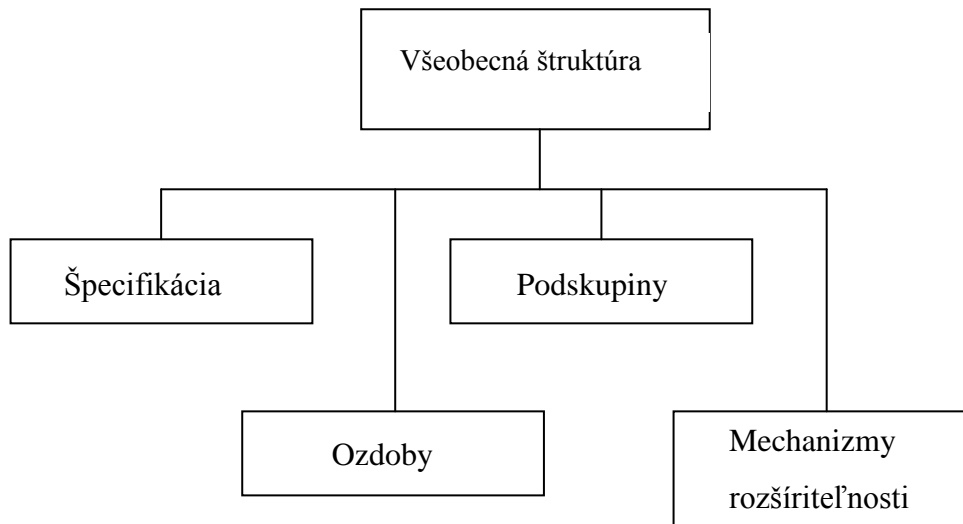
#### 2.1.1 Stavebné bloky

Jazyk je zostavený z troch stavebných blokov:

- **Predmety** (things), predstavujú samotné elementy modelu,
- **Vzt'ahy** (relationships), predstavujú prepojenia medzi jednotlivými predmetmi, určujú hlavne súvislosť a význam medzi dvomi predmetmi,
- **Diagramy** (diagrams), sú to pohľady na modely UML, predstavujú bloky predmetov, ale hlavne predstavujú vizualizáciu toho, čo systém bude robiť (analytické diagramy) a toho ako to bude robiť (návrhové diagramy).

### 2.1.2 Spoločné mechanizmy

Jazyk UML obsahuje štyri spoločné mechanizmy. Popisuje štyri základné stratégie cesty k modelovaniu objektov, ktorú sú používané v rôznych kontextoch v celom jazyku UML. [2, s. 11]



Obrázok 3. Všeobecná mechanika jazyka UML [2, s. 11]

### 2.1.3 Architektúra

Štandard IEEE definuje architektúru systému ako „najvyššiu úroveň koncepcie systému v jeho vlastnom prostredí“. Jazyk UML definuje štyri rôzne pohľady na systém:

- Logický pohľad
- Pohľad procesov
- Pohľad implementácie
- Pohľad nasadenia systému

Všetky hore uvedené pohľady sú integrované do piateho pohľadu, ktorým je **prípád užitia**.

## 2.2 Požiadavky

Určujú funkcionality vytváraného systému. Väčšina práce pri definícii a špecifikácii požiadavky prebieha hneď na počiatku celého projektu. Je to logické, pretože do ďalšej

fázy tvorby systému by ste sa len ťažko dostali bez toho čo vlastne budete tvoriť. Musíte zhruba vedieť ako bude váš systém vypadáť a čo by mal systém vykonávať. Taktiež si musíte zistiť dostupné prostriedky, či už ekonomické alebo fyzické potreby na daný systém. Takáto špecifikácia je označovaná ako inžinierstvo požiadavkov (requirements engineering).

Inžinierstvo požiadavkov teda stanovuje služby, ktoré by mal vyvíjaný systém poskytovať a taktiež obsahuje obmedzenia, s ktorými sa musí daný systém vyrovnáť a pracovať s nimi. [2, s. 42]

Zber požiadavkov na systém môže prebiehať vo forme komunikácie so zákazníkmi a užívateľmi za účelom získania ich vlastných požiadavkov na systém. Najjednoduchším spôsobom, ako môže analytik získať potrebné požiadavky, je zisk informácií v podobe dotazníka. V prílohe I. je obsiahnutý kompletný dotazník k DP.

## Dotazníkový prieskum - webová aplikácia pre začínajúce hudobné kapely

Cieľom tohto dotazníka je upresnenie popřípade vylepšenie funkcií webovej aplikácie, ktorá by mala byť čo najlepšie prispôsobená požiadavkám používateľa alebo hud. skupiny. Keďže sa táto aplikácia týka hudobného priemyslu, tak som si myslel že prípadné vylepšenia, doplnky alebo komentáre by nemohol nikto iný lepšie schváliť, popřípade navrhnúť ako sám používateľ teda práve ty (vy). Týmto ťa prosím o prečítanie už nami navrhnutých funkcií a samozrejme, to najdôležitejšie, o doplnenie tvojich postrehov, o ktoré by si mal záujem a ktoré chýbajú v iných portáloch. Ďakujeme.

\*Povinné pole

Názov kapely / speváka \*

Váš kontakt (nepovinné)

email, icq ...

Komentáre, komentovanie jednotlivých piesní \*

možnosť pridávania komentárov k jednotlivým kapelám alebo skladbám

súhlasím

Obrázok 4. Zisk informácií v podobe dotazníka (ukážka zberu dát)

### 2.2.1 Definícia požiadavkov

Požiadavok možno definovať ako „špecifikáciu toho, čo by malo byť implementované“. Rozlišujeme dva typy požiadavkov:

- **Funkčné požiadavky**, ktoré predstavujú funkcie a služby systému ako sú napríklad funkcie v študijnom systéme STAG predstavujúce „zápis skúšky“, „editáciu sylabu“ alebo „hľadanie čísla študenta“.
- **Nefunkčné požiadavky**, predstavujú funkcie, ktoré nesúvisia s funkciami z pohľadu užívateľa. Patrí sem spoľahlivosť, robustnosť, využitie programovacieho jazyka poprípade typ aplikácie (webová, klasická) ...

## 2.3 Use Case - Model prípadov použitia

V softvéri a v systémovom inžinierstve predstavuje model prípadov použitia (ďalej ako UC) zoznam krokov, ktoré väčšinou definujú interakcie medzi rolou (v jazyku UML ako „aktívny účastník“) a systémom, pre dosiahnutie určitého cieľa. Pod pojmom aktívny účastník si môžeme predstaviť osobu alebo externý systém.

### 2.3.1 Modelovanie prípadov použitia

Modelovanie prípadov použitia je jednou z foriem inžinierstva požiadavkov. Tvorba modelu UC spočíva v spracovaní požiadavkov, hľadania hraníc systému, hľadania aktérov (aktívnych účastníkov), hľadania prípadov použitia a tvorby scenárov.

Výstupom je potom model UC, ktorý obsahuje štyri komponenty:

- **Účastníci** (actors), predstavuje role, ktoré sú pridelené jednotlivým osobám, ktoré prichádzajú do styku so systémom.
- **Prípady použitia** (Use Cases), predstavuje všetky činnosti, ktoré je schopný účastník s týmto systémom vykonávať.
- **Relácie** (relationships), vzťahy ktoré nastávajú medzi jednotlivými účastníkmi a prípadmi použitia.
- **Hranica systému** (boundary), ktoré predstavuje ohraničenie modelového systému.

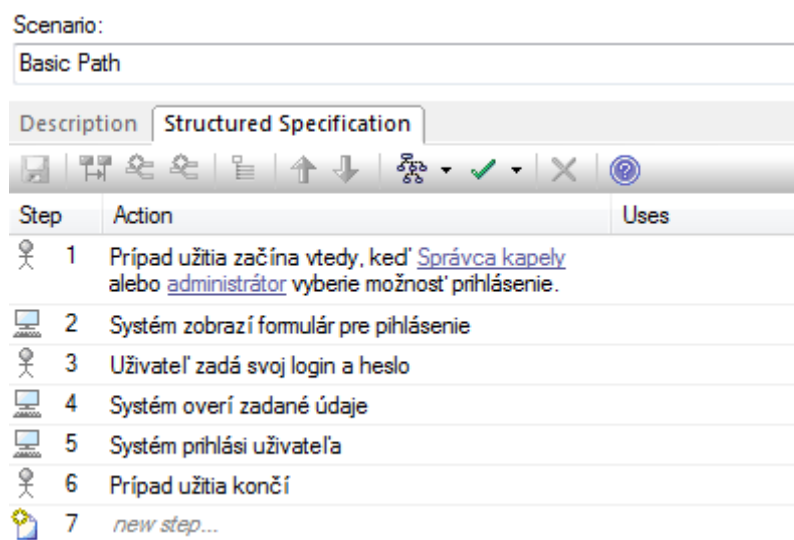
### 2.3.2 Komplexné prípady použitia – Scenár

Prípady použitia by mali vypadáť a vystupovať čo v najjednoduchšej forme. Niekedy sa ale daný problém alebo prípad použitia nedá zobrazit' v jednoduchej forme. Pre zjednodušenie takéhoto problému existuje v UML komplexný prípad použitia.

Najhlavnejším spôsobom ako zobraziť komplexný problém, je vo forme scenáru. Dôležitým aspektom scenára je skutočnosť, že sa ďalej nerozvetvuje. Každý prípad použitia obsahuje presne jeden hlavný scenár. Každý Use Case musí tiež počítať s takzvaným plánom B, teda s vedľajšími scenármi. Sú to alternatívne cesty tokom udalostí. [2, s. 69]

Scenario:  
Basic Path

Description    Structured Specification



Step	Action	Uses
1	Prípad užitia začína vtedy, keď <a href="#">Správca kapely</a> alebo <a href="#">administrátor</a> vyberie možnosť prihlásenie.	
2	System zobrazí formulár pre prihlásenie	
3	Užívateľ zadá svoj login a heslo	
4	System overí zadané údaje	
5	System prihlási užívateľa	
6	Prípad užitia končí	
7	<i>new step...</i>	

Obrázok 5. Ukážka scenára prípadu použitia v programe Enterprise Architect

## 2.4 Analytické triedy

Analytické triedy predstavujú abstraktný pohľad na systém. Systém je vyjadrený ako kolekcia tried. Tieto triedy by mali mapovať pojmy zo skutočného sveta, zjednodušene povedané, názvy obsiahnuté v týchto triedach by mali byť podľa tohto kritéria starostlivo vyberané. Analytické triedy by mali jednoznačne mapovať určitý pojem skutočného sveta (obchodného), ako je napríklad „zákazník“ alebo „produkt“. [2, s. 128-129]

Pre vyhľadávanie správnych analytických tried neexistuje žiadny algoritmus, ktorý by dokázal jednoznačne nájsť analytické triedy. Existuje veľké množstvo zaručených techník, ktoré dokážu efektívne vyhľadávať analytické triedy.

### 2.4.1 Analýza podstatných mien a slovies

Táto metóda vyhľadávania tried vychádza z existujúcich dokumentov a je veľmi jednoduchým nástrojom na analýzu textu. Podstatné mená v hľadanom texte predstavujú triedy a atribúty a naopak slovesá predstavujú operácie danej triedy alebo zodpovednosť.

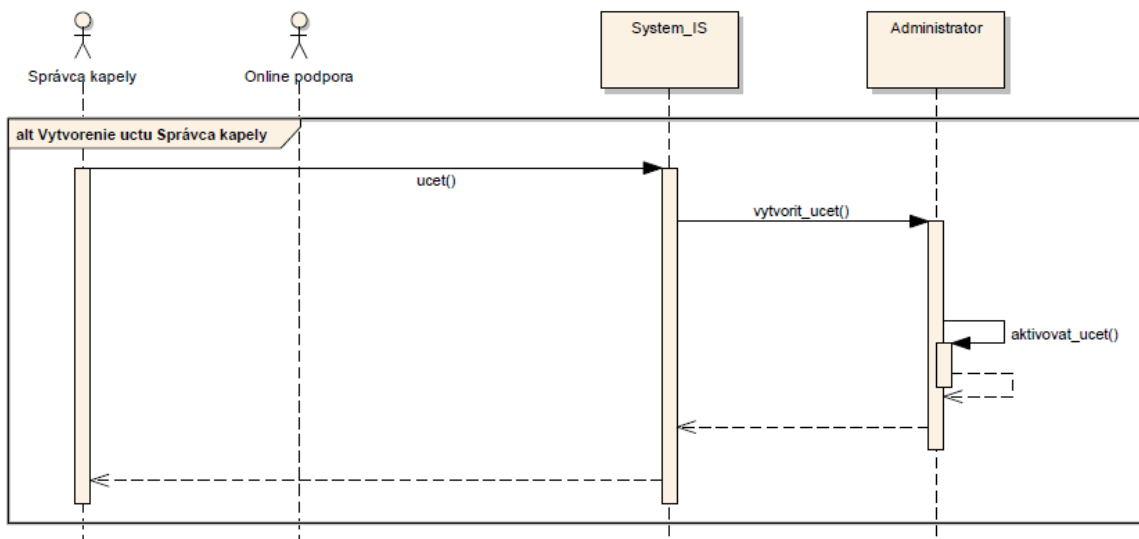
## 2.4.2 Metóda CRC

Ďalšou metódou vyhľadávania tried je metóda CRC (Class Responsibility Collaborator). Metóda zapojuje do vyhľadávania aj samotného užívateľa. Je založená priamo na brainstormingu.

## 2.5 Sekvenčné diagramy

Sekvenčné diagramy predstavujú časovo radenú sekvenciu správ. Majú veľkú výhodu oproti iným diagramom práve kôli týmto chronologickým usporiadaniam. Sekvenčný diagram neukazuje priamo väzbu medzi objektami.

Časová osa je zobrazená zhora nadol. Inštancie naopak, tie sú zobrazené vodorovne. Pod každou inštanciou sa nachádza jej čiara života, ktorá predstavuje dĺžku inštancie.



Obrázok 6. Ukážka sekvenčného diagramu

### 2.5.1 Iterácia a vetvenie

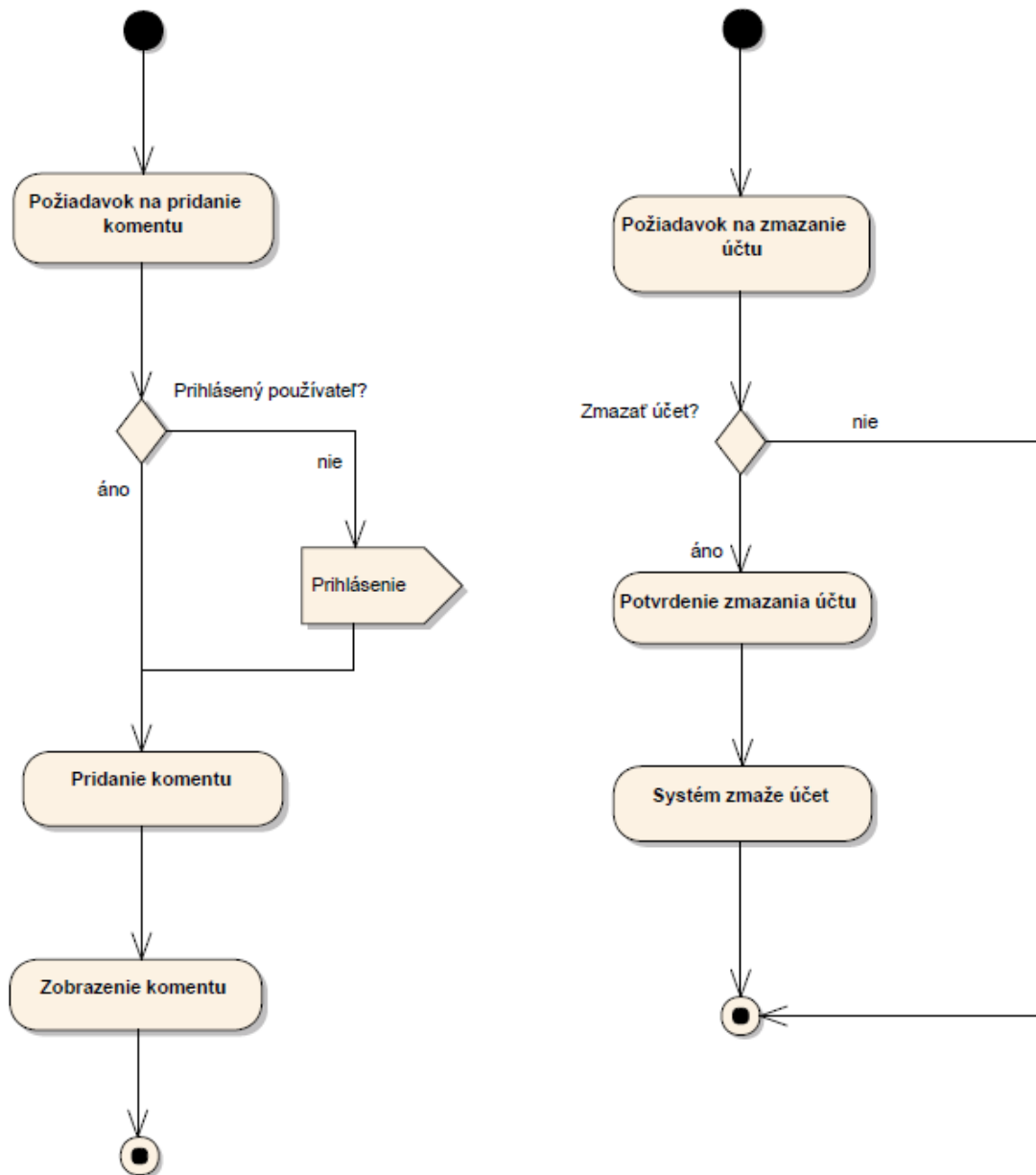
**Iterácia** je v sekvenčnom diagrame zobrazená ako uzavretá množina s interáciou pod vybraným obdĺžnikom.

**Vetvenie** sa realizuje rozdelením aktivácie minimálne na dva objekty s ochranou prvej správy pomocou podmienky. [2, s.222]

## 2.6 Aktivitné diagramy

Diagramy aktivít sú objektovo orientované diagramy tokov. Jednotlivé procesy je možné modelovať ako kolekciu aktivít a prechodov medzi nimi. [2, s. 230]

Aktivita predstavujú sieť prvkov a spojnic, ktoré sa skladajú z jednotlivých uzlov (nodes) a tie sú prepojené hranicami. Spojnice sú zobrazené ako Control flow a Object flow. Prvý predstavuje tok riadenia v danej aktivite a Object flow predstavuje naopak tok objektov v danej aktivite. Možno ich použiť pre všetky prípady použitia, triedy, komponenty a rozhrania.



Obrázok 7. Aktivitné diagramy a ich prevedenie

### 3 ZABEZPEČENIE WEBOVÝCH APLIKÁCIÍ

Dnešné moderné webové stránky a aplikácie sú stále viac a viac zložitejšie. Týka sa to hlavne množstva informácií, ktoré tieto webové aplikácie spracovávajú. V minulosti nám stačila jednoduchá www stránka na zobrazovanie všetkého potrebného a dôležitého. Postupom času nastal rozmach multimédií, obrovských webových portálov a databázových systémov. Toto množstvo informácií musí byť uchovávané v databázových tabuľkách. Tieto informácie nesú vo veľkej miere citlivé informácie a majú dôverný charakter. Preto pri tvorbe webovej aplikácie nemôžeme zabudnúť na jednu z najdôležitejších vecí a tým je zabezpečenie webovej aplikácie. Netreba podceňovať aj tie najmenšie chyby, poprípade nedostatky, ktoré môžu v konečnom dôsledku spôsobiť obrovské problémy, nielen nám ako tvorcovi webovej aplikácie, ale aj tým ktorí túto aplikáciu používajú a vkladdajú do nej dôverné informácie. Ak sa objaví vo vašej aplikácii najslabší článok zabezpečenia, je takmer isté, že to bude hlavná vstupná brána pre útočníkov.



Obrázok 8. Graf najčastejšie sa vyskytujúcich útokov na webové stránky

### 3.1 SQL Injection

Táto metóda využíva nedostatky, ktoré sú založené na nepozornosti samotného programátora. Sú to hlavne chyby v kontrole vstupných dát. Ide hlavne o rôzne relačné databázy, ktoré sa používajú pri uchovávaní informácií z webových aplikácií. Pod pojmom relačné databázy môžeme nájsť napríklad databázu MySQL, PostgreSQL a hlavne databázu ORACLE, ktorá sa objavuje a je používaná aj v tejto diplomovej práci.

Útok SQL injection zneužíva hlavne nedostatočnú kontrolu vstupných dát aplikácie. Útočníkovi sa tak naskytá možnosť jednoduchého vloženia vlastného (nebezpečného) kódu do aplikácie a tým môže upravovať, mazať dáta a položky v databázy. [3]

#### 3.1.1 Priebeh SQL injection útoku

Cieľom útočníka pri tomto type útoku je získavanie čoraz väčšieho množstva informácií. Priebeh SQL útoku možno rozdeliť do troch fáz:

##### 3.1.1.1 Prieskum aplikácie

Prvou fázou útoku je prieskum aplikácie. Útočník najskôr preskúma prezentačnú vrstvu. Zameriava sa hlavne na HTML formuláre, kde vyhľadáva prvky na vstupy dát `<input>`. Cieľom tejto fáze útoku je nájsť také vstupy, ktorých obsah bude slúžiť ako vstup do SQL dotazu a teda bude potencionálne náchylný na SQL injection. Existuje veľké množstvo nástrojov, ktoré dokážu efektívnejšie vyhľadávať tieto miesta a tým podstatne urýchlia prieskum kódu. [4]

##### 3.1.1.2 Testovanie náchylnosti na chyby

V tejto fáze testovania sa testujú vstupy nájdené v prvom kroku na jednotlivé druhy injection útokov. Cieľom tejto fázy nie je poškodiť alebo upravovať databázu. Používa sa len príkaz *SELECT*, ktorý ma za úlohu zistiť možnosti útoku a prieniku do databázy.

##### 3.1.1.3 Útok

Záverečná fáza predstavuje samotný útok, kedy útočník vloží vlastný upravený kód do aplikácie a tým zmení jej chod.

### 3.1.2 Ako sa brániť

Existuje niekoľko spôsobov, ako sa brániť proti útoku typu SQL injection.

#### 3.1.2.1 Escapovanie vstupných dát

V tomto spôsobe obrany sa jedná hlavne o odstránenie znakov ako sú úvodzovky alebo apostrofy, ktoré slúžia k zadávaniu textových reťazcov. Použitím práve escapovacích znakov dosiahneme toho, že parser to nebude brať ako zmenu kontextu príkazu, ale o priamo zadávanú textovú hodnotu.

V našom prípade si spomenieme špeciálne znaky, ktoré využíva databáza **ORACLE** k escapovaniu jednotlivých reťazcov.

- { } - Zložené zátvorky sa používajú hlavne pri escapovaní reťazcu znakov alebo symbolov.
- \ - Spätné lomítko sa používa pri escapovaní jedného znaku alebo symbolu.

#### 3.1.2.2 Kontrola dátového typu

Aby sa predišlo tomuto typu útoku je nutné vždy overovať vstupné údaje a teda ich dátový typ. Ak bude správne definovaný dátový typ, znemožníte tak útočníkovi kombinovať jednotlivé stĺpce vo vašej tabuľke. Pokiaľ by si chcel útočník „vypožičať“ heslá uložené vo vašej databáze a náhodou by sa mu podarilo obísť vaše zabezpečenie, môžete mu to znepříjemniť zahashovaním hesiel. Ak náhodou získa vaše hashe v podstate tým nič nezíska.

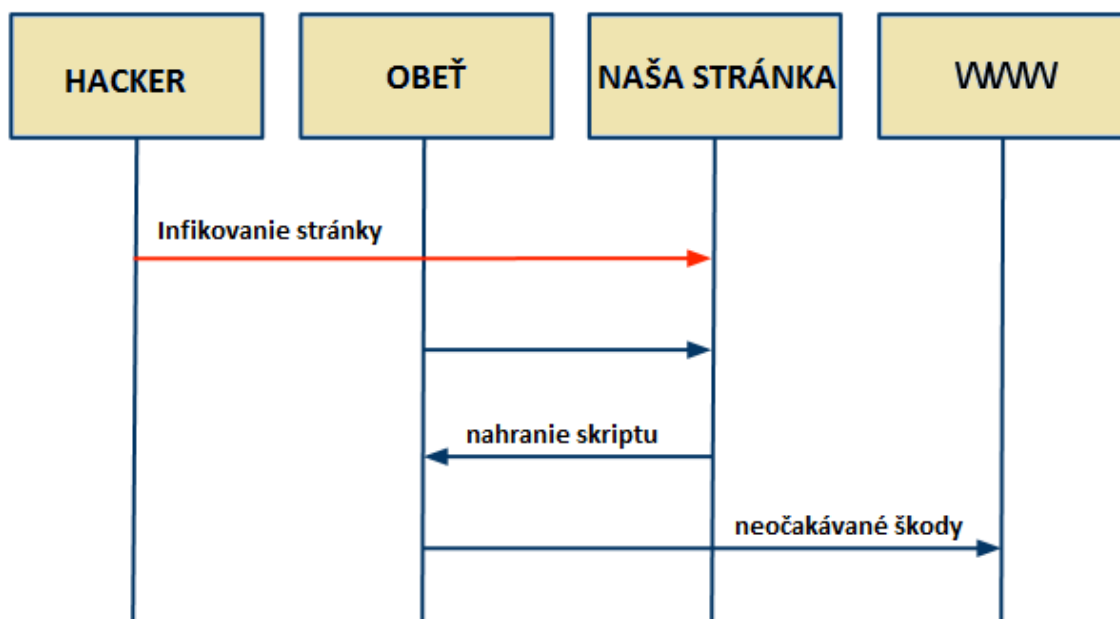
#### 3.1.2.3 Framework

Využitie frameworku (Nette) vo väčšine prípadov uľahčuje prácu s kódom a jeho zabezpečením, ale aj framework môže obsahovať chyby. Pomocou frameworku veľmi významne minimalizujete riziko. Framework sa väčšinou stará o automatické doplnenie escape znakov apod.

### 3.2 Cross-site scripting

Cross Site Scripting (tiež známy pod skratkou XSS alebo CSS) je všeobecne považovaný za jednu z hlavných hackovacích techník. Na obrázku č.9 môžete vidieť, že metóda XSS patrí medzi 3 najviac praktizované techniky. K tomuto sa ešte pridávajú ďalšie techniky z grafu, ktoré môžu byť vedľajším efektom práve XSS. Sem môžeme zaradiť napríklad krádež osobných údajov atď.

Vo všeobecnosti môžeme povedať, že táto technika umožňuje vkladať vlastný JavaScriptový, Vbscriptový alebo HTML kód do zraniteľnej dynamickej stránky za účelom oklamať používateľa, a zbierať citlivé údaje z jeho vlastného pc. Použitie XSS môže vyzrádzať súkromné informácie, manipulovať alebo priamo kraďnúť súbory cookies alebo vytvárať škodlivý kód pre koncových užívateľov.



Obrázok 9. Pohľad na priebeh útoku pomocou XSS

V typickom útoku Cross Site Scripting, útočník infikuje cieľovú stránku pomocou svojho škodlivého skriptu. Druhým krokom je nahranie škodlivého skriptu priamo do pc obeti, ktorá nič netušiac navštívi túto stránku.

Pre vývojára webu je veľmi dôležitý prvý krok, kedy musí zabrániť vloženiu tohoto škodlivého skriptu.

### 3.2.1 Ako sa brániť

Predtým ako spomeniem jednotlivé typy obrany proti XSS, by som chcel pripomenúť, že prvým krokom by mala byť kontrola zraniteľností. Táto kontrola prebieha online a okrem kontroly XSS skontroluje aj zraniteľnosti proti ostatným typom útokov, ako je napríklad SQL injection.

#### 3.2.1.1 Escapovanie

Tak ako tomu bolo pri SQL injection, existuje aj pri XSS táto forma obrany. Používaním tejto efektívnej formy obrany hovoríte webovému prehliadaču, že s dátami, ktoré odosielate, by malo byť zachádzané tak ako s dátami a nemali by byť prezentované žiadnou inou formou. Ak sa útočník rozhodne vložiť škodlivý skript do vašej stránky, obeť nebude nijakým spôsobom v ohrození pretože webový prehliadač nespustí daný skript ak nie je správne ošetrený.

V HTML môžete ošetriť nebezpečné znaky použitím postupnosti znakov `&` nasledujúcich určitým znakovým kódom. V Tabuľke 1 je uvedený príklad.

Tabuľka 1. Ukážka prevedenie escapingu v HTML kóde

Znak	Escape
"	&#34
#	&#35
&	&#36
`	&#37
(	&#38
)	&#39
/	&#40
;	&#41
<	&#42
>	&#43

Využitie tejto metódy pri HTML je celkom jednoduché. Nato aby ste úspešne ošetrili celú vašu web stránku budete musieť vedieť vykonať túto metódu aj na JavaScripte, štýloch CSS a niekedy aj na XML dátach.

Aby ste nemuseli robiť túto prácu sami, existuje veľa knižníc, ktoré vám uľahčia prácu. Dve najznámejšie knižnice sú **ESAPI** a **AntiXSS** od spoločnosti Microsoft.

### 3.2.1.2 *Filtrovanie*

Všetky XSS útoky prebiehajú vo forme nejakého užívateľského vstupu. Škodlivý kód môže pochádzať z jednoduchého formulára ( `<FORM>` ) alebo z oveľa viac zložitejšej formy ako je napríklad *JSON skript*. Preto musí programátor dávať pozor na prichádzajúce zdroje.

Najjednoduchšou a najľahšou formou ochrany pred XSS je práve použitie filtra, ktorý bude kontrolovať všetky prichádzajúce spojenia a zdroje a tým odstraňovať poprípade nahradzovať nebezpečné kľúčové slová, ako je napríklad spomínaný skriptovací tag (`<SCRIPT>`) .

Veľa vývojárov si vybrala implementáciu svojho vlastného filtrovacieho mechanizmu. Väčšinou sa tento skript píše v skriptovacom jazyku PHP alebo ASP. Existujú aj knižnice, ktoré by mali ale pochádzať z dôveryhodných zdrojov aby sa tak zabránilo ich prípadnému zneužitiu.

## 4 ORACLE

Databáza Oracle je objektovo relačný databázový systém, ktorého funkcia spočíva vo vkladaní a výbere relačných informácií a dát. Okrem základných funkcií, ktoré poskytuje každý databázový systém, ponúka navyše podporu pre objekty (BLOB) a metódy, ktoré je možno s nimi vykonať. BLOB alebo taktiež veľký binárny objekt obsahuje binárne dáta s maximálnou veľkosťou až 4 GB. Tento objekt poskytuje základ pre prácu s multimediálnymi objektami ako je video, obraz a zvuk. Príslušný databázový server je hlavným kľúčom pri riešení problému so správou týchto informácií. Spoľahlivo spravuje veľké množstvo dát vo viacuzivateľskom prostredí a tak môžu k nim užívatelia pristupovať súčasne. Databázový server taktiež zabraňuje neoprávnenému prístupu a poskytuje efektívne riešenie zotavenia po prípadnom zlyhaní systému.

### 4.1 Úložisko dát

Oracle ukladá dáta logicky vo forme tabuľkových priestorov a fyzicky vo forme dátových súborov. Každý tabuľkový priestor sa skladá z jedného alebo viacerých súborov na disku (dátové súbory). Tento dátový súbor môže patriť iba k jednému tabuľkovému priestoru. Po pridaní ho nie je možné z tabuľkového priestoru odstrániť ani priradiť k inému tabuľkovému priestoru. Vďaka ukladaniu dát z tabuliek a samotných tabuliek na disk, je možné optimalizovať spracovanie vstupných a výstupných požiadavkov na databázu. Najnovšie, databázové systémy obsahujú nástroj na rozdeľovanie, ktoré umožňuje rozdeľovanie tabuliek na základe rozdielnej sady kľúčov. Špecifické particie tak môžu byť jednoducho pridané alebo odstránené, čo pomáha spravovať veľké dátové sady. [5, s. 40-41]

### 4.2 Oracle InterMedia

Spracovanie multimédií je jednou z veľkého množstva výhod, ktoré databáza Oracle poskytuje. Táto výhoda umožňuje databázy ukladať, spravovať a vyberať obrázky, zvukové stopy, video alebo iné mediálne dáta. Oracle InterMedia spravuje multimediálny obsah pomocou vybraných funkcií:

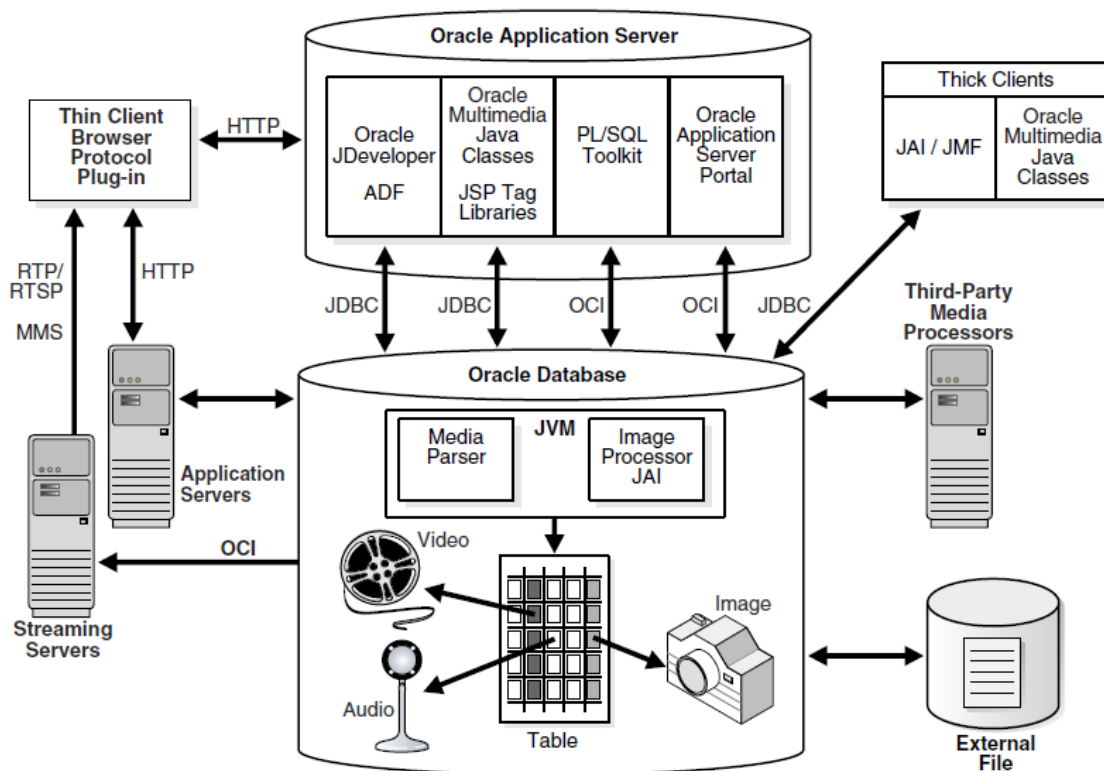
- Ukladanie a načítanie
- Správa mediálnych a aplikačných metadát

- Podpora populárnych formátov (audio, video a obrazové dátové formáty)
- Prístup cez tradičné a webové rozhrania
- Dotazovanie sa na priradené relačné dáta a mediálny obsah s pomocou indexácie

#### 4.2.1 Architektúra

Architektúra Oracle InterMedia popisuje štruktúru, cez ktorú je multimediálny obsah tak isto ako normálne dáta podporovaný v databáze. Pomocou tejto štruktúry môžu byť dáta následne bezpečne zdieľané cez viacero aplikácií, jednoducho spracovávané a spravované pomocou administratívnych nástrojov a technológií. *Obrázok 11.* popisuje architektúru v troch vrstvách:

- 1) Databázová vrstva (databáza Oracle)
- 2) Vrstva aplikačného servera (aplikačný server Oracle)
- 3) Vrstva klienta (tenký a hrubý klient)



Obrázok 10. Architektúra multimediálnej vrstvy Oracle [6]

V prvej vrstve sa uskladňuje bohatý multimediálny obsah spolu s normálnymi dátami v tabuľkách. Vstavaný modul Java VM (virtual machine) zahŕňa mediálny parser spolu s obrazovým procesorom. Tento procesor slúži na vykonávanie operácií ako je konverzia obrazových formátov, indexácia a porovnávanie obrázkov.

V druhej vrstve aplikačný server zabezpečuje prístup k Oracle InterMedia cez Java triedy, ktoré dovoľujú Java aplikáciám na všetkých vrstvách pristupovať, manipulovať a meniť zvukové, obrazové a video súbory uložené v databáze.

V poslednej alebo klientskej vrstve prebiehajú lokálne výpočty cez Java triedy (JAI a JMF). Tieto multimediálne triedy podporujú priamy prístup ku všetkým mediálnym typom od klienta.

#### **4.2.2 Oracle Call Interface (OCI)**

Oracle Call Interface je najkomplexnejšie a najvýkonnejšie rozhranie založené na programovacom jazyku C, ktoré odkrýva plné využitie databáze Oracle. Slúži na vytváranie aplikácií, ktoré používajú funkciu na volanie prístupu k databázovému serveru a kontrolujú všetky fázy prevedenia SQL príkazov. Podporuje dátové typy, volacie konvencie, syntax, a sémantiku jazyka C a C++. OCI je využívané spolu s poprednými open-source rozhraniami ako je napríklad PHP OCI 8 rozšírenie, ruby-oci8 alebo Python cx\_oracle. Pomocou tohto rozhrania sa zvyšuje výkon a škálovateľnosť pri využívaní systémovej pamäte a sieťového pripojenia. Získava sa prístup k externým databázam a vytvára aplikácie pomocou, ktorých je možno zvyšovať počet užívateľov bez nutnosti ďalšej investície do zariadenia.

##### **4.2.2.1 Výhody**

OCI poskytuje obrovské množstvo výhod, ktoré sú spájané s metódami prístupu k databázy Oracle:

- Lepšia kontrola nad všetkými aspektami návrhu aplikácie
- Vysoký stupeň kontroly nad vykonávaním programu
- Podpora dymanického SQL
- Použitie známych programovacích techník a nástrojov tretej generácie
- Asynchrónne oznamovanie udalostí pre registrované klientské aplikácie

### 4.3 Špeciálne multimedialne nástroje

Oracle používa objektové typy, podobné triedam programovacích jazykov C++ a Java, k popisu multimedialných dát. Tieto objektové typy sú: ORDAudio, ORDDoc, ORDImage a ORDVideo. Inštancia týchto objektových typov pozostáva z atribútov, vrátane **metadát, mediálnych dát a metód**. Medzi mediálne dáta patria práve zvukové, obrazové, video alebo iné dáta. Metadáta obsahujú informácie o dátach, ako je objektová dĺžka, typ kompresie alebo formát. Medzi metódy radíme procedúry, ktoré môžu byť vykonávané na daných objektoch: getContent() a setProperties().

#### 4.3.1 Objektový typ – ORDImage

ORDImage zahŕňa ukladanie, výber a spravovanie digitálnych obrázkov. Podporuje 2-D, statické, digitálne rastrové obrázky, ktoré predstavujú reálne objekty, reálneho sveta a sú uložené ako binárne súbory. Zahŕňa všetky obrázky, ktoré sú vyprodukované buď za pomoci digitálnej kamery, scannera alebo vytvorené za pomoci programových algoritmov. [6]

##### 4.3.1.1 Komponenty

Obrázky v digitálnej podobe pozostávajú z obrazových dát (bity) a atribútov, ktoré popisujú a charakterizujú obrazové dáta. V databáze sa taktiež ukladajú údaje ako sú: čas snímky, meno autora fotografie alebo prístroj, ktorého bola snímka vyhotovená. Obrazové dáta (pixel) môžu mať rôzne hĺbky (bit / pixel) závislé od toho ako bola snímka nafotená. ORDImage môže upravovať a automaticky vypisovať vlastnosti obrázkov rozdielnych formátov.

Nároky na miesto potrebné pre ukladanie obrazových dát môže byť podstatne väčšie ako pri ukladaní textových a dátových hodnôt. Pre tento problém existuje viacero kompresných formátov, ktoré dokážu daný obrázok zmenšiť len na pár bytov z pôvodnej veľkosti. Samozrejme, tieto kompresné formáty sú bezstratové a teda po dekompresii sú bitovo identické s originálom. Ak by sme použili stratovú kompresiu, obrazové dáta sa zmenia ale voľným okom je táto kompresia nezistiteľná. Preto ak porovnáme bezstratovú kompresiu so stratovou, stratová kompresia ponúka vyššiu kompresiu.

#### 4.3.1.2 *Metadáta*

Od verzie 10g, Release 2 databázy Oracle je pridaná funkcia pre podporu metadát pri všetkých multimediálnych dátach. Metadáta rozširujú objektový typ `ORDImage` pridaním schopnosti čítať (vyberať) a zapisovať (vkladať) metadáta do obrazových dát. Táto funkcia umožňuje hlavne reprezentovať metadáta aj keď sú oddelené od obrazových dát. Metadáta môžu byť uložené v databáze, sú indexovateľné, vyhľadávateľné a umožňuje využívať aplikáciám štandardné nástroje databázy Oracle.

#### 4.3.1.3 *Úprava obrazových dát*

Medzi hlavné spôsoby úpravy obrázkov patrí zmena kódovania formátu obrázku, orezanie, zmena veľkosti a generovanie náhľadov. Ak je zdrojový obrázok vo formáte RAW Pixel (`RPIX`) alebo `BMP`, obsahuje Intermedia veľké množstvo operácií pre zmenu charakteristiky formátu. [6]

### 4.3.2 **Objektový typ – `ORDVideo`**

`ORDVideo` zahŕňa ukladanie, výber a úpravu digitalizovaných video dát v databáze. Video môže byť vytvorené digitálnou kamerou, animačnými technikami a iné prístroje na zachytávanie video sekvencií. Niektoré tieto digitálne kamery zachytávajú analógový alebo spojitý signál, ako napríklad keď video zachytí obraz nahraný na magnetické média, ktoré sú následne skonvertované do digitálnych hodnôt so špecifickými vlastnosťami ako je formát, kódovanie, počet snímkov za sekundu, rozlíšenie, rozmery, dĺžka, typ kompresie, počet farieb a bitová hĺbka.

#### 4.3.2.1 *Komponenty*

Digitalizované video pozostáva z video dát a atribútov, ktoré popisujú tieto dáta. Medzi atribúty môžeme zaradiť taktiež informácie ako názov autora, čas vykonania záznamu alebo typ prístroja. `ORDVideo` dokáže:

- Automaticky získavať metadáta z video súborov všetkých podporovaných formátov
- Získavať atribúty a ukladať ich ako koment atribúty objektu v XML formáte
- Možnosť rozpoznania a podpory iných video formátov (rozšírenie)

### 4.3.3 Objektový typ – ORDAudio

Správa ORDAudio zahŕňa všetky možnosti úpravy ako predošlé objektové typy ORDImage a ORDVideo. Nahrávky a zvuk môže byť produkovaný pomocou diktafónu, mikrofónu alebo inými nástrojmi a aplikáciami v digitálnej podobe. Zariadenia produkujú analógový alebo spojitý signál, ktorý následne konvertujú do digitálnej podoby so špecifickými vlastnosťami ako je formát, kódovanie, počet kanálov, vzorkovacia frekvencia, typ kompresie a dĺžka nahrávky.

#### 4.3.3.1 Komponenty

Aplikácie často zaznamenávajú aj informácie, ktoré sa bežne vyskytujú pri zvukových stopách. Medzi tieto atribúty patrí hlavne popis nahrávky, autor nahrávky alebo umelec. Tie sú následne ukladané v podobe textu do databáze. [6]

Audio formát môže mať rozdielne typy formátov, iné kódovanie, typ kompresie, počet kanálov, vzorkovaciu frekvenciu atď. Výber a zber metadát je taktiež povolený aj pri ORDAudio.

### 4.3.4 Streamovanie obsahu z Oracle databázy

Pomocou Oracle multimédia doplnku môžete streamovať zvukový a video obsah. Tento plugin podporuje streamovací server tretích strán a dodáva tento obsah webovému prehrávaču, ktorý tento obsah následne prehrá.

#### 4.3.4.1 Oracle Multimédia Plug-in pre RealNetworks streamovací server

Tento plug-in umožňuje serveru RealNetworks streamovať mediálne dáta priamo z Oracle databázy do multimediálneho prehrávača. Plug-in je nainštalovaný spolu so serverom RealNetworks a konfigurovaný a spravovaný pomocou administratívnych nástrojov tohto serveru. Plug-in podporuje všetky dostupné formáty z databázy. [6]

## **II. PRAKTICKÁ ČÁST**

## 5 ZÁSADY PRI TVORBE WEBOVEJ APLIKÁCIE

Pri tvorbe tohto projektu sme postupovali podľa nami dopredu stanovenými postupmi, či už pri tvorbe webovej aplikácie alebo pri tvorbe samotného modelového návrhu aplikácie. Na úvod sme si vypracovali dotazníkový prieskum, ktorý bol predstavený náhodne zvoleným hudobným kapelám a spevákom. Podľa výsledkov, ktoré sme dostali z dotazníku, bol následne vytvorený modelový návrh aplikácie v jazyku UML (kapitola 2).

Webová aplikácia obsahuje pokročilé nástroje, ktoré sú obsiahnuté v databázy Oracle. Pri tvorbe takejto aplikácie si musíme dopredu stanoviť, ktoré nástroje budeme používať a ktoré naopak budú nevyužité. Náš projekt je zameraný hlavne na využitie dvoch hlavných nástrojov na prácu s multimédiami. Pri práci so zvukom a obrázkami by sme mali dopredu vedieť, aké možnosti nám ponúka databáza pri ich spracovaní a využití. Databáza je schopná tieto multimediálne dáta uložiť, upraviť alebo porovnávať ale nedokáže ich zobrazit' priamo z tabuľky. Na zobrazenie uložených obrázkov alebo hudby je potrebné tieto súbory uložiť aj na server alebo v našom prípade na pevný disk tzv. *localhost*, ktorý bude slúžiť ako odkladací priestor našej databázy. Pre zobrazenie a jednoduchý prehľad o uložených dátach bola vytvorená databáza. Do tejto databázy sa ukladajú všetky informácie či už o užívateľských dátach ale hlavne o obrázkoch a hudobných súboroch.

### 5.1 Vývojové nástroje

Dôležitou súčasťou pri vývoji aplikácie je správne zvolený programovací jazyk. V našom prípade bol použitý programovací jazyk PHP. Hlavné využitie tohto jazyka spočívalo pri písaní skriptov, ktoré slúžili najmä pre komunikáciu s databázou Oracle. Je to z toho dôvodu, že programovací jazyk PHP vo veľkej miere podporuje prácu s databázou Oracle pomocou *Oracle Call Interface* (kapitola 4.2.2).

#### 5.1.1 SQL a PL/SQL

SQL je v súčasnosti najpoužívanejší jazyk v relačných systémoch riadenia dát. Používa sa pri práci s databázovými systémami. Jazyk SQL ale nepodporuje prácu a spracovanie multimediálnych dát. Pre tento účel boli vyvinuté mnohé nadstavby pre jazyk SQL. V našom prípade pracujeme priamo s vývojovým jazykom databázy Oracle. PL/SQL je priamo vyvinutý firmou Oracle. Jazyk PL/SQL slúži hlavne na prácu a vytváranie

procedúr, cyklov alebo výnimiek. Tatiež priamo podporuje Oracle InterMedia, kde sa nachádzajú všetky nástroje potrebné pre spracovanie obrázkov, hudby atď. [7]

### 5.1.2 HTML

HTML alebo hypertextový značkový jazyk slúži na tvorbu webových stránok a iných informácií, ktoré možno zobrazit' vo webovom prehliadači. HTML je napísaný vo forme tzv. HTML elementov, ktoré pozostávajú z tagov uzatvorených lomenými zátvorkami (napr. <html>). Vo väčšine prípadov sa vyskytujú tieto tagy v pároch. Prvý je tzv. začiatočný tag a druhý ako koncový.

## 5.2 Tvorba designu webových stránok

Pre tvorbu designu webovej stránky existuje veľa grafických nástrojov, ktoré dokážu vytvorit' poprípade upraviť obrázok na požadované rozmery a taktiež do požadovaného vzhľadu. Najdôležitejší je ale jazyk, ktorým sa tieto jednotlivé časti napoja na seba v jazyku HTML.

### 5.2.1 CSS

Kaskádové štýly sú jazykom pre popis spôsobu zobrazenia stránok, ktoré sú napísané v jazykoch HTML, XHTML alebo XML. Hlavnou myšlienkou jazyka CSS je oddelit' programovú časť a štruktúru od vzhľadovej stránky. Jazyk bol navrhnutý organizáciou W3C. Najnovšia verzia je CSS3. [8]

### 5.2.2 Adobe Photoshop

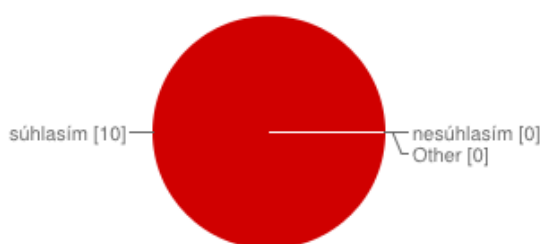
Adobe Photoshop je profesionálny grafický program určený hlavne na úpravu rastrovej grafiky. Program tatiež podporuje úpravu vektorovej grafiky, ktorá sa následne prevádza na rastrovú. Pomocou zásuvných modulov má užívateľ možnosť pridania grafických efektov. Program Adobe Photoshop bol v tejto práci využívaný hlavne na úpravu, korektúru a prispôbenie designu webových stránok. Vytvárané boli celé celky stránky alebo len malé časti ako sú napríklad rôzne ovládacie prvky alebo animácie.

## 6 DOTAZNÍKOVÝ PRIESKUM

Dotazníkový prieskum alebo dotazník je vývojový a vyhodnocovací nástroj, ktorého cieľom je rýchle zisťovanie a zberanie informácií od určitej skupiny respondentov. Ako už názov naznačuje celá funkcia je postavená na dotazoch. V našom prípade bolo respondentom položených 15 otázok s možnosťou voľby dvoch výsledkov. V poslednom prípade je respondentom položená otázka, pri ktorej majú možnosť sa slobodne vyjadriť na danú problematiku a teda vyjadriť vlastný názor.

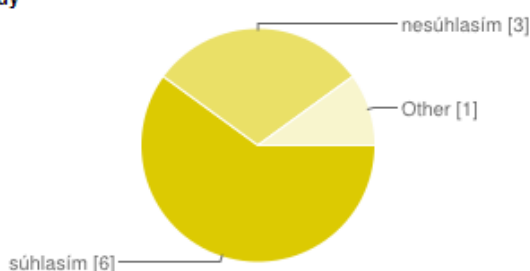
Medzi vybraných respondentov patria hlavne hudobné kapely, ktoré sa pohybujú v hudobnom priemysle, ale taktiež aj kapely, ktoré práve začínajú. Dotazníka sa zúčastnilo celkovo 10 kapiel. Všetky otázky boli vyhodnotenú a spracované do formy grafov.

Fotoreport z akcií



súhlasím	10	100%
nesúhlasím	0	0%
Other	0	0%

Akordy



súhlasím	6	60%
nesúhlasím	3	30%
Other	1	10%

Obrázok 11. Zobrazenie výsledkov dotazníkového prieskumu

Na obrázku 12 sú zobrazené dva grafy, ktoré popisujú dve funkcie webovej aplikácie a ich výsledky. V druhom prípade sa jedná o funkciu akordovania, ktorá sa javila ako najspornejšia hlavne kôli možnosti kopírovania cudzích nápadov, keďže väčšina kapiel nemá svoj vlastný album.

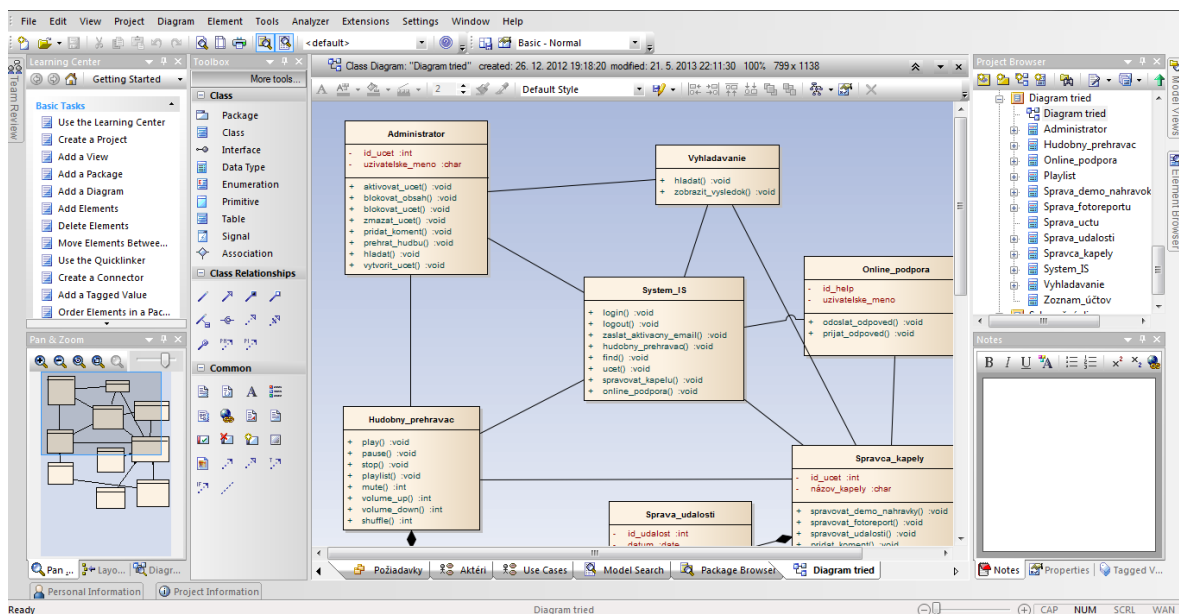
## 7 MODELOVANIE ŠTRUKTÚRY APLIKÁCIE V UML

Pre modelovanie štruktúry a vizualizácie projektu multimediálny sklad v jazyku UML, bol použitý softvér Enterprise Architect (EA). Je to nástroj pre analýzu a návrh, ktorý je založený na OMG UML. Platforma podporuje: návrh a konštrukciu softvérových systémov, modelové podnikateľské procesy a domény založené na modelovom priemysle. Nie je využívaný len na návrh architektúry systémov, ale používa sa hlavne na implementáciu týchto modelov počas celého životného cyklu vývoja aplikácie.

Program dokáže generovať priamo zdrojový kód z modelov správania. Spolupracuje s 10 hlavnými programovacími jazykmi. Medzi najznámejšie patrí napríklad programovací jazyk PHP, C, C++, Java alebo Python. Pre vývoj a prácu v programe EA bola použitá školská verzia software.

Na úvod bola v aplikácii vytvorená úvodná štúdia, ktorá obsahuje základný pohľad na navrhovaný systém. Ďalej boli vytvorené nasledujúce modely a diagramy:

- Požiadavky a Use Case diagramy
- Diagram tried
- Sekvenčné diagramy
- Diagram aktivít



Obrázok 12. Pracovné prostredie vývojovej aplikácie Enterprise Architect

## 7.1 Úvodná štúdia

Na úvod je uvedený základný popis, ktorý popisuje obsah celého dokumentu. Sú tu uvedené hlavné požiadavky na funkcionality webovej aplikácie. V účelovej časti sú uvedené hlavné dôvody tvorby daného softvéru, prečo sa vytvára a čo je jeho hlavnou funkciou a výhodou.

Rozsah obsahuje požadovanú funkcionality. Sú tu zhrnuté hlavné očakávané funkcie aplikácie a sú uvádzané ako celky napr. v našom prípade celok audio prehrávač. V ďalšej časti sú uvedení hlavní užívatelia, ktorí budú pristupovať k webovej aplikácii. V našom prípade sa jedná hlavne o užívateľov z oblasti hudobného priemyslu. Je tu taktiež uvedený modelový scénar:

*Kapela si chce založiť vlastné portfólio, založí si vlastný profil (názov kapely, dátum vzniku, heslo atď.). Administrátor bude mať možnosť zdieľania vlastných fotoreportov z koncertu, nahrávania demo verzií skladieb. Medzi úpravy skladby bude patriť jej názov, autor, celý text piesne popri prípade akordy. Administrátor bude mať taktiež možnosť organizovania vlastných koncertov, ktoré sa budú následne aktualizovať na úvodnej stránke, či už stránky kapely alebo hlavnej stránke ako náhodný výber. Ak príde na stránku bežný užívateľ, ktorý sa nechce registrovať, bude mať možnosť k prístupu vyhľadávania informácii o jednotlivých kapelách (koncerty, zoznam albumov atď.), alebo piesní, ktoré práve potrebuje. Medzi ďalšie možnosti bude patriť audio prehrávač hudby, kde bude užívateľ schopný si vytvoriť vlastný playlist alebo len si tak náhodne pustiť rozne skladby zo svojej tvorby.*

Na záver sú v úvodnej štúdií uvedené obmedzujúce podmienky. Obsahujú najmä podporu určitých štandardov v našom prípade aplikácia nemá mobilnú verziu softvéru. Aplikácia pracuje s databázovým systémom Oracle a je zabezpečené proti rôznym formám útoku ako je napríklad *SQL Injection* alebo *.htaccess*. Sekcia zodpovedné osoby obsahuje všetky formy užívateľa od administrátora hudobnej kapely až po bežného neregistrovaného užívateľa.

## 7.2 Požiadavky

Z dostupných požiadavkov na systém som použil funkčné a nefunkčné požiadavky. Funkčné požiadavky predstavujú základný predmet systému a sú merané konkrétnymi prostriedkami ako sú napríklad hodnoty jednotlivých dát, logika a algoritmy rozhodovania. Funkčné požiadavky špecifikujú čo má daná aplikácia vykonávať.

Nefunkčné požiadavky predstavujú vlastnosti v oblasti chovania, ktoré musia mať stanovené funkcie. Špecifikujú vlastnosti, ktoré musí daný produkt mať.

### 7.2.1 Funkčné požiadavky

#### **Webová aplikácia:**

- Výstupom aplikácie je webová stránka vo formáte HTML 5.0 s formátovaním CSS3.

#### **Demo nahrávky:**

- Užívateľ po prihlásení ako správca kapely získava práva na pridanie vlastnej demo nahrávky.
- Po kontrole správnej prípony súboru sa demo nahrávka nahrá do databázového priečinka.
- Po procedúre sa názov demo nahrávky zobrazí v tabuľke pre zobrazovanie uložených demo nahrávok.
- Prihlásený užívateľ bude mať možnosť túto nahrávku okomentovať, ohodnotiť a pridať akordy k skladbe.

#### **Fotoreport:**

- Správca kapely bude mať možnosť pridať vlastný fotoreport z akcií.
- K nahranej fotke bude mať užívateľ možnosť pridať popis.
- Pri pridávaní môže užívateľ meniť pomer a veľkosť fotografie a taktiež prevádzať jednoduché formy úpravy.
- Fotoreport obsahuje prezentáciu celého fotoalbumu, ktorá sa spustí po kliknutí.
- Fotoreport obsahuje aj ovládanie pomocou klávesových skratiek.

**Hudobný prehrávač:**

- Prístupovať k hudobnému prehrávaču bude mať možnosť každý bez rozdielu účtu.
- Hudobný prehrávač načítava playlist z databáze a jednotlivé nahrávky z úložiska na pevnom disku.
- Hlavné ovládacie prvky prehrávača sú play, stop, pause, pretočenie o 15 sekúnd dopredu/dozadu, náhodné prehrávanie a ovládanie hlasitosti.
- Prehrávač dokáže zobrazit' názov skladby v hornom paneli a taktiež zobrazuje stav ovládacieho prvku v spodnom paneli.

**Organizácia koncertov:**

- Organizácia koncertov je možná len pri správcovi kapely.
- Užívateľ má možnosť pridať udalosť alebo koncert s vlastným popisom.
- Zadáva dátum a čas konania akcie, názov koncertu, miesto konania a rôzne dopňujúce informácie ako je napríklad cena vstupného na akciu.

**Vyhľadávanie:**

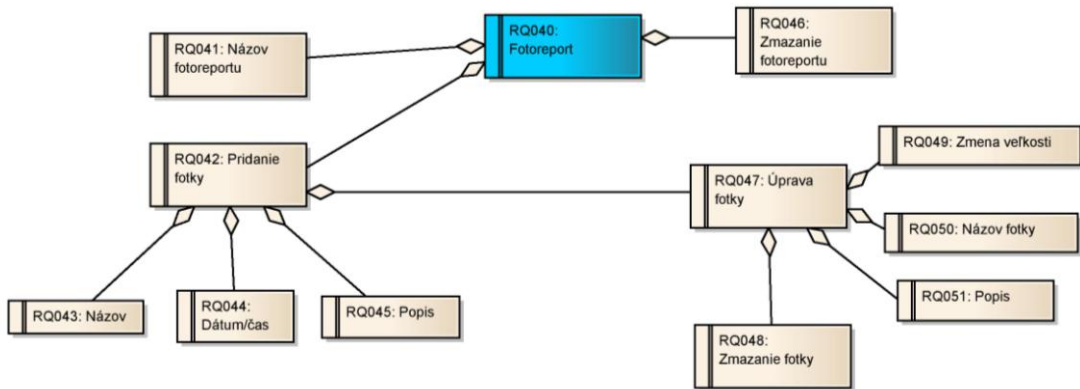
- Prístup k vyhľadávaniu bude mať každý bez rozdielu typu účtu.
- Vyhľadávanie prebieha v rámci databázi.
- Sú vyhľadávané názvy kapiel, členov atď.

**Správa účtu:**

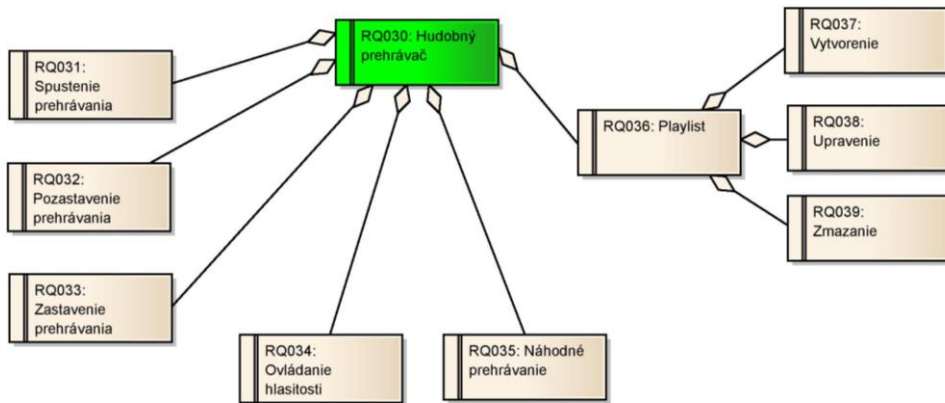
- Registrácia nového účtu prebieha z hlavnej stránky aplikácie.
- Užívateľ má možnosť voľby medzi dvomi typmi účtu (správca kapely a bežný používateľ).
- Po prihlásení do aplikácie zostane užívateľ prihlásený až do odhlásenia.
- Užívateľ má možnosť zmeny hesla.

**Prihlásenie sa do systému:**

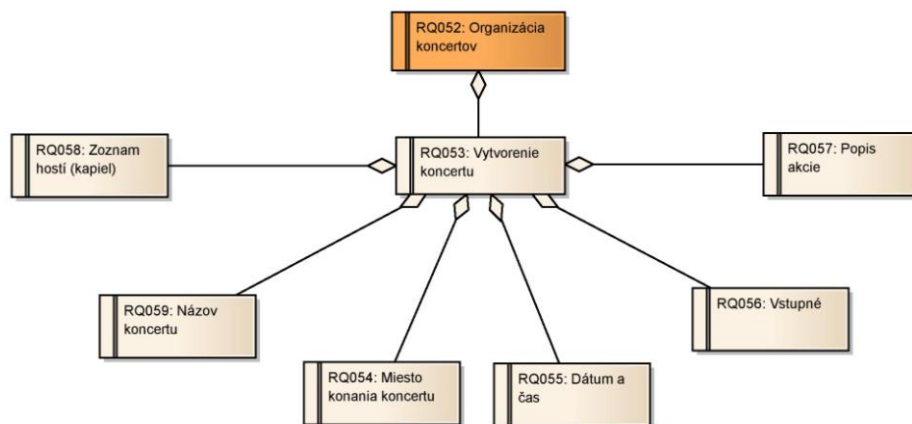
- Prihlasovanie do systému prebieha vo forme meno a heslo.



Obrázok 13. Náhľad funkčného požiadavku pre Fotoreport



Obrázok 14. Náhľad funkčného požiadavku pre Hudobný prehrávač



Obrázok 15. Náhľad funkčného požiadavku pre organizáciu koncertov

### 7.2.2 Nefunkčné požiadavky

Medzi nefunkčné požiadavky sme uviedli štyri, ktoré patria medzi hlavné vlastnosti produktu:

- Implementácia
- Dostupnosť
- Užívateľské prostredie
- Bezpečnosť

Implementácia prebieha pomocou platformy HTML a PHP s využitím databázového systému Oracle a jazyka PL/SQL. Dostupnosť predstavuje webovú aplikáciu, ktorá bude dostupná len cez webový prehliadač. Užívateľské prostredie aplikácie je prehľadné a pre užívateľa bude prívetivé s pekným GUI. Aplikácia bude zabezpečená proti rôznym útokom na heslá alebo údaje (SQL Injection, .htaccess atď.). Nefunkčné požiadavky z EA sú priložené v prílohe na CD.

## 7.3 Use Case Diagram

Tak ako v iných návrhových systémoch tak aj v UML existuje postupnosť úloh, ktoré majú svoje striktné poradie. Diagram prípadov užitia patrí medzi prvé kroky analytika. Tvorba Use Case modelu spočíva v modelovaní systému z pohľadu jeho možných použití. V jednoduchosti sa môžeme na model prípadov užitia pozerat' ako na funkcionality systému alebo čo vlastne systém bude vykonávať.

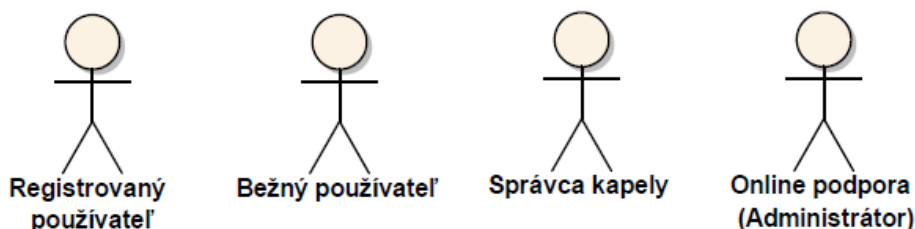
Pri tvorbe modelu prípadu užitia postupujeme podľa metódy „aktérovej školy“. Najskôr sa vyhľadajú prvky z okolia systému tzv. aktéri. Hľadáme všetky prvky, ktoré budú komunikovať a predávať informácie so systémom. Druhým bodom je nájdenie hranice systému alebo Boundary. Hľadanie prípadov užitia je všetko čo bude daný systém vykonávať. Posledným bodom pri tvorbe use case diagramu je písanie scenárov. Scenáre jednoznačne a jednoducho popisujú chod programu.

### 7.3.1 Aktéri systému

Webová aplikácia obsahuje štyroch hlavných aktérov systému. Správca kapely vykonáva najviac funkcií. Jedná sa o správcu účtu kapely, kde má možnosť vytvárať:

fotoreporty, akcie, novinky a pridávať demo nahrávky, z ktorých je následne tvorený playlist kapely. Správca kapely spravuje celý účet kapely. Registrovaný užívateľ nemá naopak prístup k vlastnému profilu alebo stránke kapely ale pri registrácii získava práva na komentovanie, hodnotenie kapely a akordovanie jednotlivých songov.

Bežný používateľ je vlastne každý návštevník webovej aplikácie, ktorý prezerá obsah stránky bez nutnosti registrácie účtu. Má len základné obmedzené podmienky prehliadania ako je prehliadanie profilu kapely, vyhľadávanie a spúšťanie nahrávok cez hudobný prehrávač. Online podpora vystupuje len ako poverený užívateľ, ktorý sa stará o komunikáciu so zákazníkom, ktorý potrebuje poradiť alebo pomôcť ohľadne používania webovej aplikácie.



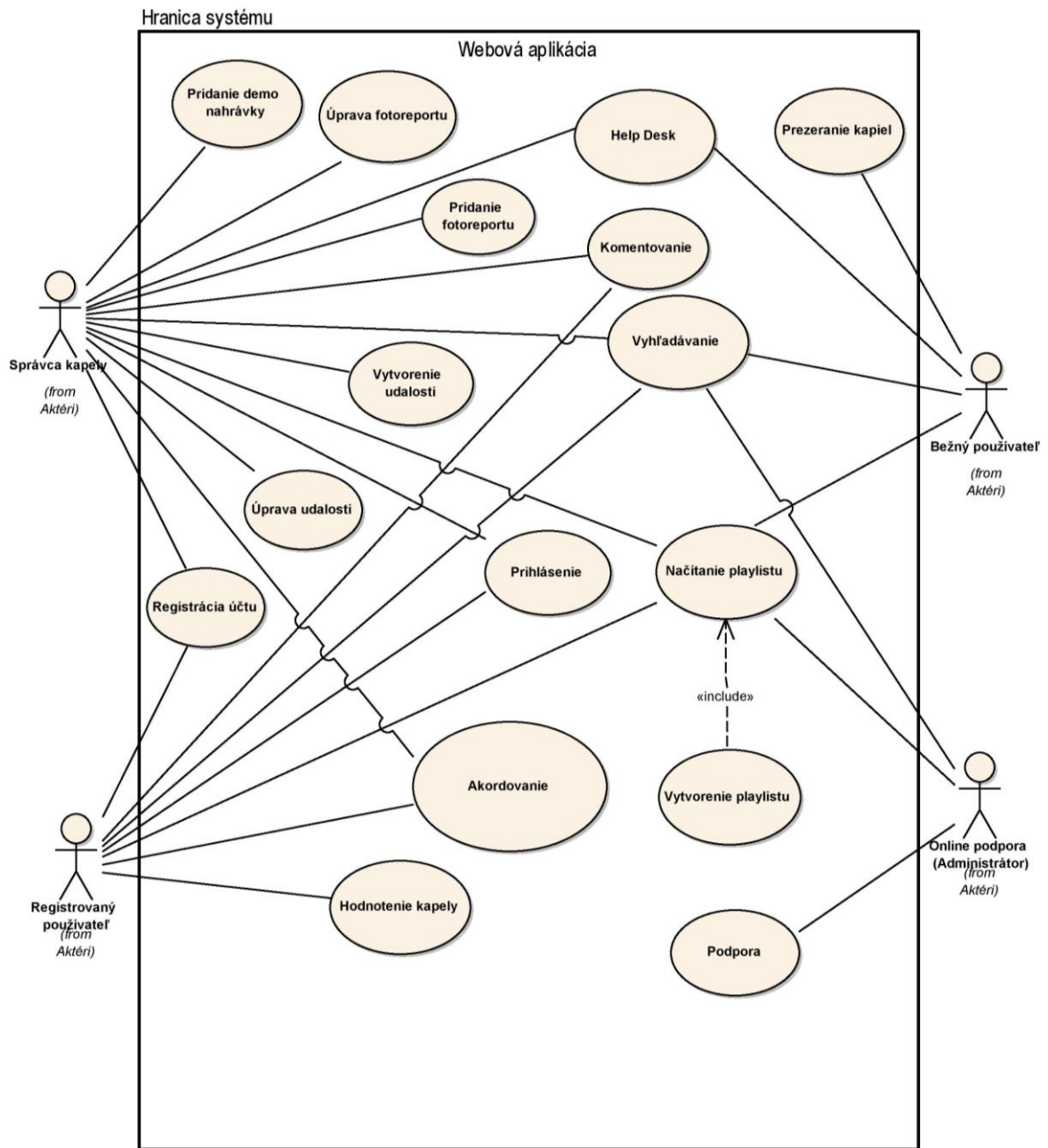
Obrázok 16. Náhľad na aktérov systému webovej aplikácie

### 7.3.2 Hranice systému

Hranice systému (Boundary) určujú hranice medzi vnútrom a okolím systému. Priešičníky medzi spojnicami Use a hranicou systému sa nazývajú rozhraním alebo interface systému. Hranica systému je znázornená na obr.

### 7.3.3 Prípady užitia (Use Case)

Po nájdení aktérov systému a hraníc systému prichádza na rad fáza, kedy sa hľadajú prípady užitia. Prípady užitia hľadáme v závislosti na aktérovi. V našom projekte máme štyroch hlavných aktérov, ktorí plnia v systéme určitú funkciu a majú rôzne práva a možnosti ako pristupovať k systému. Uvedieme si to na nasledujúcom príklade kedy zoberieme do úvahy bežného používateľa, ktorý nemá právo na komentovanie teda tento prípad užitia neprípadá tomuto aktérovi, ale naopak registrovaný užívateľ má potrebné práva a teda získava tento prípad užitia. Všetky prípady užitia a scenáre sú obsiahnuté v prílohe na CD.



Obrázok 17. Pohľad na Use Case Diagram z programu Enterprise Architect

Po úspešnom určení diagramu prípadov užívania nastáva fáza kedy je potrebné popísať vnútrajšok prípadov užívania. Aby sme mali predstavu čo sa vlastne deje vo vnútri každého Use Case, je potrebné zvoliť si správnu metódu popisu. Z možných spôsobov popisu vnútra prípadu užívania sme vybrali spôsob popisu pomocou slovných scenárov. Každý prípad užívania z diagramu obsahuje slovný popis (scenár).

Step	Action	Uses	Results	State
1	Prípád užití začne vtedy keď <a href="#">správca kapely</a> vyberie možnosť 'Pridanie demo nahrávky'			
2	Systém zobrazí formulár pre pridanie demo nahrávky			
3	<a href="#">Správca kapely</a> vyplní požadované údaje a popis k zvukovému súboru			
4	<a href="#">Správca kapely</a> vyberie zvukový záznam z umiestnenia na disku			
5	<a href="#">Správca kapely</a> potvrdí výber súboru stlačením tlačítka OK			
6	Systém skontroluje správnu príponu zvukového súboru			
7	Po overení zvukového súboru, systém priradí zvukovú nahrávku k formuláru			
8	<a href="#">Správca kapely</a> potvrdí pridanie demo nahrávky			

Step	Path Name	Type	Join
0	Basic Path	Basic Path	-

Obrázok 18. Metóda slovného popisu vnútra prípadu užitia (EA)

Step	Action	Uses	Results	State
1	Prípád užitia začína vtedy, keď je vybraná možnosť registrácia účtu.			
2	Systém zobrazí formulár pre pridanie nového účtu			
3	Žiadateľ vyberie typ účtu			
4	Žiadateľ vyplní potrebné údaje			
5	Systém skontroluje kompletnosť údajov			
6	Systém odošle aktivačný email			
7	Systém aktivuje účet.			
8	Systém uloží dáta			
9	Prípád užitia končí			
10	new step...			

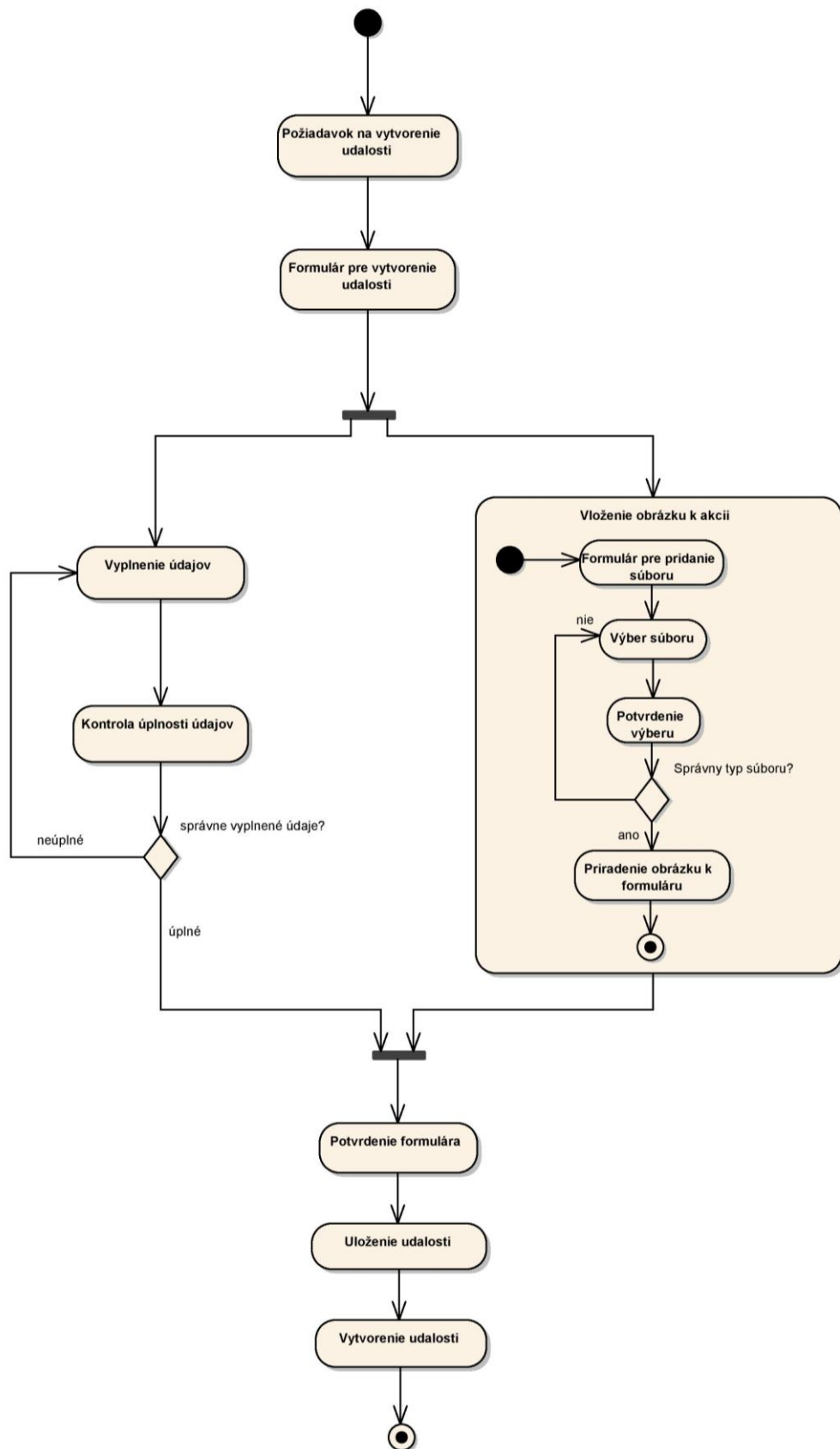
  

Step	Path Name	Type	Join
0	Basic Path	Basic Path	-
5a	Ak nie sú kompletné systém požiada užívateľa o vyplnenie pot...	Alternate	End
6a	Ak nie je Správca kapely dôveryhodný administrátor zamietne j...	Alternate	End

Obrázok 19. Metóda slovného popisu prípadu užitia a alternatívne cesty (EA)

## 7.4 Diagram aktivít

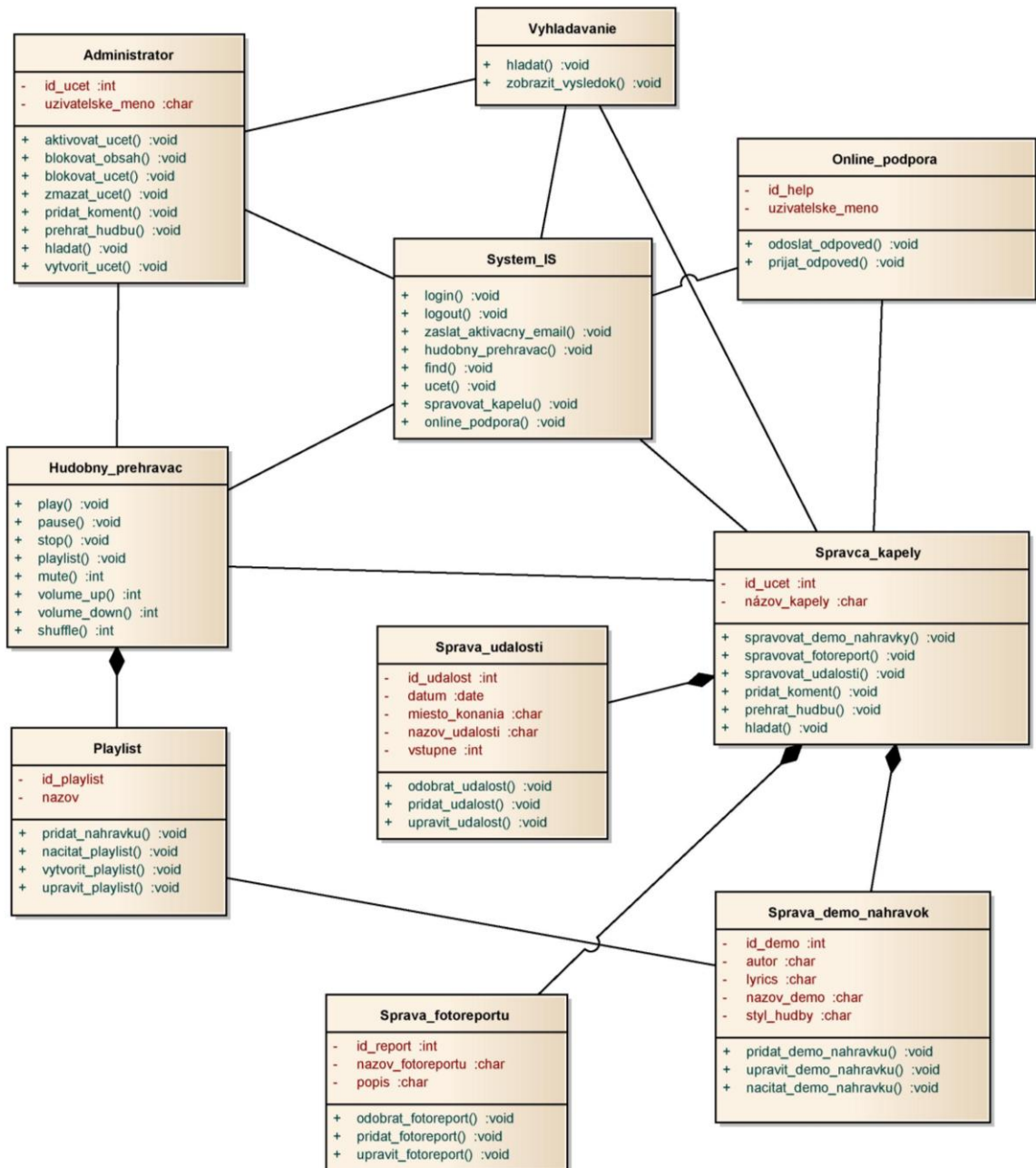
V predošlej kapitole sme popisovali chod procesu pomocou slovného popisu. Okrem slovného popisu možno prípad užitia popísať aj pomocou diagramu aktivít. V tomto prípade sme použili aj diagram aktivít vo vnútri prípadu užitia. Nejedná sa však už o popis chodu procesu ale jedná sa o popis chodu programu, teda o popis činností vo vnútri programu. Jedná sa už o samotné aktivity programu, takže v našom prípade ide o samotné aktivitné diagramy. Diagramy aktivít sú obsiahnuté ako príloha na priloženom CD.



Obrázok 20. Aktivitný diagram pre vytvorenie udalosti (EA)

## 7.5 Diagram tried

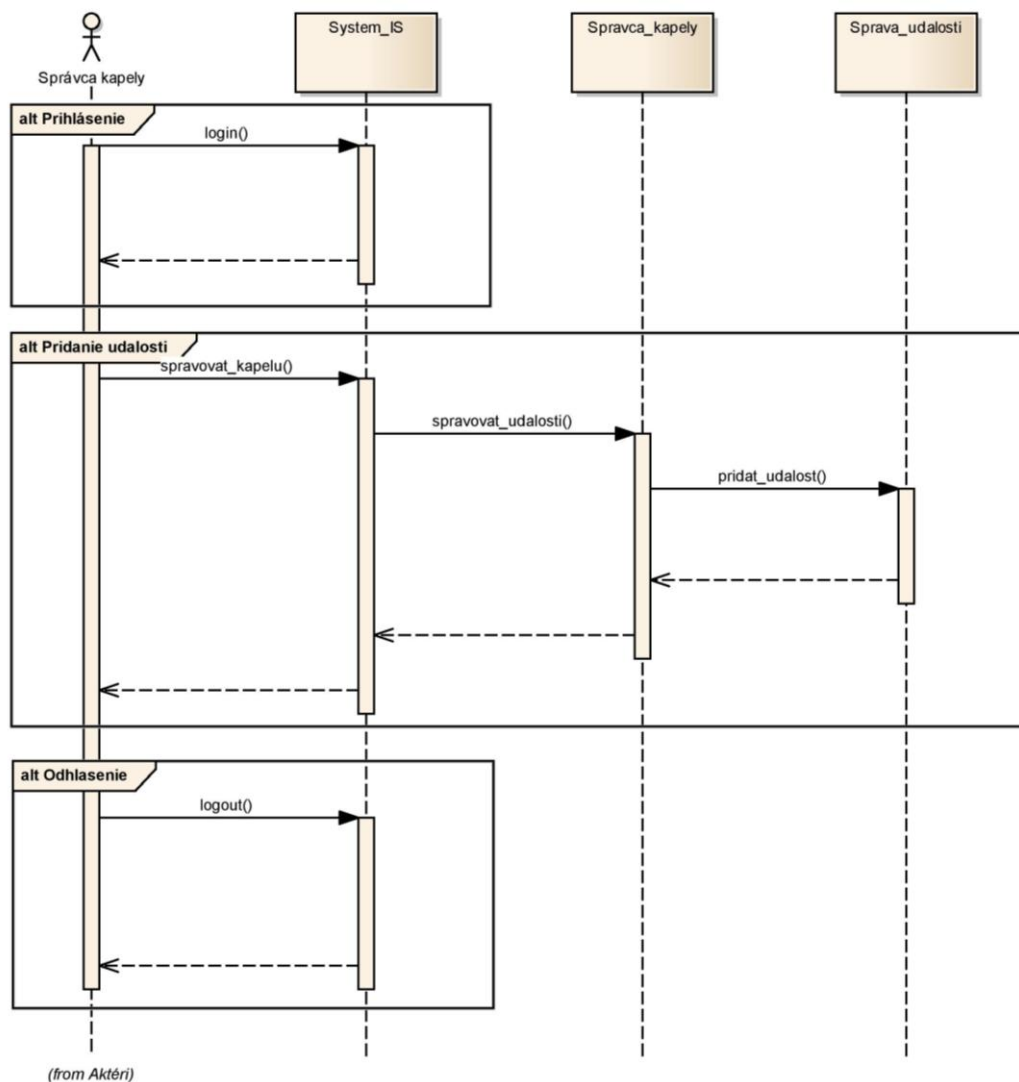
Diagram tried určuje triedy a vzťahy medzi nimi. Používa sa na úrovni analytického modelovania a na úrovni technologického návrhu. Prvým krokom bolo nájsť tieto konkrétne triedy. Ďalším krokom bolo nájsť interakcií (vzťahov) medzi triedami v modeli tried.



Obrázok 21. Diagram tried (EA)

## 7.6 Sekvenčné diagramy

Sekvenčné diagramy zobrazujú sekvencie správ, ktoré sú posielané medzi jednotlivými úlohami, ktoré sú usporiadané v časovej postupnosti. V sekvenčnom diagrame najskôr určíme aktéra a jednotlivé triedy pomocou, ktorých sa vykonávajú jednotlivé úlohy. Jednotlivé časové udalosti sú zobrazené pomocou tzv. fragmentov. Tieto fragmenty sú časovo usporiadané tak ako sa vykonávajú v reálnom čase. Ak sa pozrieme na prvú situáciu (Prihlásenie), kde správca kapely volá operáciu `login()` na triedu `System_IS`. Zo systému dostáva odpoveď v podobe návratovej hodnoty `Is Return`.



Obrázok 22. Ukážka sekvenčného diagramu pre pridanie udalosti (EA)

## 8 WEBOVÁ APLIKÁCIA

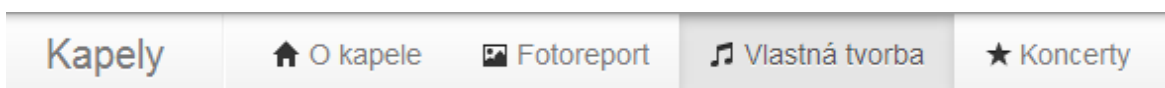
Výstupná webová aplikácia je vytvorená a naprogramovaná pomocou špeciálnych programovacích nástrojov. Medzi hlavné nástroje, ktoré boli použité, patrí najmä: PHP, HTML. Špeciálne procedúry pre prácu a spracovanie audio súborov boli vytvorené pomocou jazyka PL/SQL. Pre procedúry a audio súbory bol vytvorený špeciálny hudobný prehrávač, ktorého funkcie sú popísané v nasledujúcich kapitolách.



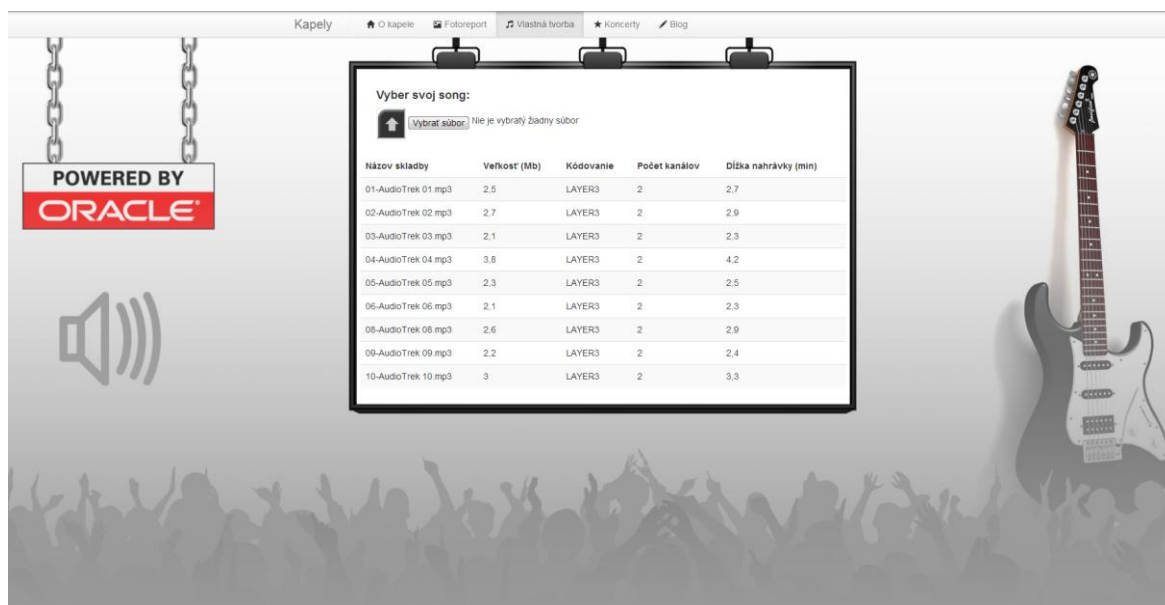
Obrázok 23. Úvodná stránka webovej aplikácie

### 8.1 Spracovanie audio súborov

Po registrácii a následnom prihlásení novej kapely sa správcovi naskytá pohľad na rozhranie pre správu vlastnej kapely. Spracovanie obrázkových súborov, audio súborov, tvorba vlastných udalostí a správa fotoreportu sú hlavnými funkciami aplikácie. Medzi najdôležitejšie funkcie, ktoré sú poskytnuté užívateľovi určite patrí spracovanie vlastných demo nahrávok. Nahratie vlastných demo nahrávok má možnosť každý užívateľ, ktorý si pri registrácii zvolil profil vlastnej kapely. Vo vlastnom profile následne správca vyberie z horného menu možnosť vlastná tvorba.



Obrázok 24. Horné menu pre správcu kapely po prihlásení.



Obrázok 25. Náhľad na hlavnú stránku pre správu demo nahrávok.

### 8.1.1 Príprava databázy

Celá databázová štruktúra bola navrhnutá v rámci projektu, do ktorej boli dodatočne implementované štruktúry, tabuľky a procedúry pre spracovanie audio súborov. Špeciálnym nástrojom pre spracovanie audio súborov bol nástroj ORDAudio. V databázovom systéme boli doplnené dve tabuľky pre spracovanie audio súborov a jedna tabuľka pre výpis vlastností a metadát z predchádzajúcich dvoch tabuliek. Skôr ako boli vytvorené tabuľky bolo potreba určiť špeciálne umiestnenie, kde sa budú ukladať audio súbory určené pre spracovanie databázou a načítavanie audio prehrávačom.

```
create directory mp3 as 'c:\dev\www\kapely\mp3';
```

Po vytvorení *directory* boli vytvorené potrebné tabuľky. Prvá tabuľka spracováva zvukový súbor a nahráva ho do tabuľky v podobe dátového typu BLOB. Spolu s dátovým typom je v tabuľke obsiahnutý aj stĺpec *názov* pre zvukový záznam. Druhá tabuľka slúži pre spracovanie a výpis vlastností zo zvukového záznamu. Zvukové záznamy, ktoré sa nahrávajú do prvej tabuľky a teda aj do dátového typu BLOB sú prevedené do špeciálneho dátového typu ORDAudio. Takto upravené audio súbory sú na záver spracované pomocou procedúr na výpis vlastností a metadát z daných súborov. Tieto vlastnosti sú následne ukladané do tabuľky *vlastnosti\_demo*.

### 8.1.1.1 Špeciálne procedúry a funkcie

V predošlej kapitole bolo spomenuté vpoužitie špeciálnych procedúr pre spracovanie audio súborov. Prvá procedúra, ktorá bola vytvorená, slúži na vloženie audio súboru do dátového typu BLOB. Na začiatku procedúry sa inicializuje tabuľka pre pridanie následného súboru. Nastavia sa základné hodnoty pre jednotlivé parametre. Premenná, ktorá vstupuje do procedúry je názov demo nahrávky, ktorá sa nachádza už v spomínanej zložke, kde sa ukladajú všetky audio dáta.

Inicializácia tabuľky pre dátový typ BLOB:

```
INSERT INTO MP3 (audio_name, audio) VALUES (p_audio_name,  
EMPTY_BLOB ( ) )
```

Druhá špeciálna procedúra, ktorá bola vytvorená má funkciu nahratia dátového súboru vo formáte dátového typu BLOB do ORDAudio. Na začiatku prebieha inicializácia tabuľky. V nasledujúcej časti prebieha nahrávanie súboru BLOB do ORDAudio a nastavenie jednotlivých vlastností.

```
INSERT INTO audio_table VALUES(moje, NULL, ORDAudio.init());  
UPDATE audio_table T SET T.audio.source.localData = (SELECT audio  
FROM mp3 M WHERE M.id = moje) WHERE T.id = moje;  
COMMIT;  
SELECT audio INTO obj FROM audio_table WHERE id = moje FOR UPDATE;  
obj.setProperties(ctx);  
obj.setUpdateTime(SYSDATE);  
UPDATE audio_table SET audio = obj WHERE id = moje;
```


Tretia procedúra, ktorá bola vytvorená, má funkciu pre vyberanie a vkladanie vlastností a metadát z audio súborov uložených v dátovom type ORDAudio. Procedúra je schopná vytiahnuť základné vlastnosti zvukového súboru ako sú: názov demo nahrávky, veľkosť súboru, kódovanie, počet kanálov alebo dĺžka nahrávky.

```
audio.getContentLength  
audio.getaudioDuration
```

### 8.1.2 Upload audio súborov

Nahrávanie zvukových záznamov a demonahrávok prebieha pomocou *HTML form*, v ktorom je volaný špeciálny php skript *upload.php* so vstupom v podobe vybraného súboru. Na začiatku skriptu *upload.php* sú nastavené základné hodnoty ako maximálna povolená veľkosť súboru alebo maximálna dĺžka vykonávania php skriptu.

**Vyber svoj song:**

  01-AudioTrek 01.mp3

Názov skladby	Veľkosť (Mb)	Kódovanie	Počet kanálov	Dĺžka nahrávky (min)
---------------	--------------	-----------	---------------	----------------------

Obrázok 26. Formulár pre vybratie a nahranie zvukového záznamu.

```
ini_set('upload_max_filesize', '10M');
ini_set('post_max_size', '10M');
ini_set('max_input_time', 300);
ini_set('max_execution_time', 300);
```

Vybraný súbor je kontrolovaný viacerými podmienkami. Veľmi dôležitou podmienkou, bez ktorej nemôže existovať žiadny formulár je obmedzenie a ošetrovanie prípon vstupných súborov. V našom prípade vyberáme súbory so zvukovými príponami ako napr: .mp3, .ogg, .wav a málo využívaný ale o to kvalitnejší zvukový súbor s príponou FLAC.

```
$allowed_filetypes = array('.ogg', '.mp3', '.wav', '.flac');
```

Po vybratí správneho typu súboru skript vykoná parsovanie prípony a názvu nahrávky, kedy názov nahrávky je ďalej spracovávaný databázovým systémom a jednotlivými procedúrami. Názov nahrávky vlastne slúži ako vstupná premenná v každej procedúre.

```
$s = oci_parse($c, "begin load_audio_file_test (:bind2); end;");
oci_bind_by_name($s, ":bind2", $filename);
oci_execute($s, OCI_DEFAULT);
oci_free_statement($s);
```

Na záver sú v skripte vykonávané jednotlivé procedúry pre celý postup nahratia súboru do BLOB cez ORDAudio až po uloženie vlastností zvukového súboru do tabuľky vlastností. Po ukončení všetkých procedúr sú dané premenné určené pre prácu s databázou vymazané a inicializované na počiatočnú teda nulovú hodnotu.

A green rectangular button with rounded corners and a white border, containing the word "Spät" in white text.

Vaša požiadavka na vloženie súboru bola úspešná

Obrázok 27. Potvrdzujúca hláška pri správnom nahratí zvukového záznamu.

#### 8.1.2.1 Konfigurácia PHP

Pri spracovaní a ukladaní väčších zvukových alebo iných dát pomocou PHP nastáva problém s obmedzeniami, ktoré je treba upraviť na požadovanú hodnotu. Upravené hodnoty v konfiguračnom súbore *php.ini* sú:

- *max\_execution\_time* – čas, ktorý je potrebný na vykonanie skriptu
- *post\_max\_size* – maximálna veľkosť POST dát, ktorá je povolená
- *upload\_max\_filesize* - maximálna povolená veľkosť pre súbor

#### 8.1.3 Spracovanie vlastností zvukových súborov

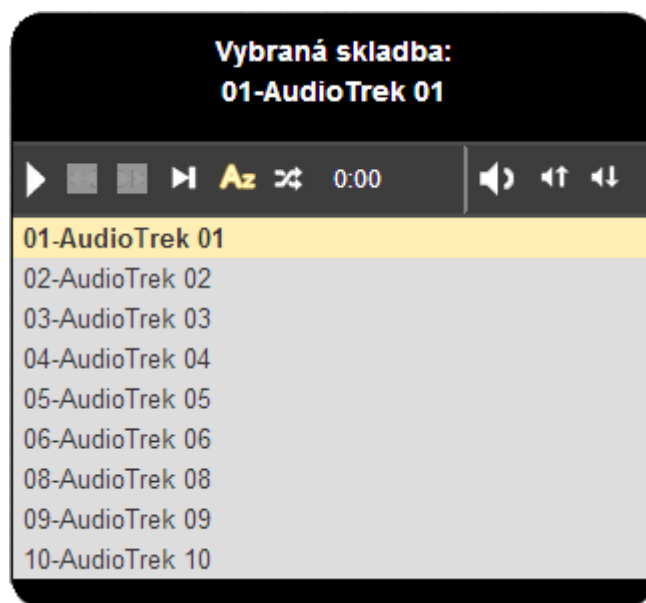
Po úspešnom nahratí zvukového záznamu do tabuľky vykoná aplikácia výpis vlastností do tabuľky na stránke vlastnej tvorby kapely. Pre každú kapelu je vytvorená vlastná *session*. Jediný rozdiel sa nachádza na hlavnej stránke, kde je v sekcii *zoznam koncertov* použitá celková tabuľka so všetkými vytvorenými koncertami pre jednoduchšiu orientáciu. Výpis jednotlivých vlastností prebieha vo forme tabuľky, do ktorej sú vypísané jednotlivé vlastnosti ako sú napr. názvy nahrávok, veľkosť súborov alebo dĺžka demo nahrávky. Veľkosť súborov je upravená na Mb a dĺžka demo nahrávky ako minúty.

Názov skladby	Veľkosť (Mb)	Kódovanie	Počet kanálov	Dĺžka nahrávky (min)
01-AudioTrek 01.mp3	2,5	LAYER3	2	2,7
02-AudioTrek 02.mp3	2,7	LAYER3	2	2,9
03-AudioTrek 03.mp3	2,1	LAYER3	2	2,3
04-AudioTrek 04.mp3	3,8	LAYER3	2	4,2

Obrázok 28. Výpis jednotlivých vlastností zo zvukových súborov.

## 8.2 Audio prehrávač

Pre prezentáciu zvukových súborov, ktoré boli vložené a spracované pomocou databázy, bol vytvorený zvukový prehrávač, ktorý je dostupný každému užívateľovi bez rozdielu na typ účtu. Tento prehrávač sa nachádza na stránke každej kapely. Každá kapela má tak možnosť prezentácie svojej tvorby priamo na svojej profilovej stránke.



Obrázok 29. Audio prehrávač vo webovej aplikácii

### 8.2.1 Načítanie demo nahrávok

Pre načítanie jednotlivých zvukových záznamov do playlistu prehrávača slúži php skript. Audio player načítava zvukové záznamy z umiestnenia *directory*, ktorý bol vytvorený v rámci databázy Oracle. Ako kontrolná hodnota v skripte slúži názov demo nahrávky, podľa ktorej je prechádzaná zložka s jednotlivými demo nahrávkami, ktoré sú porovnávané s názvom z tabuľky a zároveň pri zhode názvov vypisované do playlistu.

Pre krajší vzhľad názvu je z celého názvu odstránená zvuková prípona. Priložená ukážka časti zdrojového kódu zobrazuje prechádzanie *directory* a vypisovanie názvov demo nahrávky bez prípony do playlistu.

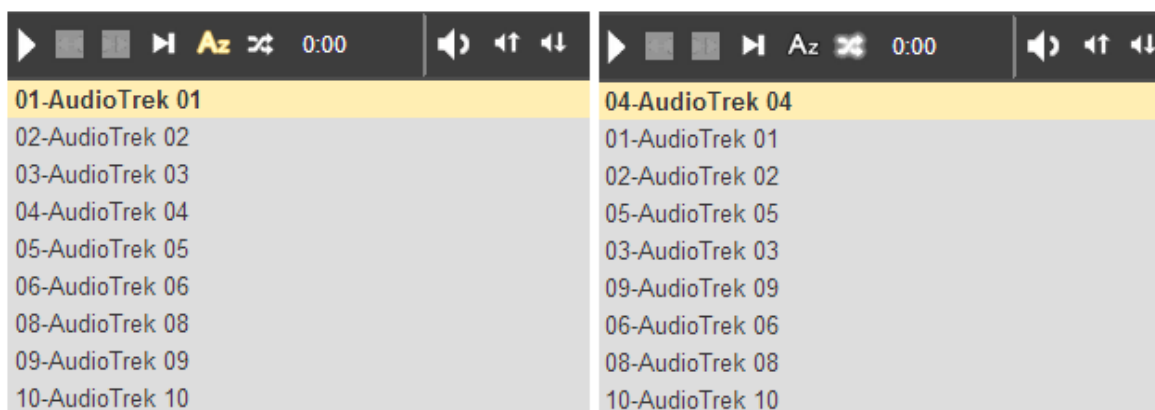
```
foreach($songlist as $key => $val){
    <li>
        <a href = "mp3/<?php echo $val ?>" data-format = "mp3">
            <?php echo substr($val,0,strlen($val)-4) ?>
        </a>
    </li>
```

### 8.2.2 Ovládacie prvky

Audio prehrávač obsahuje základné ovládacie prvky *prehrať*, *zastaviť*, *pauza*, *posunutie o 15 minút dopredu/dozadu* a *preskočenie na ďalšiu skladbu*. Ovládanie hlasitosti sa nachádza v pravej hornej časti prehrávača a sú tu možnosti *vypnúť/zapnúť hlasitosť* a *zvýšenie/zníženie hlasitosti* o jeden stupeň. Medzi špeciálne funkcie prehrávača patrí funkcia *shuffle*. Táto funkcia vykonáva náhodné prehádzanie jednotlivých nahrávok, čím je umožnené náhodné prehrávanie skladieb. Druhá funkcia je zároveň opačnou funkciou k funkcii *shuffle*. Vracia zoradenie skladieb do pôvodnej podoby.



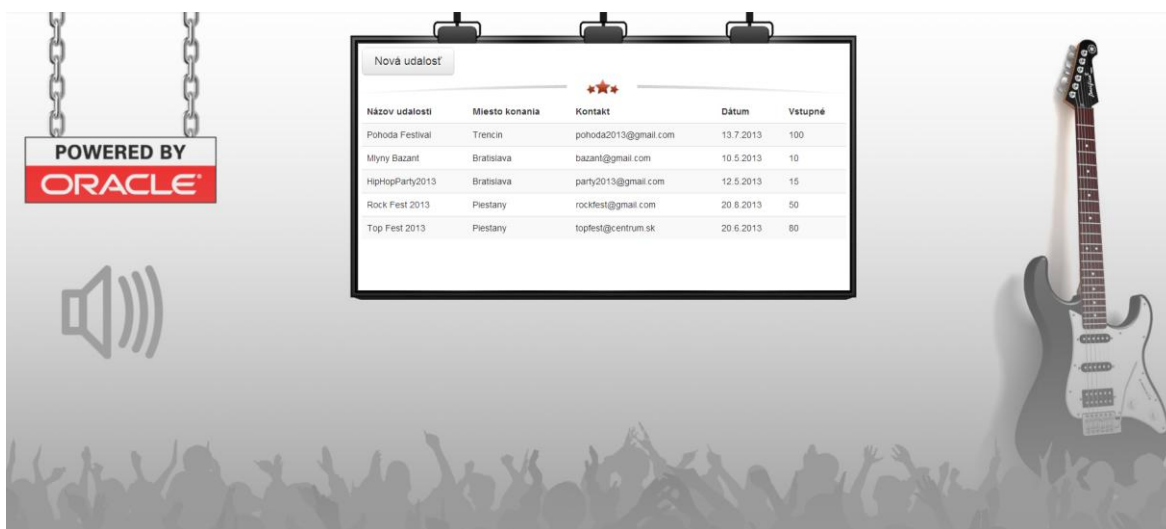
Obrázok 30. Funkcie audio prehrávača



Obrázok 31. Prostredie audio prehrávača so zapnutou funkciou *shuffle*

### 8.3 Organizácia koncertov

Organizovanie udalostí a koncertov patrí medzi hlavné funkcie aplikácie. Každý správca kapely má po prihlásení možnosť vytvorenia vlastnej udalosti. Prehľad jednotlivých koncertov kapely možno zhliaďnúť na ich vlastných profilových stránkach alebo na hlavnej stránke webovej aplikácie, kde má možnosť každý užívateľ vidieť všetky dostupné koncerty pre všetky kapely.



Obrázok 32. Vzhľad webovej stránky pre organizáciu koncertov

Výpis jednotlivých koncertov je vo forme tabuľky. Jednotlivé vstupy vo formulári sú ošetrené proti napadnutiu databázy. Tieto ošetrenia voči SQL Injection sú popísané nižšie v kapitole 9.4.

Nová udalosť

Názov udalosti	Miesto konania	Kontakt	Dátum	Vstupné
Pohoda Festival	Trencin	pohoda2013@gmail.com	13.7.2013	100
Mlyny Bazant	Bratislava	bazant@gmail.com	10.5.2013	10
HipHopParty2013	Bratislava	party2013@gmail.com	12.5.2013	15
Rock Fest 2013	Piestany	rockfest@gmail.com	20.8.2013	50
Top Fest 2013	Piestany	topfest@centrum.sk	20.6.2013	80

Obrázok 33. Výpis jednotlivých udalostí

## 9 OŠETRENIE A ZABEZPEČENIE ZDROJOVÉHO KÓDU WEBOVEJ APLIKÁCIE

Zabezpečenie webovej aplikácie patrí v dnešnej dobe medzi veľmi dôležitú požiadavku, na ktorú je kladený čoraz väčší a väčší dôraz pri tvorbe webovej aplikácie. Objavujú sa stále nové typy a druhy útokov, či už na webové stránky alebo na rôzne databázové systémy.

Pri zabezpečení našej webovej aplikácie som sa zamerlal hlavne na ošetrovanie vstupov a na odosielanie dát na výstup z aplikácie. Pri ošetrovaní zdrojového kódu proti XSS sú využívané dve metódy. Pri zabezpečení databázového systému ORACLE sa využíva špeciálna metóda „bind“. K zabezpečeniu nepochybne patrí aj konfigurácia jednotlivých konfiguračných súborov. Pri PHP je to *php.ini* a pri zabezpečení servera APACHE sa použil „skrytý“ súbor *.htaccess*.

### 9.1 Konfigurácia *php.ini*

Konfiguračný súbor sa nachádza v koreňovom adresári PHP. Pomocou jeho konfigurácie a zmeny nastavenia požadovaných hodnôt môžeme pomôcť k lepšiemu zabezpečeniu webovej aplikácie a jej zdrojového kódu. *Tabuľka 2* uvádza prehľad jednotlivých direktív.

*Tabuľka 2. Výpis direktív z konfiguračného súboru PHP*

Direktíva	Popis	Hodnota
<b>allow_url_fopen</b>	Zaobchádzanie s http ftp ako so súborom	Off
<b>error_reporting</b>	úroveň podrobností zobrazovaných chýb	E_ALL
<b>log_errors</b>	Logovanie chýb	On
<b>display_errors</b>	Zobrazovanie chybových hlášok	Off - production On - development
<b>register_globals</b>	Vytvorenie globálnych premenných zo vstupných dát	Off
<b>magic_quotes_gpc</b> <sup>1</sup>	spúšťa addslashes na GET, POST ...	Off

<sup>1</sup> Magic\_quotes\_gpc - Direktíva magic\_quotes\_gpc sa neodporúča v novších verziách PHP zapínať.

## 9.2 Konfigurácia webového servera Apache (.htaccess)

Medzi metódy zabezpečenia webovej aplikácie a webového servera môžeme taktiež zaradiť konfiguračný súbor *.htaccess*. Popri optimalizácií a čistých URL dokáže tento konfiguračný súbor vykonať určitú formu zabezpečenia.

Prvou funkciou, ktorá je aplikovaná na webovú aplikáciu je funkcia zakázania zobrazovania súborov s príponou *.txt* alebo ich prípadné stiahnutie.

```
<filesmatch "\.(txt)$">
order allow,deny
deny from all
</filesmatch>
```

Druhou funkciou je funkcia ochrany proti spam botom, kde konfiguračný súbor zistí infikáciu a odkáže ich na stránku *spampoison.com*.

```
RewriteCond %{HTTP_USER_AGENT} Wget [OR]
RewriteCond %{HTTP_USER_AGENT} CherryPickerSE [OR]
RewriteCond %{HTTP_USER_AGENT} CherryPickerElite [OR]
RewriteCond %{HTTP_USER_AGENT} EmailCollector [OR]
RewriteCond %{HTTP_USER_AGENT} EmailSiphon [OR]
RewriteCond %{HTTP_USER_AGENT} EmailWolf [OR]
RewriteCond %{HTTP_USER_AGENT} ExtractorPro
RewriteRule ^.*$ http://www.spampoison.com/ [L]
```

### 9.2.1 SEO a clean URL

Súbor *.htaccess* slúži hlavne pri úprave URL. Pre väčšinu užívateľov pôsobí takáto upravená URL dobre a je prehľadnejšia. V konfiguračnom súbore sa pre prepísanie URL používa príkaz `mod-rewrite`. Pre správnu funkčnosť tohto módu je potrebné zapísať na začiatok dokumentu nasledujúce pravidlá:

```
RewriteEngine on // RewriteBase /
```

Prvé pravidlo umožňuje používanie módu pre prepis. Druhé pravidlo nastavuje hlavný adresár, z ktorého budú čerpané a prepisované jednotlivé súbory/prípony. Toto pravidlo som využil pre prepis prípony *.php* pre krajší vzhľad danej URL v prehliadači. Nasledujúci zdrojový kód prepíše (odstráni) všetky prípony *.php* zo súborov:

```
RewriteRule ^([\-,a-z,0-9,_]+) $ /pages/$1.php [L]
RewriteRule ^([\-,a-z,0-9,_]+)\/$ pages/$1.php [L]
```

### 9.3 Cross-site scripting (XSS)

Ošetrovanie zdrojového kódu webovej aplikácie z pohľadu XSS je možné pomocou dvoch metód. Prvou metódou je použitie špeciálnej funkcie `htmlspecialchars()`. Táto funkcia vykoná konverziu špeciálnych znakov na HTML entity. Jednotlivé špeciálne znaky sú popísané v Tabuľke 3.

Tabuľka 3. Špeciálne znaky používané pri použití `htmlspecialchars`.

Znak	Kódovanie
<	&lt
>	&gt
&	&amp
"	&quot
'	&apos
(	&#40
)	&#41
#	&#35
%	&#37
;	&#59
+	&#43
-	&#45

Nasledujúci kód zobrazuje použitie funkcie `htmlspecialchars` pri odosielaní vstupných hodnôt z formulára HTML na výstup. Pri poslednej časti kódu je použitá funkcia (`int`). Funkcia `int` sa používa len v tých prípadoch, keď sme si istí dátovým typom `number`, ktorý je definovaný v databáze Oracle.

## Tvoj koncert

Názov udalosti

Miesto konania

Dátum konania udalosti

Kontakt

Vstupné

Zavrieť

Obrázok 34. Formulár, pri ktorom sa ošetrujú zadávané hodnoty.

Vstupný reťazec v podobe skriptu pre kontrolu XSS:

```
'"><SCRIPT>alert(\\'XSS !\\')</SCRIPT>'
```

Použitie zabezpečenia:

```
$kontakt_1 = htmlspecialchars($_POST["kontakt"]);
```

```
$nazov_1 = htmlspecialchars($_POST["nazov"]);
```

```
$datum_1 = htmlspecialchars($_POST["datum"]);
```

```
$vstupne_1 = (int)$_POST["vstupne"];
```

Pri vložení skriptu pre kontrolu náchylnosti na XSS, uvedený kód v podobe skriptu vypíše potencionálne nebezpečný vstup. Avšak znaky „<“ a „>“ sú nahradené HTML entitami a text je spracovávaný ako text. Tieto HTML tagy sú nahradené špeciálnymi znakmi z Tabuľky 3.

Zápis upraveného výstupu:

```
'&quot;&gt;&lt;SCRIPT&gt;alert(\'XSS!\')&lt;/SCRIPT&gt;';
```

Druhou nemenej účinnou metódou pri ošetrovaní zdrojového kódu je metóda *strip\_tags*. Na rozdiel od predchádzajúcej metódy, ktorá nahradzuje HTML tagy vybranými špeciálnymi znakmi z tabuľky, táto metóda odstráni z reťazca všetky HTML a PHP tagy.

Zápis upraveného výstupu podľa *strip\_tags*:

```
">alert(\'XSS utok!\')
```

### 9.3.1 Zabezpečenie z pohľadu užívateľa

Zabezpečenie webovej aplikácie nevykonáva len programátor. Z pohľadu užívateľa je to hlavne nastavenie a zabezpečenie internetového prehliadača. Vypnutie javascriptu v prehliadači je jednou z týchto možností zabezpečenia. Avšak pri vypnutí javascriptu nastáva problém pri zamedzení a nefunkčnosti všetkých funkcií webovej aplikácie, ktorá využíva javascript. Ďalším oveľa účinnejším spôsobom je využitie doplnkov vo webovom prehliadači. Medzi prehliadače, ktoré používajú túto podporu patrí Google Chrome. Medzi doplnky, ktoré slúžia na ochranu proti XSS patrí SCRIPT NO, NOTSCRIPT atď.

## 9.4 SQL Injection

Táto metóda využíva nedostatky, ktoré vznikajú pri kontrole vstupných dát. Pri metóde SQL Injection môže hacker ovládnuť celú databázu a tým aj vymazať celú štruktúru alebo len pozmeniť alebo zneužiť získané dáta. Ide o tzv. zmenenie SQL Query, cez nezabezpečený vstup. Pomocou univerzálneho zápisu `OR 1 = 1`, môžeme otestovať náchylosť na SQL Injection:

```
``SELECT * FROM USERS WHERE ID = 4' OR '1' = '1``
```

Pre zabezpečenie databázy Oracle sa používa metóda BIND argumentov. Táto metóda je taktiež dôležitá pre výkon databázy Oracle. Bindovanie znižuje náchylnosť na SQL Injection pretože dáta, ktoré sú priradené s premennou bindu nie sú nikdy považované za SQL príkaz.

Použitie zabezpečenia vo webovej aplikácii:

```
$s = oci_parse($c, "insert into dp_udalost values
(udalost.nextval, :bind1, :bind2, :bind3, :bind4, :bind5)");
oci_bind_by_name($s, ":bind1", $kontakt_1);
oci_bind_by_name($s, ":bind2", $nazov_1);
oci_bind_by_name($s, ":bind3", $miesto_konania_1);
oci_bind_by_name($s, ":bind4", $datum_1);
oci_bind_by_name($s, ":bind5", $vstupne_1);
```

## ZÁVER

Cieľom tejto záverečnej práce bolo predviesť pokročilé techniky v databázovom systéme Oracle pre prácu s multimediálnymi dátami. Tieto techniky sú prezentované na vytvorenej webovej aplikácii pre začínajúce kapely, ktorá využíva databázový systém Oracle a jej špeciálne funkcie. Využitie sú hlavne funkcie pre prácu s obrázkovými a zvukovými dátami. Skôr ako bola vytvorená samotná webová aplikácia, bola navrhnutá a namodelovaná štruktúra aplikácie pomocou jazyka UML. Webová časť aplikácie je vytvorená pomocou skriptovacieho jazyka PHP, ktorý má široké možnosti pri spracovaní a komunikácii s databázou Oracle. Využíva hlavne rozhranie OCI, ktoré je určené priamo pre komunikáciu s databázou.

Podstatnou nevýhodou pri spracovaní multimediálnych dát je v tom, že databáza sama o sebe tieto dáta nedokáže zobrazit' do výstupnej podoby. Preto je potrebný server, ktorý dokáže držať alebo ukladať tieto dáta. V našom prípade bol použitý ako odkladací priestor pevný disk počítača a tým aj beh aplikácie na localhoste. Zároveň sa nám podaril vyriešiť problém s častými problémami, ktoré nastávajú pri prepájaní webového servera Apache, jazyka PHP a databázy Oracle. Celý postup riešenia prepojenia je popísaný v priloženej príručke.

Medzi hlavné výhody určite patrí spracovanie informácií a metadát či už z obrázkov alebo z audio súborov. Informácie obsiahnuté v súboroch bližšie charakterizujú daný súbor a poskytujú tak viacero možností pri jeho spracovaní. Výhodou pri práci s informáciami a metadátami je ich nízka náročnosť na tabuľkový priestor alebo inak metadáta stačí len zobrazit' pomocou špeciálnych príkazov priamo vo webovej aplikácii.

Zabezpečenie webovej aplikácie prebiehalo vo forme ošetrovania vstupov a hlavne výstupov z webovej aplikácie. Zdrojový kód bol zabezpečený proti XSS a databázový systém proti SQL Injection so špeciálnymi Bindovacími premennými. Webový server Apache bol taktiež zabezpečený pomocou konfiguračného súboru .htaccess.

Zadanie diplomovej práce a určené špecifikácie sa mi podarilo splniť. Všetky funkcie webovej aplikácie vrátane zabezpečenia a procedúr databázy Oracle sú funkčné.

## ZAVĚR V ANGLIČTINĚ

The aim of this thesis was to demonstrate advanced techniques in Oracle database for handling multimedia data. These special techniques are presented in web application for beginning bands, which uses the Oracle database and its special features. There are mainly used functions for working with pictures and audio data. There were designed and modeled structure of the application using UML before the whole creation process of web application. Web part of the application is developed with PHP scripting language, which has a wide range of process and communication use with the Oracle database. PHP mainly uses OCI interface, which is designed to communicate directly with the database.

The main disadvantage of processing multimedia data is that the database itself can't display the multimedia data into the final form. For this purpose exist only one solution and that is the need of server or local storage that can keep or store these data. In our case, there was used a hard drive for storing the data and thus the application is running on localhost. At the same time we managed to solve the problems, which occur when we are tried to link the Apache web server, PHP and Oracle database. The whole process for handling of these connections is described in attached manual.

The main advantages are definitely all about the processing of the information and metadata either from images or audio files. The informations contained in these files further characterize and provide so many opportunities during processing. The next advantage of working with these informations and metadata is in their low intensity on a table space or other metadata you just viewed using special commands directly in a web application.

The security of web application was processed in the form of input and output treatment. The source code has been secured against XSS and the Oracle database system against SQL Injection with special Bind variables. Apache web server was also ensured by its configuration file .htaccess.

I successfully accomplished all the specifications and terms, which was in the assignment of this thesis. All functions including web application security and Oracle database precedures are functional.

**ZOZNAM POUŽITEJ LITERATÚRY**

- [1] *Úvod do softwarového inženýrství* [online]. Ostrava, 2002 [cit. 2013-04-15]. Dostupné z: <http://vyuka.fai.utb.cz/mod/resource/view.php?id=3194>. Skripta. Technická univerzita Ostrava.
- [2] UML a unifikovaný proces vývoje aplikací. první. Brno: Computer Press, 2003. ISBN 80-7226-947-X.
- [3] ZEMEK, Lukáš. *Bezpečnost webových aplikací*. Zlín, 2012. Bakalárska práca. UTB.
- [4] KVOČKA, Martin. *Útoky typu SQL injection*. Brno, 2009. Bakalárska práca. Masarykova univerzita.
- [5] PROCHÁZKA, David. Oracle: průvodce správou, využitím a programováním nad databázovým systémem. 1. vyd. Praha: Grada, 2009, 168 s. ISBN 978-80-247-2762-2.
- [6] ORACLE. *Oracle® Multimedia: User's Guide* [online]. 1999-2007 [cit. 2013-04-22]. ISBN B28415-02. Dostupné z: [http://docs.oracle.com/cd/E15312\\_01/appdev.112/e10621.pdf](http://docs.oracle.com/cd/E15312_01/appdev.112/e10621.pdf)
- [7] PL/SQL. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2013-05-22]. Dostupné z: <http://en.wikipedia.org/wiki/PL/SQL>
- [8] Kaskádové štýly. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2013-05-22]. Dostupné z: [https://cs.wikipedia.org/wiki/Kask%C3%A1dov%C3%A9\\_styly](https://cs.wikipedia.org/wiki/Kask%C3%A1dov%C3%A9_styly)
- [9] GUI. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2013-05-23]. Dostupné z: [http://sk.wikipedia.org/wiki/Grafick%C3%A9\\_pou%C5%BE%C3%ADvate%C4%BESk%C3%A9\\_rozhranie](http://sk.wikipedia.org/wiki/Grafick%C3%A9_pou%C5%BE%C3%ADvate%C4%BESk%C3%A9_rozhranie)

**ZOZNAM POUŽITÝCH SYMBOLOV A ZKRATIEK**

- STAG informačný systém pre študijnú agendu používaný na niektorých vysokých školách.
- OMG Object Management Group je medzinárodné, voľne prístupné a neziskové štandardizačné konzorcium v oblasti počítačového priemyslu.
- GUI Grafické používateľské rozhranie (Graphical User Interface), umožňuje ovládať zariadenie pomocou súboru interaktívnych obrazových prvkov. [9]
- UML Unified Modeling Language, je grafický jazyk na vizualizáciu a navrhovanie programových systémov.
- IEEE Institute of Electrical and Electronics Engineers, je inštitút pre elektrotechnické a elektronické inžinierstvo.
- JSON JavaScript Object Notation je spôsob zápisu dát nezávislý na počítačovej platforme, určený pre prenos dát.
- JAI Java Advanced Imaging, rozšírenie platformy Java, ktoré obsahuje súbor objektovo orientovaných rozhraní podporujúcich jednoduchý programovací model. Pomocou tohto modelu majú možnosť spravovať obrázkové súbory.
- JMF Java Media Framework, knižnica umožňujúca audio, video a ostatným médiám možnosť ich pridávania do Java aplikácií a apletov.
- OCI Oracle Call Interface, pozostáva z nástrojov API jazyka C, ktorý poskytuje rozhranie pre databázu Oracle.

**ZOZNAM OBRÁZKOV**

<i>Obrázok 1. Sémantický diagram vývoja softvérového produktu</i> .....	13
<i>Obrázok 2. Jednotlivé pracovné postupy iterácie zobrazené do koláčového grafu</i> .....	16
<i>Obrázok 3. Všeobecná mechanika jazyka UML [2, s. 11]</i> .....	18
<i>Obrázok 4. Zisk informácií v podobe dotazníka (ukážka zberu dát)</i> .....	19
<i>Obrázok 6. Ukážka scenára prípadu použitia v programe Enterprise Architect</i> .....	21
<i>Obrázok 7. Ukážka sekvenčného diagramu</i> .....	22
<i>Obrázok 8. Aktivitné diagramy a ich prevedenie</i> .....	23
<i>Obrázok 9. Graf najčastejšie sa vyskytujúcich útokov na webové stránky</i> .....	24
<i>Obrázok 10. Pohľad na priebeh útoku pomocou XSS</i> .....	27
<i>Obrázok 11. Architektúra multimedialnej vrstvy Oracle [6]</i> .....	31
<i>Obrázok 12. Zobrazenie výsledkov dotazníkového prieskumu</i> .....	39
<i>Obrázok 13. Pracovné prostredie vývojovej aplikácie Enterprise Architect</i> .....	40
<i>Obrázok 14. Náhľad funkčného požiadavku pre Fotoreport</i> .....	44
<i>Obrázok 15. Náhľad funkčného požiadavku pre Hudobný prehrávač</i> .....	44
<i>Obrázok 16. Náhľad funkčného požiadavku pre organizáciu koncertov</i> .....	44
<i>Obrázok 17. Náhľad na aktérov systému webovej aplikácie</i> .....	46
<i>Obrázok 18. Pohľad na Use Case Diagram z programu Enterprise Architect</i> .....	47
<i>Obrázok 19. Metóda slovného popisu vnútra prípadu užitia (EA)</i> .....	48
<i>Obrázok 20. Metóda slovného popisu prípadu užitia a alternatívne cesty (EA)</i> .....	48
<i>Obrázok 21. Aktivitný diagram pre vytvorenie udalosti (EA)</i> .....	49
<i>Obrázok 22. Diagram tried (EA)</i> .....	50
<i>Obrázok 23. Ukážka sekvenčného diagramu pre pridanie udalosti (EA)</i> .....	51
<i>Obrázok 24. Úvodná stránka webovej aplikácie</i> .....	52
<i>Obrázok 25. Horné menu pre správcu kapely po prihlásení</i> . .....	52
<i>Obrázok 26. Náhľad na hlavnú stránku pre správu demo nahrávok</i> . .....	53
<i>Obrázok 27. Formulár pre vybratie a nahratie zvukového záznamu</i> . .....	55
<i>Obrázok 28. Potvrdzujúca hláška pri správnom nahratí zvukového záznamu</i> . .....	56
<i>Obrázok 29. Výpis jednotlivých vlastností zo zvukových súborov</i> . .....	57
<i>Obrázok 30. Audio prehrávač vo webovej aplikácii</i> .....	57
<i>Obrázok 31. Funkcie audio prehrávača</i> .....	58
<i>Obrázok 32. Prostredie audio prehrávača so zapnutou funkciou shuffle</i> .....	58

*Obrázok 33. Vzhľad webovej stránky pre organizáciu koncertov ..... 59*

*Obrázok 34. Výpis jednotlivých udalostí ..... 59*

*Obrázok 35. Formulár, pri ktorom sa ošetrujú zadávané hodnoty. .... 63*

**ZOZNAM TABULIEK**

<i>Tabuľka 1. Ukážka prevedenie escapingu v HTML kóde .....</i>	<i>28</i>
<i>Tabuľka 2. Výpis direktív z konfiguračného súboru PHP .....</i>	<i>60</i>
<i>Tabuľka 3. Špeciálne znaky používané pri použití htmlspecialchars. ....</i>	<i>62</i>

## **ZOZNAM PRÍLOH**

Zdrojový kód aplikácie a jednotlivé skripty PL/SQL sú priložené na priloženom CD.

Užívateľská príručka na prepojenie Apache, PHP a OCI na priloženom CD.

Kompletný dotazníkový prieskum a grafy na priloženom CD.