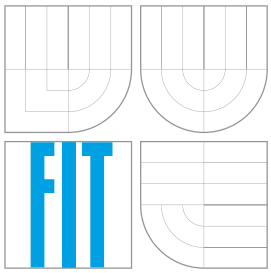
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

UKLÁDÁNÍ A VYHLEDÁVÁNÍ ROZSÁHLÝCH SLOVNÍKOVÝCH DATABÁZÍ PRO MOBILNÍ ZAŘÍZENÍ NA PLATFORMĚ ANDROID

SEARCH IN LARGE DICTIONARY DATABASES FOR MOBILE DEVICES ON ANDROID PLATFORM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

MATÚŠ KRŠKA

VEDOUCÍ PRÁCE
SUPERVISOR

doc. RNDr. PAVEL SMRŽ, Ph.D.

BRNO 2016

Abstrakt

Práce se zabývá studií ukládání slovníkových dat ve formátu LMF a práci s nimi. Analyzuje možnosti zpracování souborů XML, ve kterých data LMF obvykle bývají uložené. Práce hodnotí existující formy řešení na platformě Android. Předkládá vlastní řešení v podobě Android aplikace, které vychází z navrženého datového a funkčního modelu. Implementace se soustřeďuje na vyřešení problémů spojených s vyhledáváním víceslovných spojení a kontextových příkladů použití a vytvoření jednoduchého grafického rozhraní vhodného na zobrazování výsledků vyhledávání. Práce hodnotí dosažené výsledky na základě testů vykonaných na reálných uživatelích.

Abstract

This work studies persisting of dictionary data in LMF format and means of working with it. It analyses options for parsing XML files in which the LMF data is usually stored. This work evaluates existing solutions on the Android platform. It presents its own solution in form of an Android application which is based on presented data and function models. Implementation is focused on resolving problems of multi-word expression searching, context example searching and creation of simple graphical user interface suitable for displaying search results. This work evaluates achieved results based on tests performed on real users.

Klíčová slova

LMF, XML, Android, slovníkové databáze, Java, SQLite

Keywords

LMF, XML, Android, dictionary databases, Java, SQLite

Citace

Matúš Krška: Ukládání a vyhledávání rozsáhlých slovníkových databází pro mobilní zařízení na platformě Android, bakalářská práce, Brno, FIT VUT v Brně, 2016

Ukládání a vyhledávání rozsáhlých slovníkových databází pro mobilní zařízení na platformě Android

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Doc. RNDr. Pavla Smrže, Ph.D. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Matúš Krška
16. května 2016

Poděkování

Rád bych poděkoval vedoucímu práce panu Doc. RNDr. Pavlu Smržovi, Ph.D. za cenné rady, tipy a připomínky, které mi značně pomohli při psaní bakalářske práce.

Obsah

Kapitola 1

Úvod

V našej spoločnosti má takmer každý prístup k rýchlym informáciam. Je to dôsledkom vývoja informačných technologií a digitalizáciou takmer všetkých aspektov spoločnosti. V posledných rokoch došlo najmä k rapidnému vývoju osobných zariadení a chytrých telefónov. Pri každom ohliadnutí okolo seba vo verejnosti je vidieť aké percento ľudí vlastní a aktívne využíva dané zariadenia.

Elektronický svet sprevádza človeka v podstate počas celého dňa. Keď vstane skontroluje predpoveď počasia, poštu alebo najnovšie správy. Cestou do práce alebo školy vyhľadáva lokáciu potrebných spojov hromadnej dopravy, číta knihu, počúva hudbu. V škole alebo práci často využíva určitu formu elektronických nástrojov a pomôcok, bez ktorých si už často nevie svoju činnosť ani predstaviť. Vo volnom čase potom relaxuje pri videách, filmoch, seriáloch, hrách alebo využíva sociálne siete tak ako častokrát inokedy v rámci dňa. Sociálne siete samotné sú takmer neoddelitelnou súčasťou sociálneho života hlavne mladých ľudí a dôkazom informatizácie spoločnosti.

Každá informácia je k dosahu, je však dôležité jú správne spracovať a vhodne s ňou naložiť. Často sú informácie k dispozíci len v cudzom jazyku, najmä v angličtine, ktorá je rozšírená po celom svete, a je teda kľúčove ovládať minimálne základy. Keď však treba, nie je náročne vyhladať si výrazy v rôznych podobách slovníka, často elektronickej. Existuje mnoho foriem, populárnou však je najmä aplikácia pre chytrý telefón.

Najznámejším a najrozšírenejším systémom pre chytré telefóny je Android, ktorý zažil od svojho predstavenia v roku 2007 obrovský nárast popularity. Oproti konkurencii má výhodu hlavne v slobode výberu a rôznorodosti, ktorá sa nedá porovnávať s žiadným iným systémom. Vyvinuli sa naň nespočetné množstvo aplikácií, medzi ktoré patria aj slovníky. Slovníky môžu byť jednojazyčné výkladové alebo častejšie viacjazyčné prekladové. Pre každodenné a praktické využite je slovník prekladový, ktorý umožňuje vyhladať význam výrazov a fráz jednoho jazyka v druhom. Pre Android existujú napríklad aplikácie HandyLex, DIC-o Czech-English Dicionary alebo Aard, každá so svojimi výhodami a nevýhodami. Spoločným problémom však býva neúplna alebo žiadna podpora vyhľadávania viacslovných spojení a zobrazenie výrazu v kontexte. Preto sa táto práca zaoberá analýzou problematiky a návrhom, implementáciou a testovaním slovníka pre systém Android, ktorý by tieto nedostatky odstránil.

Kapitola 2

Analýza

2.1 Lexical Markup Framework

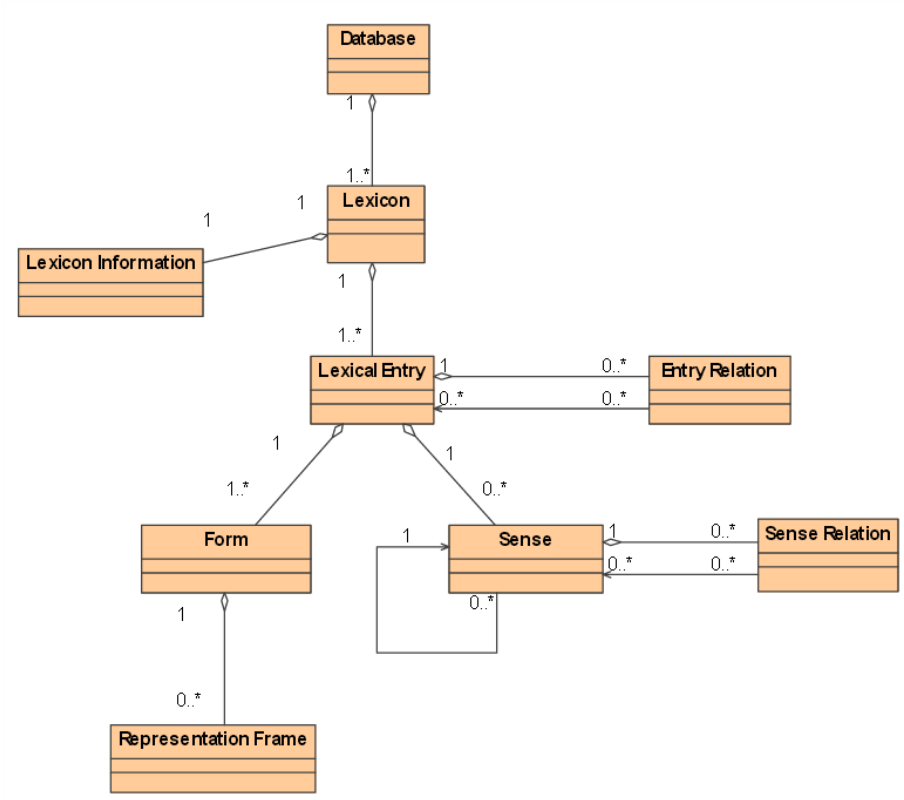
Základnou a nevyhnutnout súčasťou problematiky elektronických slovníkov je uchovávanie obsahu jazyka. Bez správneho systému perzistencie takých komplexných a rozmanitých dát je zvládnutie problému spracovania ľudského jazyka všeobecne nemožné.

Jedným z takýchto systémov je *Lexical Markup Framework* (ďalej len LMF). LMF je štandardizovaný, flexibilný a sofistikovaný systém, ktorý sa používa na popis obsahu prirodzeného jazyka a prípravu na jeho strojové spracovanie. Tento systém bol v roku 2008 uvedený ako medzinárodný štandard organizáciou ISO pod číslom 24613, oficiálny dokument je k dispozícii na oficiálnych internetových stránkach ISO [4]. Za úlohu si dáva poskytnúť spoločný model pre vytvorenie respektíve použitie lexikálnych zdrojov, spravovanie výmeny dát medzi týmito zdrojmi a spájanie takýchto zdrojov do rozsiahlých slovníkových databázi. Štruktúra LMF podporuje ako jednojazyčné tak aj dvoj a viacjazyčné slovníky. Model je schopný popísať všetky aspekty prirodzeného jazyka ako je mofológia, syntax, sémantika a preklad. Reprezentácia takéjto štruktúry býva typicky uložená v súboroch XML.

Štruktúra LMF je zložená z balíkov a to balíka základného (*Core package*) a balíkov rozširujúcich základný balík (*Extensions*).

2.1.1 Core package

Základný balík je jadrom štruktúry, ktoré popisuje elementárne vzťahy informácií v lexikálnom zázname. Takáto štruktúra sa dá popísať pomocou tried a vzťahmi medzi nimi. Základnou triedou je trieda databáza (*Database*). Databáza reprezentuje podmožinu všetkých slovníkov, obsahuje teda jednu alebo viacero tried slovník (*Lexicon*). Slovník obsahuje elementárne, administrátivne atributy a iné všeobecné atributy. Slovník potom zastrešuje všetky lexikálne záznamy (*Lexical Entry*). Trieda lexikálneho záznamu spravuje najvyššiu úroveň rozdelenia jazyka, teda reprezentuje slová a základne slovné spojenia v jazyku. Slovník je týmto spôsobom definovaný ako množina všetkých základnych slovných elementov jazyka. Trieda záznamu pozostáva z tried forma (*Form*) a zmysel (*Sense*), pričom forma obsahuje textový reťazec reprezentujúci slovo a zmysel popisuje jeho význam. Ak má slovo viacero textových reprezentácií býva forma spojená s viacerými snímkami reprezentácie (*Represantation Frame*).



Obrázek 2.1: Diagram tried základného balíka LMF [3]

2.1.2 Extensions

Narozdiel od základného balíka ponúkajú rozširujúce baliíky možnosť špecialicácie slovníka. Slovník potom môže byť poskladaný z takej podmnožiny rozširujúcich balíkov, ktorá sa zameriava na určité použitie daného slovníka. Najčastejšie použitia sú NLP a MRD.

NLP balíky (*natural language processing*) sa zameriavajú na jazykovedné spracovanie jazyka. Využívaju sa na sprostredkovanie interakcie medzi človekom a počítačom a riešia také problémy ako porozumenie prirodzeným jazykom, odvodenie významov zo vstupov ľudského jazyka a generovanie prirodzeného jazyka zo strojovej reprezentácie. Medzi takéto balíky patria napríklad balíky pre syntax, sémantiku a aj morfológiu. Morfologický balík rozširuje triedu form zo základného balíka o entity lemma, word form a stem.

MRD balík (*machine-readable dictionary*) sa zasa zameriava, ako názov hovorí, na strojové spracovanie jazyka a hlavným účelom tohto balíka býva vytvorenie slovníkovej databáze pre použitie v elektronických slovníkoch. Tento balík rozširuje základný balík o podstatné triedy ako napr. ekvivalent a kontext. Tieto dve triedy sa viažu na triedu zmysel zo základného balíka.

2.2 Spracovanie XML súborov

Pre výmenú ľubovolných dát je potreba správneho výberu ich reprezentácie. V prípade LMF systému sa vo väčšine prípadov jedná o formát XML. XML, teda rošíritelný značko-

vací jazyk (*Extensible Markup Language*) je jazyk určený na serializáciu dát. Formát XML sa stal jedným z hlavných štandardov pre výmenu dát v situáciach, ktoré vyžadujú veľkú mieru kompatibility, flexibility a škálovateľnosti. Avšak niektoré základne vlastnosti, ako jeho výrečnosť a samopopisné textové XML značky a atributy, spôsobili obavy o výkon spracovania. Na odstránenie tohoto problému boli vyvinuté viaceré postupy na deserializáciu dát uložených v tomto formáte. Všeobecne sa jedná o dva prístupy: založené na načítani XML súboru do stromovej štruktúry a založené na postupnom prúdovom spracovaní XML a udalostí, ktoré pri ňom nastávajú.

Tieto prístupy a ich formy budú v zápätí predstavené bližšie.

2.2.1 Spracovanie typu DOM

Spracovanie typu DOM (*Document Object Model*) je založené na vytvorení stromovej štruktúry objektov, ktoré v sebe držia všetky dáta obsiahnuté v súbore XML. Pri spracovaní DOM prístupom je celý súbor XML načítaný do objektov. Znamená to, že celý obsah súboru je uložený v pamäti a je plne k dispozícií na potrebné účely. Pri prechádzaní súborom XML je spracovaný každý uzol štruktúry a následne je zaradený do stromu na správne miesto.

Tento postup je obvykle zvolený pre dokumenty z menším obsahom dát, pretože spracovanie a vytvorenie takejto stromovej štruktúry môže byť časovo náročné a teda nevhodné. Taktiež väčší obsah dát znamená viac priestoru v pamäti na uloženie objektov reprezentujúcich štruktúru stromu. Treba preto rozumne uvážiť, či takýto spôsob je vhodný pre dané využitie.

Uchovávanie celého obsahu však prináša mnohé výhody, ktoré sa nedajú iným spôsobom nahradiť. Pri viacnásobnom využití dát nie je potreba spracovávať súbor XML odznova. Teda pri potrebe rozličného dotazovania dát je možné rýchle vyhľadávanie v stromovej štruktúre a nájdenie požadovaného výsledku. To je veľká výhoda ak je dôležité mať rýchly ale rozmanitý prístup k dátam uložených v XML.

Obrovskou výhodou je taktiež možnosť manipulácie s dátami, či už sa jedná o ich zmenu alebo preskupenie. Objekty a ich atributy v stromovej štruktúre môžu byť jednoducho menené a každý uzol sa dá zaradiť do iného podstromu, než bol pôvodný stav.

Preto je spracovanie DOM používane hlavné v prípadoch kedy je potreba rozličných operácií s dátami, ich prípadne editovanie a celkovo komplexné spracovanie. Taktiež sa predpokladá, že ich velkosť bude relatívne malá, tak aby nedošlo k obrovskej časovej a systémovej náročnosti.

2.2.2 Spracovanie typu SAX

Alternatívou pre spracovanie modelom DOM je prístup SAX(*Simple API for XML*). Jedná sa o definíciu rozhrania pre prúdové spracovanie XML dokumentov, ktorá využíva detekovania udalostí nastávajúcich pri čítani súboru. Narozdiel od DOM, kde sa načítava obsah celý, spracovanie SAX prechádza každú časť dokumentu sekvenčne.

Pri postupnom čítaní zo súboru sú tvorené SAX udalosti, na ktoré užívatel musí správne zareagovať, teda korektne programovo spracovať. Príkladmi takýchto udalostí sú:

- XML textové pole
- začiatočné XML značky
- ukončovacie XML značky

- XML komenty

Keď načítavanie súboru narazí na nejakú udalosť, napríklad značku <item> , zavolá sa uživateľsky definovaná metóda (ak existuje) pre spracovanie danej značky, ktorá má za úlohu s takouto udalosť správne naložiť. Následne pri nájdení značky </item> je volaná iná metóda, ktorá spracuje takúto ukončovaciu značku. Je teda kompletne na užívateľovi aké akcie vykoná a často sa vyžaduje, aby si sám strážil stav, v ktoróm sa načítavanie práve nachádza. Obvykle teda býva také spracovanie určitou formou automatu.

Výhodou takéhoto prístupu je reálna možnosť spracovania súboru rozličnej veľkosti. Pre špecifiký dotaz je možné spracovať len uzly dokumentu, ktoré su preň podstatné, ostatné sú ignorované. Dochádza teda k podstatnému zjednodušenia časovej zložitosti a pamäťovej náročnosti.

Takýto postup je taktiež vhodný napríklad pre indexovanie, konverziu a formátovanie, teda prípady, kďe je potrebné spracovať súbor len jedenkrát.

Nevýhodou SAX je absencia možnosti pokročilej validácie XML dokumentu. V prípadoch, napríklad, kedy sa jeden uzol odkazuje na druhý, je overenie či odkazovaný uzol existuje nemožné. Takisto nie je možnosť preskupovať a modifikovať dáta.

2.2.3 Spracovanie typu StAX

Podobným typom ako je SAX je aj typ StAX (*Streaming API for XML*). Taktiež sa jedná o prúdove spracovanie XML dokumentu, ktoré využíva udalosti, no od SAX sa líši v tom ako tieto udalosti získava. Kým SAX spracovanie samo prechádza dokument vyvoláva udalosti a pre ne definované metody, StAX spracovanie definuje postup opačný. Definované metody určujú kedy a ako sa ma dokument prechádzať a teda majú kontrolu nad vyvolávaním udalostí.

Keďže metódy kontrolujú prechod dokumentom, je možné spracovanie XML súboru zastaviť v ktoromkoľvek uzle, čo pri spracovaní typu SAX nie je možné. Taktiež je možné jednoducho definovať spracovanie poduzlov vybraného uzla, čo pri SAX je komplikovanejšie, pretože si treba poznačit v ktorom uzle sa spracovanie nachádza a v metódach pre spracovanie daných poduzlov je nutné prihliadať na túto informáciu.

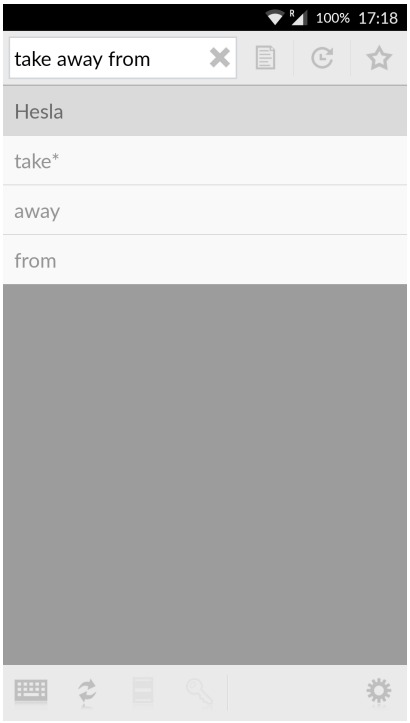
StAX rozhranie takisto umožnuje XML dokument zapisovať, no nepodporuje ani len základnú validáciu XML schém, ktorú SAX zvládne.

2.3 Existujúce riešenia

Na platforme Android je vyvinutých mnoho existujúcich riešení elektronických slovníkov. Každé z nich má svoje silné stránky, ale aj nedostatky.

2.3.1 HandyLex

Séria HandyLex od firmy Lingea je dobre spracovaná kolekcia viacjazyčných slovníkov. Pre každú kombináciu jazykov ponúka Lingea dve verzie a to základnú a tzv. Plus. Najpopulárnejším z nich je anglicko-český slovník v základnej verzii , ktorý je ako jediný zadarmo. Na oficiálnych stránkach spoločnosti Lingea je uvedené, že slovníky obsahujú 35 000 záznamov pre základnú verziu a 100 000 pre verziu plus. V aplikácii samotnej je uvedné, že sa jedná o 18 000 záznamov. Základná verzia však podľa popisu nepodporuje vyhľadávanie viacslovných výrazov, čo je zrejmé pri vyhľadaní výrazu “take away from” (2.2). Pri tomto vyhľadaní ponúkne len heslovité odkazy na jednotlivé slová.



Obrázek 2.2: Vyhľadanie viacslovného výrazu v aplikácii HandyLex

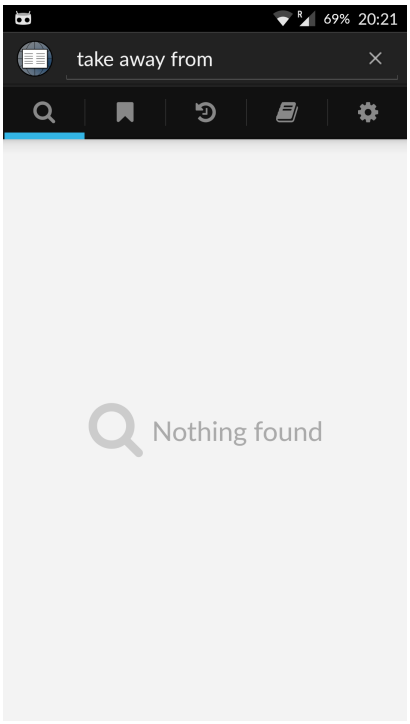
Jednotlivé výsledky vyhľadaných hesiel následne ponúkajú dobre formatovaný a presný zoznam prekladov. Pri každom výsledku je možnosť zobraziť viacej informácií o danom hesle napríklad synonýma, slovné spojenia, odvodené slová a zoznam fráz, avšak nie každé heslo obsahuje všetky informácie. Zaujímavé je takisto uvedenie časovania slovies. Problematické môže byť navigovanie v aplikácií, kde pri obrovskom množstve možností a informacií nie je vždy jednoduchá a intuitívná navigácia na požadovanú stránku s informáciami, ktoré užívateľa zaujímajú.

- Výhody
  1. množstvo informácií
  2. prehladný výpis výsledkov
- Nevýhody
  1. absencia celotextového vyhľadávania
  2. komplikovaná navigácia v aplikácii

2.3.2 Aard2

Ďalším elektronickým slovníkom je aplikácia Aard2 od pána Igora Tkacha. Aplikácia je dostupná z obchodu platformy Android zadarmo, neobsahuje však žiadne slovníkové dáta. Na oficiálnej stránke ponúka autor mnoho odkazov na viaceré slovnikové databáze, ktoré sú k dispozícii užívateľovi na stiahnutie do aplikácie. Slovník podporuje používanie viacerých slovníkov súčasne a po vyhľadaní výrazu zobrazuje heslá v jednotlivých slovníkoch. Po vyhľadaní výrazu teda nie je vidno priamo preklad v danom jazyku, ale len názov slovníku,

v ktorom bol daný kľúč nájdený. To znamená, že pre zobrazenie výsledku prekladu je nutné prejť na ďalšiu stránku výberom jedného kľúča, čo je veľmi nepraktické. Negatívom je takisto absencia akejkoľvek podoby vyhľadávania viacslovných výrazov, príkladom znova spojenie "take away from" (2.3).



Obrázek 2.3: Vyhľadanie viacslovného výrazu v aplikácii Aard2

Dobre je zvládnuté hlavné menu, veľkými pozitívami sú funkcionalita histórie a záložiek. V historii sa zobrazujú naopsledy vyhľadané výrazy, to umožnuje návrat k predošlým vyhľadávaniam aj po dlhšom čase. Záložky ponúkajú uloženie vybraných výrazov, ktoré užívateľa zaujímajú najviac.

- Výhody
  1. veľká ponuka slovníkových dát
  2. história, záložky
- Nevýhody
  1. celotextové vyhľadávanie nie je podporované
  2. zobrazovanie výsledkov

2.3.3 DIC-o

Veľmi populárnou skupinou aplikácií sú slovníký DIC-o od rovnomennej spoločnosti. Medzi ne patrí aj DIC-o Czech-English teda česko-anglický slovník. Po otvorení tohto slovníka je na hlavnej stránke k dispozícii celý zoznam abecedne zoradených hesiel. Po vložení jednotlivých znakov vyhľadávaného slova sa náhľad na zoznam hesiel posunie na správnu pozíciu a zobrazí najbližšie výsledky. Vyhľadávanie je veľmi rýchle a plynulé, výsledky su dostatočne

presné. Nepochopiteľné je však to, že po zozname sa môže užívateľ voľne posúvať smerom dole aj nahor a to neobmedzene, teda po okamihu je náhľad úplne inde, než na vyhľadávanom výraze. Neideálne je takisto vyriešené zoskupovanie prekladov iba pod jedným heslom. Napríklad po vyhľadaní slova “love” je zobrazené jediné heslo ako aj s prekladmi podstatného mena tak aj slovesa.

- Výhody
    1. rýchlosť vyhľadávania
    2. prehľadné menu
- Nevýhody
    1. zoskupovanie prekladov
    2. posuvný zoznam všetkých hesiel

Kapitola 3

Návrh riešenia

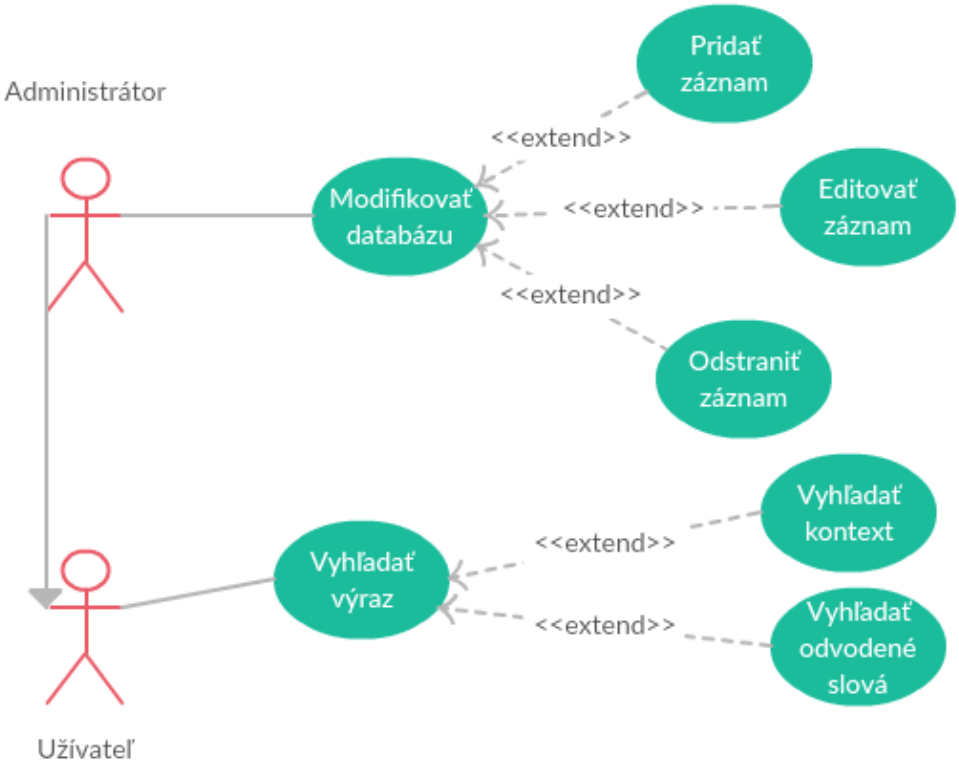
Pôvodne mala táto práca ponúknuť len rozšírenie pre existujúce riešenie, ktoré by prinieslo lepšie spracovanie najmä celotextového vyhľadávania, no postupom času prišlo k rozhodnutiu vyvinúť samostnanú aplikáciu, ktorá bude alternativou k existujúcim riešeniam.

3.1 Funkčný model

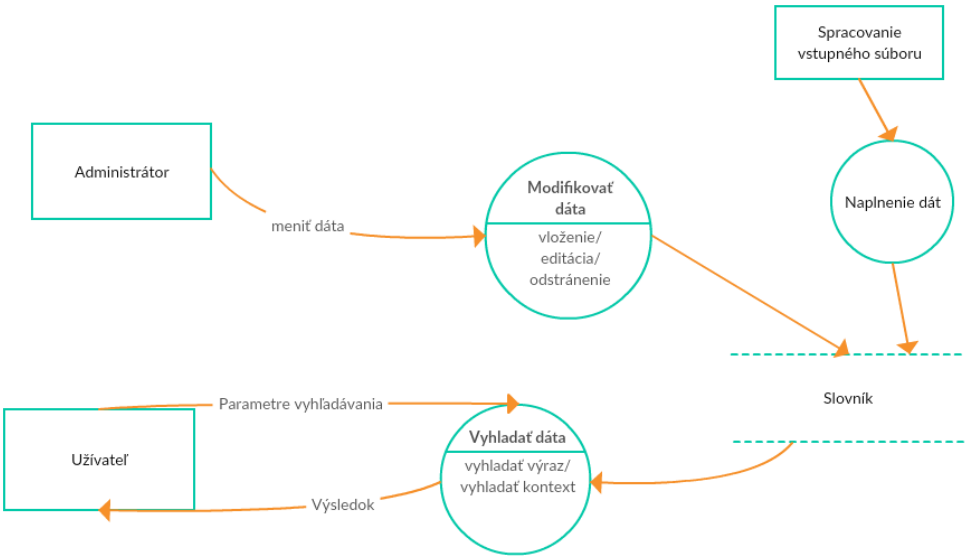
Pre správnu realizáciu systému je treba navrhnúť vhodný funkčný model, ktorý popisuje ako bude výsledný systém fungovať, ako bude pracovať s dátami a akým spôsobom poskytne interakciu užívateľovi. Z takéhoto návrhu je potom môžné pristúpiť k návrhu dátového modelu a samotnej implementácií, tak aby systém vhodne poskytoval všetky základné funkcie, ktoré sa od neho vyžadujú.

Pre slovníkovú aplikáciu, ktorá je cieľom tejto práce, je najdôležitejšia akcia vyhľadávania výrazov uživateľom. Je mu potreba správne sprostredkovať všetky informácie, ktoré sú obsiahnuté v slovníku a to v správnom poradí a rozsahu. Keďže sa jedná o aplikáciu pre systém Android a väčšina zariadení sú chytré telefóny, vzniká problém obmedzenej zobrazovacej plochy v porovnaní s klasickými monitormi. Pri prvom vyhľadaní je teda podstatné zobrazovať len základné informácie tak, aby užívateľ mal náhľad na čo najviac potencionálnych výsledkov. Pri dvojjazyčných slovníkoch by sa malo jednať o zoznam základných tvarov, ktoré odpovedajú vloženému vyhľadávanému reťazcu, ich základné vlastnosti a zoznam prekladov pre každý z nich. Medzi základne vlastnosti môže patriť napríklad slovný druh alebo výslovnosť. Až v ďalšom kroku by sa mala užívateľovi poskytnúť akcia zobrazenia rozširujúcich informácií podľa jeho potrieb. Príkladom takých informácií môžu byť súvisiace výrazy, použitie výrazov v kontexte a odvodené formy. Taktiež je dôležité, aby administrátor resp. vývojár takejto aplikácie mal možnosť modifikovať dáta, s ktorými užívateľ pracuje, tak aby v prípade opráv nepresností alebo ich rozšírenia bolo možné tieto dáta aktualizovať na koncovom zariadení užívateľa.

Na základe týchto potrieb bol zhotovený diagram prípadov použitia (*Use Case Diagram*) 3.1 a diagram toku dát (*Data Flow Diagram*) 3.2. Z týchto diagramov je vidieť, že s dátami bude užívateľ pracovať len vo forme čítania a administrátor vo forme zapisovania. Tok dát nie je komplikovaný, nevzniká potreba viacerých úložísk a vystačí len jedno, teda slovník. Administrátor má za úlohu spravovať dáta, aktualizovať ich a prípadne opravovať tak, aby boli korektné a obsahovali požadované dáta. Užívateľ je v tomto prípade konzumentom dát a na základe parametrov vyhľadávania sú mu predložené požadované informácie. Prípadov použitia nie je mnoho, avšak o to podstatnejšie je ich správne realizovať.



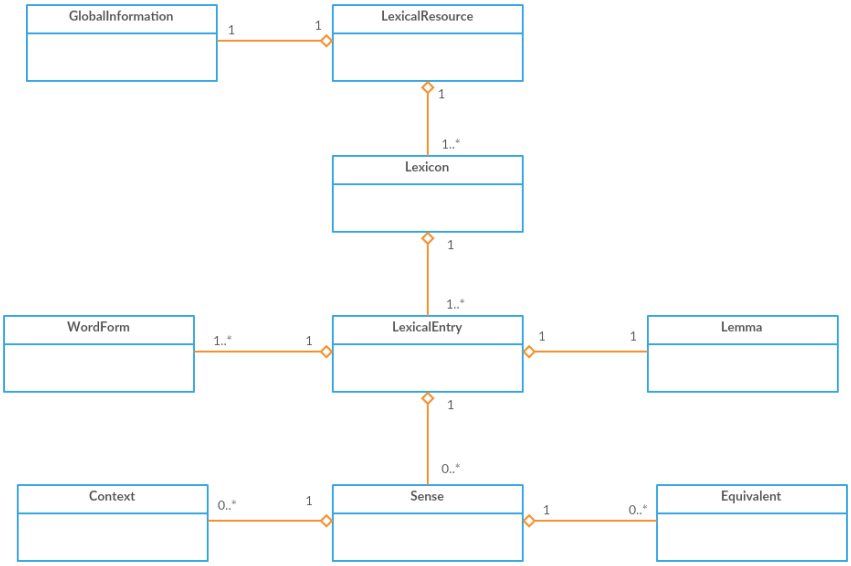
Obrázek 3.1: Diagram prípadu použitia



Obrázek 3.2: Diagram toku dát

3.2 Dáta a dátový model

Pri vyvíjaní mi bol k dispozicií na použitie od ústavu zapožičaný slovníkový súbor XML obsahujúci slovníkove dáta vo forme LMF. Jedná sa o dvojjazyčný slovník popísaný LMF



Obrázek 3.3: Diagram tried popisujúci použitý formát LMF

formátom zloženým z povinného základného balíka a rozširujúcich balíkov. Pre vytvorenie tohto slovníkového súboru bol použitý určite morfologický balík a balík pre strojové spracovanie, pretože súbor obsahuje entity lemma, ekvivalent a kontext, ktoré sa ukázali pre výsledne riešenie ako kľúčové. Predpokladaný formát LMF je popísaný diagramom tried na obrázku 3.3.

XML súbor mal teda podobu ako je na príklade 3.4. Lexikálne záznamy su očíslované atribútom id uzla LexicalEntry. Záznam obsahuje uzol feat, ktrorý obsahuje attribut att s hodnotou "partOfSpeech", ktorá hovorí o tom, že daný uzol obsahuje ešte atribut val,v ktorom je obsiahnutý slovný druh. Nie každý záznam obsahuje validný slovný druh, najmä viacslovné výrazy nemajú tento atribut vyplnený. Použitý slovník obsahuje približne 390000 lexikálnych záznamov, nie všetky sú však korektné. Niektoré vôbec neobsahujú zmysluplné dáta, ostatné môžu mať pár prekladov nesprávne.

Zvolené názvoslovie v XML súbore, a teda v diagrame tried, sa mierne líši od štandardu LMF, v nasledujúcej tabuľke je konverzia jednotlivých tried:

LMF štandard	**XML forma**
Database	LexicalResource
Lexicon	Lexicon
Lexicon information	GlobalInformation
Lexical Entry	LexicalEntry
Lemma	Lemma
Form	WordForm
Sense	Sense
Context	Context
Equivalent	Equivalent

Pre akciu samotného vyhľadávania sa najskôr počítalo, že vyhľadávanie bude prebiehať nad súborom so slovníkovými dátami priamo, teda pre nájdenie zadaného výrazu sa bude zakaždým spracovávať XML dokument vhodným spôsobom. Kvôli rozsahu a veľkosti

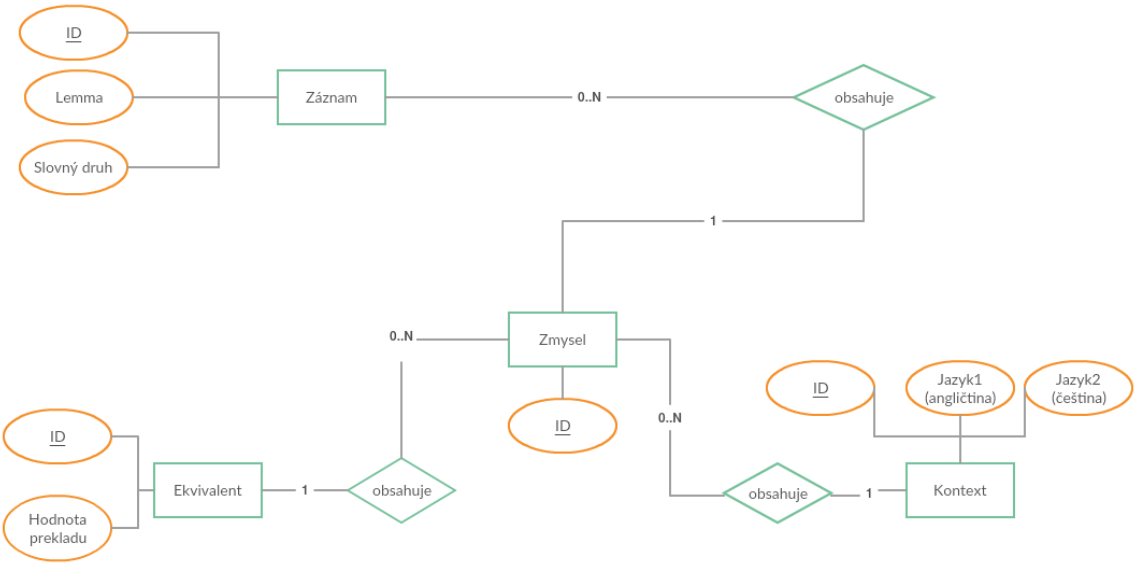
```xml
<?xml version="1.0" encoding="UTF-8"?>
<LexicalResource dtdVersion="16">
    <GlobalInformation>
        <feat att="languageCoding" val="ISO 639-3" />
    </GlobalInformation>
    <Lexicon>
        <LexicalEntry id="15">
            <feat att="partOfSpeech" val="noun" />
            <Lemma>
                <feat att="writtenForm" val="bike" />
            </Lemma>
            <WordForm>
                <feat att="phoneticForm" val="baIk" />
                <feat att="writtenForm" val="bike" />
            </WordForm>
            <Sense id="50">
                <feat att="senseNumber" val="1" />
                <Equivalent>
                    <feat att="language" val="ces" />
                    <feat att="writtenForm" val="kolo" />
                </Equivalent>
                <Equivalent>
                    <feat att="language" val="ces" />
                    <feat att="writtenForm" val="bicykl" />
                </Equivalent>
                <Context>
                    <TextRepresentation>
                        <feat att="languageIdentifier" val="eng" />
                        <feat att="text" val="ride a bike" />
                    </TextRepresentation>
                    <TextRepresentation>
                        <feat att="languageIdentifier" val="ces" />
                        <feat att="text" val="jet na kole" />
                    </TextRepresentation>
                </Context>
            </Sense>
```

Obrázek 3.4: Slovníkové dáta v XML súbore

slovníka bolo vybrané prúdove spracovanie. Takéto riešenie sa neskôr ukázalo ako veľmi nevhodné, pretože napriek rýchlosti prúdového spracovania boli časy vyhľadávania neúnosne a reálne nepoužiteľné.

Ďalším návrhom teda bolo jednorazové prúdové spracovanie a uloženie dát zo súboru XML do relačnej databáze, tak aby následne prebiehali dotazy na potrebné vyhľadávania nad danou databázou. Produktom tohto návrhu bol relačný model popísaný ER diagramom na obrázku 3.5 , pričom jednotlivé entity budú rozdelené do viacerých tabuliek. Rozdelenie je definované na základe prvého znaku základnej podoby slova (lemma). To znamená, že počet tabuľiek sa teoreticky a prakticky rovná počtu znakov anglickej abecedy, pretože základné podoby slova sú v našom súbore so slovníkovými dátami uvedné v angličtine. V prípadoch kedy základná forma nezačínala písmenom abecedy sa daná entita zaradila do tabuľky písmena “A”. Úlohou takéhoto rozdelenia bolo ešte viac zefektívniť vyhľadávanie a minimalizovať čas potrebný na získanie výsledku.

ER diagram sa skladá zo 4 entít: záznam, zmysel, ekvivalent a kontext. Entita záznam reprezentuje triedu LexicalEntry, obsahuje základnú podobu slova - lemma a atribut slovného druhu. Táto entita by sa dala ešte rozšíriť o atribut výslovnosti. Entita zmysel má



Obrázek 3.5: ER diagram popisujúci databázový model

v tomto modele len úlohu oddeľovať množiny ďalších entít na základe zmyslu, na ktoré sa viažu, nemá teda žiadne atributy okrem jednoznačného identifikátoru. Tejto entite odpovedá trieda Sense. Na zmysel je naviazaná entita ekvivalent, ktorá je nositeľom priameho prekladu lexikálneho záznamu. Obsahuje atribut, ktorý uchováva hodnotu prekladu. Na zmysel sa viaže ešte podstatná entita kontext, ktorá obsahuje príklad použitia daného záznamu v reálnych prípadoch. Keďže sa jedná o dvojjazyčný slovník, je uvedené použitie ako v jazyku anglickom, tak aj v jazyku českom.

Kapitola 4

Implementácia

Implementácia prebiehala prirodzene na platforme Android, teda v jazyku Java. Pre vývoj bolo použité vývojové prostredie Android Studio, ktoré je samozrejme najvhodnejším nastrojom pre prácu na projekte pre zvolený systém. Jedná sa o vysoko sofistikované prostredie s mnoho podpornými vlastnostami pre danú platformu. Keďže vývoj pre Android je veľmi normalizovaný, mnoho zo zvolených prostriedkov je určených práve pre Android, kde je ich výkon optimalizovaný.

Pôvodne mal byť slovník implementovaný ako dvojjazyčný slovník, teda anglicko-český a česko-anglický, napokon sa však podarilo vytvoriť len prvú časť a realizovať preklad z angličtiny do češtiny.

4.1 Spracovanie vstupného súboru a uchovanie dát

Prvým krokom vývoja bolo spracovanie vstupného XML súboru, ktorý obsahuje všetky dáta, ktore budú v aplikácií využité. Pre prácu s XML súborom bola vytvorená trieda `DictionaryParser`. Pôvodný návrh bol taký, že pri každom vyhľadaní výrazu užívateľom spracuje táto trieda vstupný súbor a vyhľadá v ňom požadovaný výsledok. Pre tento účel bol vybraný nástroj Aalto, ktorý je implementáciou spracovania XML typu StAX a bol by teda vhodný pre rozsiahly súbor. Počas pokusu integrovania tohto nástroju do môjho projektu nedošlo k úspechu, pretože Android priamo nepodporuje externé knižnice a nepodarilo sa mi prepojiť túto knižnicu na projekt pre Android. Preto prišlo k rozhodnutiu použit internú implementáciu pre spracovanie XML súboru, teda triedu `android.util.Xml`. Jedná sa o spracovanie typu SAX, ktorý by mal byť takisto vhodný pre rozsiahle XML súbory. V triede `DictionaryParser` boli vytvorené metódy, ktoré spracujú udalosti vyvolané prechodom cez XML. Ukázalo sa však, že obsah dát tohoto súboru je tak obrovský, že ani tento postup nebol schopný vykonať svoju činnosť v takom čase, ktorý by bol reálne použiteľný pre účely tejto práce. Pri vyhľadaní jedného výrazu sa jednalo o čas v ráde minút (okolo 6 minút).

Preto bolo navrhnuté alternatívne riešenie a to jednorázove spracovanie súboru XML s použitím vyššie spomenutých tried a naplnenie databáze dátami pre ďalšie použitie v aplikácii. Platforma Android natívne podporuje databázu typu SQLite. Pre vytvorenie takejto databáze bol použitý model popísany v sekcii 3.2. Programové napojenie na túto databázu má za úlohu trieda `DBStarter`, ktorá bola pôvodne potomkom Android triedy `android.database.sqlite.SQLiteOpenHelper` (dedenie bolo pri konci riešenia upravené z dôvodov popísaných v sekcii 4.4). Táto trieda priamo riadi vytvorenie databáze a jej

editáciu.

Nástrojom pre dotazovanie nad databázou je trieda `DBManager`. Stará sa o vytvorenie spojenia nad databázou a skladanie a realizáciu dotazov potrebných pre účely vyhľadávania z iných tried. Práca s databázou sa vykonáva pomocou objektov `Cursor`, ktoré uchovávajú informácie o jednom riadku danej tabuľky. Platoforma Android má však pamäťové obmedzenie na počet týchto objektov, ktoré môžu byť súčasne otvorené, preto sa dáva pozor na správnu prácu s nimi a ich uzatváranie.

Pre prácu s entitami vo vnútri kódu boli vytvorené odpovedajúce triedy, ktoré sú nosičmi dát v rámci rôznych operácií. Tieto triedy sú: `LexicalEntry`, `Sense` a `LexContext`. Entita ekvivalent je obsiahnutá v tomto prípade v triede `Sense` ako zoznam reťazcov, teda prekladov pre jeden zmysel. Tieto triedy potom boli použité pri jednorázovom prechode vstupných súborov pre uloženie dát pred ich vložením do databáze a taktiež boli neskôr použité pri vyhľadávaní výsledkov a ich zobrazovanie.

4.2 Stratégia vyhľadávania

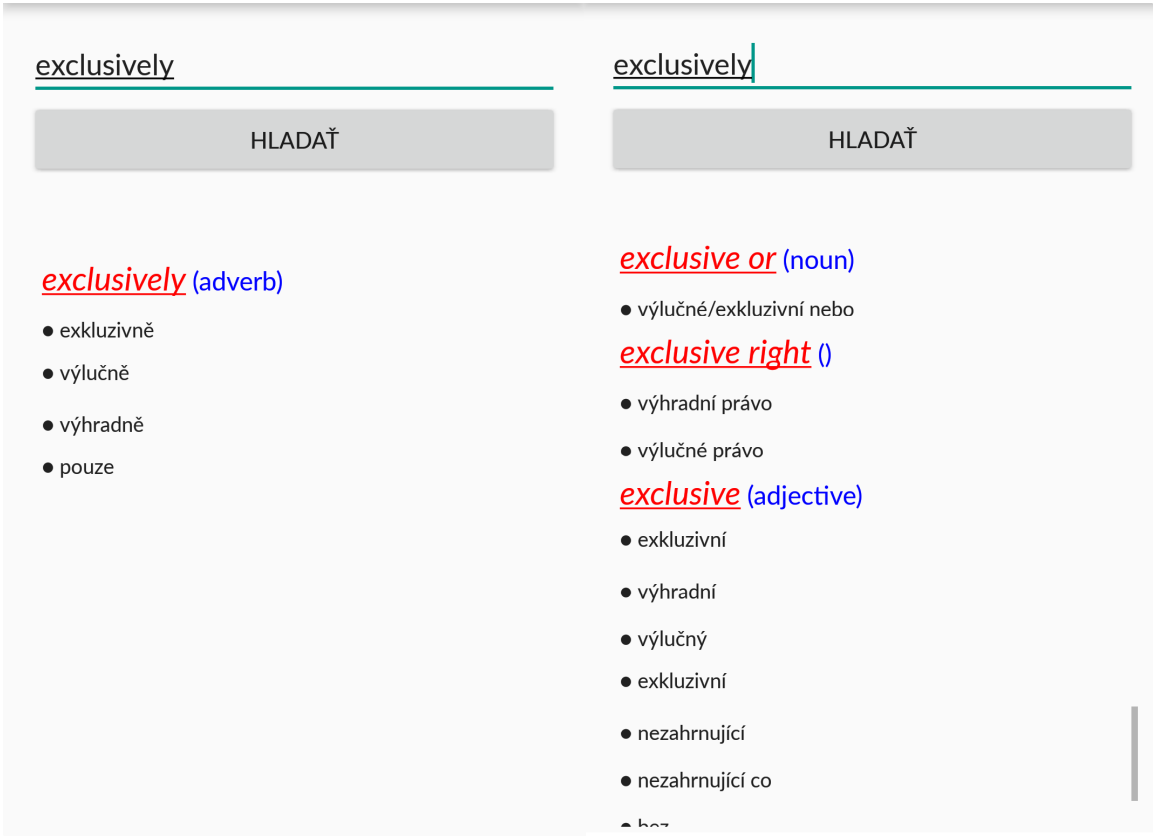
Jednou z najpodstatnejších častí implementácie tejto práce bolo navrhnutie a realizácia správneho vyhľadávania v dátach. Musel byť zvolený taký postup, ktorý umožnuje získavať výsledky, ktoré su vhodné pre účely mobilnej aplikácie. Systém vyhľadávania taktiež musí byť flexibilný a zachovať sa správne v rozličných situáciach. Veľmi dôležité v rámci tejto práce je celotextové vyhľadávanie, ktoré musí tento systém určitým spôsobom podporovať. Nie vždy je však možné dosiahnuť požadovaný výsledok. Aj v situáciach kedy užívateľ zadá zmysluplné vstupné parametre, nemusí byť výsledok obsiahnutý v dátach, s ktorými aplikácia pracuje, pretože sú prirodzene obmedzené.

Prvotné vyhľadávanie začína už v momente kedy užívateľ zadáva vstupný výraz na vyhľadávanie. Je vhodné, aby mu boli ponúkané výrazy na vyhľadávanie, ktore začínajú na ním zadaný reťazec. Od určitého zadaného písmena sa užívateľovi ponúkne zoznam základných podôb slov lexikálnych záznamov (atribútov lemma), ktoré sú asynchronným procesom vyhľadané v tabuľkách lexikálnych záznamov. Počet znakov, ktorý spúšťa toto vyhľadávanie bol nastavený v rámci tejto práce na dva. Pri jednom by sa v podstate mal ponúknuť celý obsah tabuľky, ktorý odpovedá zadanému písmenu, čo nie je veľmi vhodné. Pri dvoch písmenách už je ponúkaný zoznam relevantný a vyhľadávanie takého zoznamu trvá dobu, ktorá je nepatrná.

Pri vybratí finálneho výrazu na vyhľadanie je spustený taktiež asynchronný proces, ktorý už vyhľadáva výrazy dôkladnejšie. Logika tohto vyhľadávania je obsiahnutá v triede `Searcher`. Najskôr sa vyhodnotí či je zadaný reťazec jednoslovný alebo viacslovný.

V prípade, že sa jedná o jednoslovný výraz berie sa do úvahy uživateľské nastavenie aplikácie, ktoré udáva či sa majú vyhľadavať základne podoby slov. Ak je zvolená táto preferencia, tak sa vstupný reťazec uvedie do základnej formy. To sa realizuje pomocou triedy `EnglishStemmer`, ktorá implementuje nájdenie koreňa anglických slov. Implementácia vychádza zo zdroja [1]. Algoritmus je mierne upravený a do úvahy sa berú aj nepravidelné slovesá uvedené v [2]. Následne sa daný reťazec hľadá presnou zhodou v odpovedajúcej tabuľke lexikálnych záznamov podľa atributu lemma. Ak je výsledok prázdny tak sa hľadá podobne, ale nie je nutná presná zhoda, ale aby lemma začínala na vstupný reťazec. Výsledkom takéhoto vyhľadania je zoznam zakladných podôb slov a ich preklad. Ako príklad je vyhľadanie reťazcov “absolutely” a “exclusively” v prípade, že je zapnutá aj vypnutá užívateľská preferencia základnych foriem. Pri vstupe “absolutely” je koreň slova “absolute” a teda pri zvolení preferencie je výsledok hľadania forma “absolute” a k nej príslušné preklady.

Pri odobratí preferencie je výsledok rovnaký ako vložený výraz, teda forma “absolutely”. Pri vstupe “exclusively” je koreň slova vyhodnotení ako “exclusi” a pri zvolení preferencie je vyhľadaných viacero výrazov ako “exclusionism”,“exclusion” a “exclusive” a iné. Pri zmene preferencie je samozrejme výsledok len reťazec “exclusively” a jeho preklady 4.1.



Obrázek 4.1: Rozdiel vo vyhľadaní s rôznym uživatelským nastavením

Ak sa jedná o viacslovný výraz tak sa prihliada na uživateľské nastavenie, ktoré určuje, či sa má ponechať poradie prvého slova. Takéto nastavenie môže často spresniť výsledky, ktorých by inak bolo príliž veľa. Ako prvý krok vyhľadania viacslovného výrazu je vyhľadanie v lexikálnych záznamoch priamo v základných podobách slov. Vezme sa každé slovo vstupného reťazca a hľadá sa taká základna podoba, ktorá by obsahovala podreťazce, ktoré začínajú na všetky slová vstupného reťazca. Teda pri zadaní slov A a B musí základná podoba obsahovať podreťazec AX, ktorý začína na slovo A alebo podreťazec XA, ktorý končí na slovo A a zároveň obsahovať podreťazec BX, ktorý začína na slovo B alebo podreťazec XB, ktorý na dané slovo končí. Výsledkom je zoznam základných podôb, ktoré spĺňajú takéto podmienky a ich preklady. V prípade, že je výsledok prázdny pristúpi sa k rozšírenému vyhľadávaniu. Takéto vyhľadávanie spočíva v nájdení kontextových výrazov, ktoré by obsahovali slová zadané vo vstupnom reťazci. Výsledkom je potom zoznam kontextov v ktorých sa zadané slova používajú. Napríklad pri vyhľadaní slov “take from” je nájdená základná podoba “take away from” , no pri zadaní výrazu “in the heat” sú nájdené kontexty “in the heat of the moment” a “in the heat of the elections” spolu s prekladmi.

Po vyhľadaní, ktorého výsledkom je zoznam základných výrazov má užívateľ možnosť vyhladať kontextové výrazy, ktoré sa viažu na daný lexikálny záznam. Takéto vyhľadanie

je ale jednoduché a jedná sa o všetky kontexty, ktoré sú vo vzťahu s daným lexikálnym záznamom. Táto funkcionalita je pre užívateľa veľmi zaujímavá a pre výslednu aplikáciu kľúčová. Užívateľ po vyhľadaní výsledku má okrem zobrazenia priameho doslovného prekladu možnosť nahliadnuť na reálne príklady použitia, ktoré preňho môžu byť rozhodujúce, pretože častokrát je informácia o doslovnom preklade pre jazykové použitie nedostačujúca. Na to aby užívateľ pochopil celý význam slova a vedel ho správne zaradiť do jeho slovnej zásoby je potrebné vedieť ako môže dané slovo byť použité v danom jazyku. Vďaka tejto funkcionalite efektivita a platnosť informácie, ktorú užívateľ získa pri vyhľadaní výrazu v tomto riešení nesmierne stúpa.

4.3 Uživateľské rozhranie

Časť implementácie, ktorá je pre užívateľa njaviac zjavná je spracovanie grafického uživateľského rozhrania (GUI). Jedná sa o časť ktorá užívateľovi prezentuje a ponúka všetky informácie a funkcie, ktoré sú v aplikácii dostupné. Rozhranie by malo byť spravené tak, aby bolo prehľadné, jednoduché a aby s ním pracovalo rýchlo a efektívne.

Rozhranie tejto aplikácie tvorí základná obrazovka, ktorú užívateľ uvidí v okamihu kedy aplikáciu otvorí. Na tejto obrazovke sú umiestnené tri komponenty: textové poľe, tlačidlo a zoznam. Poľe slúží na zadávanie vstupných reťazcov na vyhľadávanie. Počas zadávania výrazu sa pod týmto textovým poľom zobrazí zoznam, ktorý užívateľovi ponúka možnosti na vyhľadanie, ktoré odpovedajú zadanému reťazcu. Po výbere jedného prvku z tohoto zoznamu sa ihneď spustí vyhľadávanie daného prvku.

Pod týmto poľom sa nachádza tlačidlo, ktoré spúšťa vyhľadávanie. Ak sa ponúkne užívateľovi možnosť pri zadávani textu, nie je toto tlačidlo potrebné, avšak pri dlhších reťazcoch, ktoré obvykle nie sú priamo ponúkane alebo pri opätovnom vyhľadaní je toto tlačidlo veľmi užitočné.

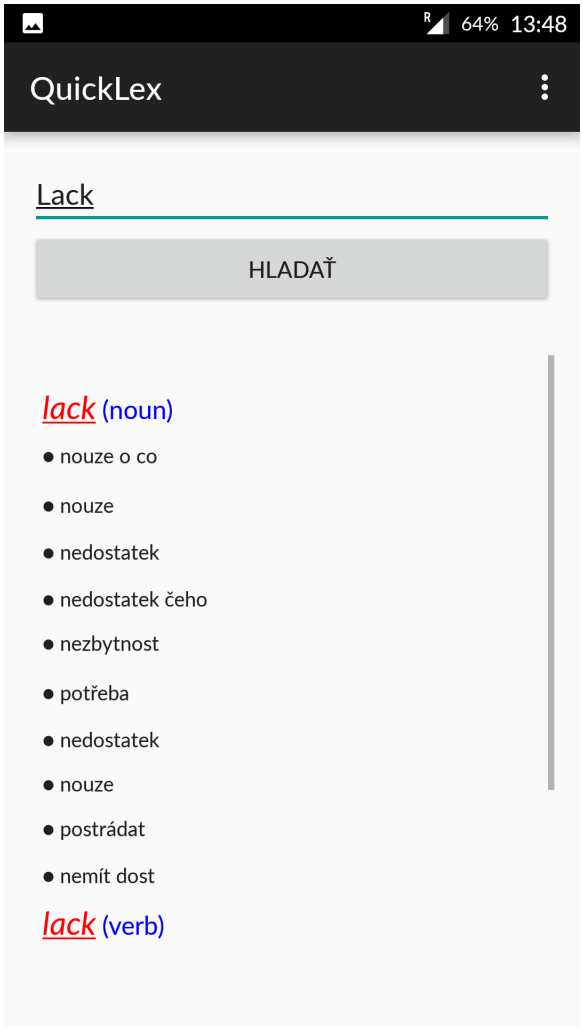
Najväčšiu plochu obrazovky zaberá zoznam, ktorý slúži na zobrazovanie výsledkov vyhľadávania. Zoznam obsahuje objekty triedy `android.text.Spanned` , ktorá umožnuje uchovávať formatovaný text niektorými značkami HTML. Je tak možné vizuálne oddeliť rôzne druhy zobrazovaných informácií.

Ak je zadávaný výraz nájdený v základných formách lexikálnych záznamov, výsledkom je zoznam daných záznamov spolu so slovným druhom a rôznymi prekladmi 4.2. Každý záznam je farebne oddelený, takisto aj slovný druh, tak aby boli informácie prehľadnejšie. Každý jeden preklad je uvedený na samostatnom riadku.

Základná forma je podčiarknutá, čo má u užívateľa vyvolať pocit, že sa jedná o odkaz. Kliknutím na odkaz sa spustí vyhľadávanie kontextového použitia daného lexikálneho záznamu a výsledok je zobrazený v rovnakej komponente zoznamu 4.3. Zoznam obsahuje príklad použitia v anglickom jazyku, ktorý je farebne zvýraznený a následne aj jeho český preklad. Takýto zoznam je zobrazený aj v prípade, že sa jednalo o viacslovné vyhľadávanie, ktorého výsledok bol nájdený len v kontextových použitiach. Samozrejmosťou je možný návrat z kontextového zoznamu naspäť na zoznam lexikálnych záznamov.

4.4 Nasadenie ku koncovému užívateľovi

Dôležitým krokom je takisto naplánovať akým spôsobom sa produkt dostane ku koncovému užívateľovi a ako bude potenciálne v budúcnosti udržovaný. Najväčším problémom v tomto smere bolo vyriešenie zahrnutia dát v aplikácií. Bolo vylúčené, aby bolo nutné k aplikácií



Obrázek 4.2: Výsledok zobrazujúci lexikálne záznamy

zahrňovať súbor XML s povôdnými dátami jednak preto, že daný súbor zaberá príliž veľký (okolo 500 MB) a jeho spracovanie zaberá neúnosne veľa času (pri naplnení databáze sa jedná o približne 3 hodiny). Bolo teda potrebné nájsť spôsobom, ktorým by sa dáta efektívne dostali ku koncovému užívateľovi.

Štandardom v rámci platformy Android je súbor typu .apk (*Android application package*). Jedná sa o archivačný súbor, ktorý slúži pre distribúciu a inštaláciu aplikácií pre Android. Takýto archivačný súbor potom môže byť poskytnutý užívateľovi na priamu inštaláciu. Pre zahrnutie dát do takéhoto súboru bola použitá trieda `SQLiteAssetHelper`, z ktorej nakoniec dedila vlastná trieda `DBStarter`. Potom bolo možné umiestniť databázu, ktorá bola prvotne jedenkrát naplnená, do zložky `assets` vo vnútri projektu (následne aj súboru .apk) a pri nainštalovaní užívateľom aplikácia pracuje práve s touto databázou. Táto databáza takisto môže byť v budúcnosti ľahko upravovaná aj na koncových zariadeniach užívateľov. Výsledný súbor .apk má veľkosť len 65 MB.

Pri úplnom dokončení tejto aplikácie by mohla byť umiestnená na službu Google Play, odkiaľ môže byť ľahko k dispozícii akejkoľvek osobe, ktorá vlastní zariadenie so systémom Android. Toto nebolo realizované, pretože aplikácia nie je vhodná na rozšírenie medzi veľký

Obrázek 4.3: Zobrazenie kontextového použitia výrazu

počet ľudí a na nahranie aplikácie na platformu Google Play je požadovaný účet, za ktorý je treba zaplatiť. Táto platforma by neskôr bola vhodná na zbieranie spätnej odozvy od väčšieho množstva užívateľov. Google Play taktiež umožnuje rýchly a efektívny spôsob, ktorým sa dá aktualizovať a pozmenovať aplikácia na koncových zariadeniach.

Kapitola 5

Testovanie

Veľmi dôležitým krokom v rámci životného cykla vývoja programového riešenia je fáza testovania. Testovanie môže prebiehať rôznymi spôsobami ako je automatizované testovanie, presné meranie výsledkov, no pre účely mobilnej aplikácie je najpodstatnejšie otestovať ako koncový užívateľ pracuje s aplikáciou, čo sa mu na nej páči a čo by vylepšil.

Preto som sa v rámci tejto práce rozhodol otestovať výslednú aplikáciu s reálnymi subjektami, ktoré by ju potencionálne využívali. Snažil som sa vybrať pestré spektrum subjektov tak, aby výsledky testov priniesli čo najväčšiu škálu názorov a problémov. Testovanie prebiehalo tak, že som poskytol aplikáciu osobe, ktorú som najskôr istý čas sledoval ako s ňou pracuje a neboli jej poskytnuté žiadne informácie okrem faktu, že sa jedná o slovník, ktorý je schopný prekladať z angličtiny do češtiny. Po určitej dobe, obvykle pár minútach som užívateľovi sprostredkoval pár tipov, ktoré by mu pomohli využiť funkcie, na ktoré sám neprišiel a pomohli lepšie pracovať s aplikáciou. Na koniec bol subjekt dotazovaný na vlastnosti aplikácie, ktoré sa mu páčili a ktoré sa mu nepáčili, teda veci, ktoré by si predstavoval inak a ktoré by vylepšil.

5.1 Subjekt 1

Subjekt 1 bol muž, ktorý študuje informačné technológie, no s platformou Android má minimálne skúsenosti. Angličtinu ovláda priemerne.

Prácu s aplikáciou začal vyhľadaním jednoduchého slova a podrobním skúmaním výsledku. Skritizoval jemné nepresnosti v dátach, ktoré mu zo začiatku kazili dojem z aplikácie. Len náhodou prišiel na funkcionalitu prechodu na kontextové príklady použitia, no táto časť aplikácie sa mu zdala veľmi užitočná. Až po vyzvaní skúsil užívateľ vyhľadať komplikovanejšie výrazy. Subjekt pri konci skúmal prvky grafického rozhrania a skritizoval pár z nich.

Užívateľ negatívne ohodnotil to, že bola možnosť prechodu na ďalší riadok pri zadávaní výrazu pre vyhľadávanie, túto akciu na virtuálnej klávesnici by nahradil tlačidlom pre akciu vyhľadania. Taktiež by zmenil polohu hlavného tlačidla pre vyhľadávanie. Veľmi pozitívne sa však vyjadril k ponúkaniu výrazov pre vyhľadávanie a možnosti vyhľadávania viacslovných spojení.

5.2 Subjekt 2

Subjekt 2 bol muž, ktorý sa venuje vývoju informačných systémov a má takisto skúsenosti s vývojom pre Android. Pred týmto testom však nikdy nepoužíval žiaden mobilný slovník.

Subjekt začal vyhľadávaním jednoduchších výrazov, no postupne prechádzal na zložitejšie viacslovné. Po istom čase sa mu začali zobrazovať kontexty namiesto lexikálnych záznamov, pretože dané výrazy boli nájdene len medzi príkladami v kontexte. Užívateľa to nerozrušilo, práve naopak zaujalo a následne začal skúšať čoraz komplikovanejšeie výrazy, medzi ktorými už boli aj také, ktoré slovník nebol schopný vyhľadať. Na koniec si vyhľadal ešte pár jednoduchých výrazov a náhodou prišiel na funkcionalitu prechodu zo záznamov na príklady použitia.

Užívateľ veľmi kladne ohodnotil najmä rýchlosť vyhľadávania a ponúkanie výrazov pri vkladaní reťazca pre vyhľadávanie. Taktiež ocenil schopnosť slovníka spracovať aj komplikovanejšie fráze. Vylepšil by však dizajn grafického a pridal by animáciu, ktorá by značila, že sa daný výraz vyhľadáva. Kritizoval ešte fakt, že je pomerne ľahké nechtiac ukončiť aplikáciu spätným tlačidlom, ktoré sa používa na prechod z príkladov použitia naspäť na lexikálne záznamy.

5.3 Subjekt 3

Subjekt číslo 3 bola žena, ktorá sa profesionálne zaoberá angličtinou a má skúsenosti s prekladom. V minulosti už využívala niektoré elektronické slovníky no príliž sa neorientuje v technických záležitostiach, ktoré sa týkaju napríklad mobilných aplikácií.

Pri samostatnej práci sa aplikáciou ihneď vyskúšala nájdenie zložitejšieho výrazu "raining cats and dogs". Aplikácia vyhľadala kontext "rain cats and dogs" s prekladom, na čo užívateľka reagovala veľmi pozitívne. Následne vyskúšala jednoduchšie slová, avšak sama neprišla na funkcionalitu odkazu základnej formy, ktorý vedie na zoznam kontextových použití.

Na koniec pozitívne ohodnotila to, že aplikácia dokáže nájsť aj komplikovanejšie výrazy a rôzne idiomy a ocenila aj funkcionalitu kontextových príkladov. V aplikácii jej chýbalo však napríklad uvedenie výslovnosti pri základnom vyhľadávaní, zobrazenie formy množného čísla a navrhla to, aby bolo možné nájsť príbuzné a odovodené slová.

5.4 Subjekt 4

Subjekt 4 bol muž, ktorý má skúsenosti s vývojom pre Android a uživateľskými rozhraniami. Pred týmto testovaním nepoužíval mobilné slovníky a v angličtine má priemerné znalosti.

Prácu s aplikáciou začal vyhľadávaním jednoduchých slov, skúmal preklady a varianty slovných druhov. Postupne sa dostal aj do nastavení uživateľských preferencií, kde skúsil niektoré zapnúť a následne vyhladať výraz a sledoval, aké maju nastavanie dopady na vyhľadané záznamy. Po chvíli mu bolo poradené nech vyskúša akciu prechodu na príklady kontextového použitia, na ktorú sam neprišiel a taktiež nech vyhľadá zložitejšie viacslovné výrazy.

Užívateľovi sa páčila funkcionalita prechodu na kontextové výrazy a pozitívne ohodnotil rýchlosť vyhľadávania. Vylepšil by však grafické spracovanie a zobrazovanie výsledkov, ktoré sa mu zdalo príliž jednoduché. Taktiež by doplnil animáciu pri vyhľadávaní výsledkov.

5.5 Subjekt 5

Subjekt 5 bola žena, ktorá sa venuje vývoju softvérových riešení. Má skúsenosti s používaním rôznych mobilných aplikacií.

Uživateľka sa v úvodných momentoch zamerala na testovanie ovládania aplikácie, otvorila nastavenia pomocou tlačidla na obrazovke a skúsila či funguje aj "fyzické" tlačidlo pod obrazovkou, ktoré typicky slúži na zobrazenie nastavenia v Android aplikáciach. Následne začala vyhladávať výrazy a skúšala vyhľadávanie spustiť kliknutím na zobrazovanú ponuku slov aj dedikovaným tlačidlom na obrazovke. Sama prišla na prechod na kontextové príklady záznamov a neskôr vyskúšala aj zložitejšie výrazy.

Na záver skritizovala absenciu tlačidla na vyhľadávanie na klávesnici, ktorá sa zobrazí pri vkládaní textu pre vyhľadávanie a taktiež umiestnenie tlačidla "Hladať" na obrazovke. Pochválila však rýchlosť vyhľadávania a prehľadné zobrazenie výsledkov.

5.6 Subjekt 6

Subjekt 6 bol muž, ktorý profesionálne analyzuje a testuje informačné riešenia. Aktívne využíva rôzne aplikácie na platforme Android, no s anglickým jazykom má len základne skúsenosti.

Užívateľ začal vyhladávať pomerne jednoduché výrazy a experimentoval s ovládaním aplikácie. Samostatne prišiel aj na funkcionalitu odkazu na kontextové použitie a takisto preskúmal aj uživateľské nastavenia. Sám začal upozorňovať na veci, najmä v rámci uživateľského rozhrania, ktoré sa mu nepáčili. Následne po pokyne vyskúšal aj zložitejší viacslovný výraz.

Negatívne sa vyjadril k nekonzistentosti jazyka, ktorý sa vyskytoval v ovládaní aplikácie, kde sa mieša jazyk systému Android, v tomto prípade angličtina s vlastným ovládaním v slovenčine a prekladami výrazov v češtine. Inak sa mu uživateľské rozhranie páčilo, zdalo sa mu jednoduché a prehľadne, zmenil by len umiestnenie tlačidla "Hladať". Takisto sa mu páčila kontextová funkcionalita a schopnosť vyhľadávať viacslovné výrazy.

5.7 Výsledky testovania

Výsledky, ktoré vychádzajú z testovania na reálnych užívateľoch sú pre túto prácu a jej budúcnosť nesmierne dôležité. Na základe nich a budúcich ohlasov je možne aplikáciu doupravovať tak, aby sa odstránili nedostatky, na ktoré uživatelia najviac upozorňujú. Taktiež je možné z testovania spoznať, ktoré funkcie a časti aplikácie majú úspech a sú spracované vhodným spôsobom.

Z prvotného testovania je zrejmé, že subjekty testovania reagovali na výslednú aplikáciu skôr pozitívne než negatívne a veci, ktoré by vylepšili sú väčšinou len kozmetického charakteru.

Negatívne boli hodnotené hlavne časti týkajúce sa grafického rozhrania a dizajnu aplikácie. Tieto časti môžu byť v budúcnosti pomerne ľahko upravené tak, aby výsledná podoba aplikácie vyzerala lepšie a aby problémy, ktoré uživatelia s aplikáciou mali boli odstránené.

Užívatelia hodnotili kladne najmä funkcionalitu prechodu z lexikálnych záznamov na zoznam príkladov kontextového použitia, schopnosť aplikácie vyhladávať viacslovné a komplikovanejšie výrazy a taktiež rýchlosť vyhľadávania. Tieto časti boli pre túto prácu kľúčove a fakt, že mali pri testovaní pozitivný ohlas svedčí o úspechu výsledného riešenia.

Kapitola 6

Záver

Táto práca sa zameriava na zhodnotenie súčasného stavu ukladania a prehľadávania slovníkových databázi na platforme Android. Obsahuje časť, ktorá analyzuje štandard pre ukladanie slovníkových dát, teda Lexical Markup Framework. Práca analyzuje spracovanie XML súborov, v ktorých typicky bývajú uložené takéto dáta a takisto rozoberá a hodnotí existujúce riešenia.

V rámci práce boli iteratívne vytvorené viaceré navrhy od začiatočnej fáze, kde sa najskôr predpokladalo len spracovanie XML súborov bez použitia databáze až po finálny návrh, ktorý predkladá kompletný návrh funkčného a databázového modelu vlastného riešenia.

Výsledkom tejto práce je implementácia takéhoto vlastného riešenia vo forme aplikácie pre platformu Android. Aplikácia bola vyvinutá v jazyku Java s pomocou vývojového prostredia Android Studio. Táto aplikácia v sebe obsahuje databázu, ktorá bola jednorázovo naplnená zo vstupného súboru XML, ktorý obsahoval slovníkové dáta vo forme LMF. Aplikácia potom sprostredkováva vyhľadávanie výrazov v naplnenej databázi, pričom v rámci tejto práce bolo podstatné správne spracovať najmä zložitejšie viacslovné spojenia. Výsledky vyhľadávania sú sprostredkované uživateľom v rámci jednoduchého a prehľadného grafického rozhrania.

Práca hodnotí úspech výsledného riešenia pomocou testovania na reálnych užívateľoch. Výsledky týchto testov hovoria o dobrom spracovaní kľúčových aspektov tejto práce a navrhujú úpravy do budúcnosti. Riešenie zvláda prácu s komplikovnejšími viacslovnými výrazmi a obsahuje funkcionalitu zobrazenia kontextových príkladov použitia vyhľadaných výrazov, teda časti, ktoré si práca dala za úlohu vyriešiť. V rámci ďalšieho vývoja by aplikácia mohla byť upravená tak, aby poskytovala ešte viac rozšírene možnosti vyhľadávania a poskytovania informácií ako napríklad zobrazovanie výslovnosti, vyhľadávanie príbuzných a odvodených slov alebo zobrazovanie rôznych foriem slov. Taktiež by bolo vhodné dokončiť aplikáciu funkcionalitou zpätného jazykového prekladu z češtiny do angličtiny, ktorá bola pôvodne plánovaná.

Literatura

[1] The English (Porter2) stemming algorithm. [Online; navštívené 6.2.2016].
URL http://snowball.tartarus.org/algorithms/english/stemmer.html

[2] Irregular Verb Dictionary. [Online; navštívené 15.3.2016].
URL http://www.englishpage.com/irregularverbs/irregularverbs.html

[3] Francopoulo, G.; George, M.; Calzolari, N.; aj.: Lexical Markup Framewrok (LMF). 2006 [cit. 2016], [Online; navštívené 19.4.2016].
URL http://www.tagmatica.fr/lmf/LMFPaperForLREC2006FinalSubmission31March06.pdf

[4] International Organization for Standardization: ISO 24613:2008 , Language resource management - Lexical markup framework (LMF). [Online; navštívené 18.4.2016].
URL http://www.iso.org/iso/catalogue_detail.htm?csnumber=37327

Přílohy

Seznam příloh

Příloha A

Obsah CD

Priložené CD obsahuje zdrojové súbory a dokumenty tejto práce v následovnej adresárovej štruktúre:

doc/	elektronická verzia tohto dokumentu
doc/src/	zdrojové súbory pre zostavenie tohto dokumentu v nástroji LaTeX
src/	zdrojové súbory pre projekt v prostredí Android Studio
app/	Inštalačný súbor .apk pre platformu Android