VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

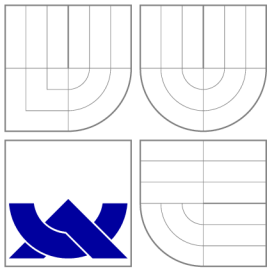
INOVACE UŽIVATELSKÉHO ROZHRANÍ DATABÁZE WORDNET

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

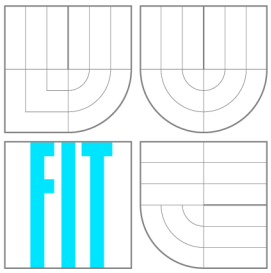
AUTOR PRÁCE
AUTHOR

ADAM KOPŘIVA

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

INOVACE UŽIVATELSKÉHO ROZHRANÍ DATABÁZE WORDNET

WORDNET DATABASE USER INTERFACE INNOVATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — ADAM KOPŘIVA
AUTHOR

VEDOUCÍ PRÁCE — Ing. PETR CHMELAŘ
SUPERVISOR

BRNO 2008

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav informačních systémů Akademický rok 2007/2008

Zadání bakalářské práce

Řešitel: **Kopřiva Adam**
Obor: Informační technologie
Téma: **Inovace uživatelského rozhraní databáze WordNet**
Kategorie: Uživatelská rozhraní

Pokyny:

1. Seznamte se s lexikální databází angličtiny WordNet a jejími rozhraními v Javě jako je JWord.
2. Identifikujte a pokuste se vyřešit problémy uživatelského rozhraní těchto aplikací.
3. Vytvořte grafický nástroj pro dotazování, zobrazení výsledků a zadávání nových informací do databáze WordNet.
4. Zhodnoťte vlastnosti a případné vylepšení nástroje.

Literatura:

- The George Washington University. *JWord 3.0* [online]. [cit. 2007-10-02]. Dostupný z WWW: <http://www.seas.gwu.edu/~simhaweb/software/jword/>
- Miller, J. et al. *Introduction to WordNet: An Online Lexical Database* [cit. 2007-10-02]. 1993. Dostupný z WWW: <wordnet.princeton.edu/5papers.pdf>
- Herout, P. *Učebnice jazyka Java*. České Budějovice: Kopp, 2007. 381 s. ISBN 978-80-7232-323-4.
- Herout, P. *Java - grafické uživatelské prostředí a čeština*. České Budějovice: Kopp, 2007. 320 s. ISBN: 80-7232-237-0.

Při obhajobě semestrální části projektu je požadováno:

- 1. a 2. bod zadání.

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

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

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

Vedoucí: **Chmelař Petr, Ing.**, UIFS FIT VUT
Datum zadání: 1. listopadu 2007
Datum odevzdání: 14. května 2008

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

L.S.

doc. Ing. Jaroslav Zendulka, CSc.
vedoucí ústavu

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. Pan

Jméno a příjmení: **Adam Kopřiva**
Id studenta: 78925
Bytem: Měnín 391, 664 57 Měnín
Narozen: 22. 05. 1986, Brno
(dále jen "autor")

a

2. Vysoké učení technické v Brně

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

.....

(dále jen "nabyvatel")

Článek 1
Specifikace školního díla

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

Název VŠKP: Inovace uživatelského rozhraní databáze WordNet
Vedoucí/školitel VŠKP: Chmelař Petr, Ing.
Ústav: Ústav informačních systémů
Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v:

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

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

Článek 2
Udělení licenčního oprávnění

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

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

Článek 3
Závěrečná ustanovení

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

V Brně dne: .....

..... Adam Kozák .....

Nabyvatel Autor

Abstrakt

Tato práce se věnuje inovaci grafického uživatelského rozhraní programu JWord 3.0. Jedná se o program, který pracuje s elektronickou lexikální databází angličtiny WordNet. Slova v této databázi nejsou narozdíl od běžného slovníku řazena abecedně, ale jsou seskupena do synonymických řad. Tyto řady pak spojují různé sémantické vztahy. Při návrhu implementace byl kladen důraz zejména na rychlost vyhledání informací k zadanému slovu. To mělo za následek přechod od současného vyhledávání v souborech k získávání informací pomocí dotazů v databázi. Výsledný program se skládá ze dvou částí - programového a grafického uživatelského rozhraní. Programové rozhraní pracuje nad databází MySQL WordNet 3.0. Uživatelské rozhraní pak pomocí programového zobrazuje hledané významy zadaného slova a vztahy k dalším slovům (synonyma, antonyma, hyponyma, hypernyma).

Klíčová slova

WordNet, JWord, MySQL, grafické uživatelské rozhraní, GUI, Java, sémantický web

Abstract

This thesis considers innovations of graphical user interface for JWord 3.0. It's a program, which works with electronic lexical database of English - WordNet. The words in this database aren't sorted in the alphabetical order like in usual dictionaries, but they are grouped to synonymic groups. These groups are then connected with different semantic relations. During the implementation design I have focused mainly on the speed of searching for informations about the entered word. This lead to the change of searching engine from searching in files to querying database. The resulting program is composed of two parts - program and graphical user interface. Program interface works with MySQL WordNet 3.0 database. User interface then shows meanings of the entered word and relations to other words (synonyms, antonyms, hyponyms, hypernyms).

Keywords

WordNet, JWord, MySQL, graphical user interface, GUI, Java, semantic web

Citace

Adam Kopřiva: Inovace uživatelského rozhraní databáze WordNet, bakalářská práce, Brno, FIT VUT v Brně, 2008

Inovace uživatelského rozhraní databáze WordNet

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Petra Chmelaře

.....
Adam Kopřiva
11. května 2008

Poděkování

Chtěl bych poděkovat panu Ing. Petru Chmelařovi za jeho pomoc a podporu při řešení této bakalářské práce.

Obsah

Kapitola 1

Úvod

1.1 Sémantický web

V posledních letech můžeme vidět snahu o vytvoření sémantického webu [10]. Sémantika je v jazykovědě nauka o významech slov a vyšších jazykových celků. Sémantickým webem můžeme nazvat takovou síť, kde dokumenty obsahují skryté značky, které poskytují informace o významu obsažených dat. Idea sémantického (významového) webu si zakládá na doplnění sítě webových stránek sítí výroků. Tyto výroky pak lze narozdíl od současných webových stránek automatizovaně zpracovávat. Sémantika tedy umožňuje počítačové zpracování psaného textu, snadné vyhledávání a kategorizaci.

Nejedná se o další počítačový jazyk, informační systémy jsou vytvářeny jako automatizované systémy, do nichž lidé informace zadávají prostřednictvím promyšlených rozhraní. Pro přidání sémantické hodnoty je tedy jen třeba rozšířit tato rozhraní (obvykle formuláře administračních aplikací) o několik jednoduchých prvků.

Základním prvkem semantického webu je ontologie. Ontologie umožňuje tvorbu metadat, neboli informací o informacích. Tato metadata dodávají dokumentu či jeho částem tolik potřebné významové informace. Účelem ontologií je podpora porozumění mezi lidmi, podpora komunikace mezi počítačovými systémy a podpora návrhu znalostně orientovaných systémů. Jedním z možných zdrojů pro budování ontoligí (formální specifikace pojmů a vztahů mezi nimi) může být i *WordNet*.

1.2 WordNet

WordNet je výkladový slovník moderní angličtiny, původem z Princetonské Univerzity. Jak již napovídá sám název, jedná se o síť slov propojených sémantickými vztahy. Vývoj této lexikální databáze začal již v roce 1985 a nyní se nachází ve verzi 3.0 a obsahuje již přes 155000 unikátních řetězců.

Program, který dokáže vyhledávát informace v databázi WordNet a přes grafické rozhraní zobrazit výsledek uživateli vznikl na univerzitě George Washingtona. Jedná se o program JWord a nachází se ve verzi 3.0. Autoři však vyzývají k inovaci jeho grafického uživatelského rozhraní.

Ze zásad anglického lexikálního slovníku WordNet vycházejí i příbuzné projekty pro další jazyky. Jedním z nich je *EuroWordNet*, který obsahuje skupinu sémantických slovníků pro několik evropských jazyků. Dalším projektem je *Global WordNet Association*, který navazuje na EuroWordNet a podporuje vytváření dalších slovníků pro další světové jazyky.

Vznikají i nové projekty pro vytváření slovníků s dalšími jazyky např. *BalkaNet*.

1.3 Struktura práce

Úvod práce věnuji podrobnému seznámení s elektronickou lexikální databází angličtiny WordNet. Od začátku jeho vývoje až po současnou verzi databáze. Rozebírám způsob uložení slov a vztahy mezi nimy. V další kapitole analyzuji program JWord 3.0, jehož novější verzí se tato práce zabývá. Popisuji také jeho výhody a nedostatky. V následující kapitole navrhuji způsob uložení dat slovníku a volbu vhodného programovacího jazyka. V kapitole, věnující se implementaci, popisuji obě části programu - programové a grafické uživatelské rozhraní. Jsou vysvětleny všechny použitelné funkce pro další vývoj programového rozhraní. Na závěr práce uvádím zhodnocení dosažených výsledků a nabízím další možnosti pro rozšíření programu.

Kapitola 2

Seznámení s databází WordNet

2.1 Lexikální databáze angličtiny

WordNet [8] je elektronická lexikální databáze angličtiny, kterou vytvořili vědci na Princetonské Univerzitě podle současných teorií o lidské lexikální paměti. Anglická podstatná jména, slovesa a přídavná jména jsou seskupena do *synonymických* řad. Příslovce jsou organizována na základě jiných vztahů a ostatní slovní druhy nejsou do slovníku zahrnuty vůbec. Nejsou totiž důležité pro strojové zpracování textu a také mezi nimi neexistují vhodné sémantické vztahy.

Synonymické řady nazýváme synsety. Jedná se o síť slov, spojených sémantickými vztahy. Různé vztahy vyskytující se mezi nimi pak spojují tyto vzniké synsety.

2.1.1 Vztahy mezi slovy

Každá řada synsetů zastupuje jeden základní prvek lexikálního konceptu databáze. Vztahy mezi synsety mohou být buď *hyperonymické*, pokud spojují synsety s obecnějším významem, nebo *hyponymické* pokud spojují synonymické řady s konkrétnějším významem. WordNet pracuje i s dalšími vztahy mezi slovy, které budou rozebrány později (*synonyma, antonyma, meronyma*).

2.1.2 Řazení slov

Lexikální informace nejčastěji, a především standardně, řadíme podle abecedy. To však rozmístí slova se stejným nebo podobným významem po celém slovníku. Naneštěstí, neexistuje jiná alternativa, jak jednodušeji slova seřadit, aby je čtenáři našli. Abecedně řazená slova mají tu nevýhodu, že hledání v takto seřazeném seznamu se stává obtížným a časově náročným. Práce s tímto standardním slovníkem mnoho lidí rozptyluje, přerušuje totiž jejich činnost a stejně tak jejich myšlenkové pochody.

V době masivního využívání počítačů již exisuje řešení tohoto problému. Jeden z důvodů rozvoje elektronických slovníků (lexikálních databází) je vyhledávání informací pomocí počítačů. Počítače dokáží vyhledávat v abecedně řazeném seznamu neporovnatelně rychleji než lidé. Výsledek hledání často program zobrazí tak rychle, jak uživatel slovo vybere na obrazovce nebo zadá pomocí klávesnice. Od doby, co se využívají počítačové programy jako výkladové slovníky a uživatelé pomocí nich vyhledávají informace, lze poměrně jednoduše přeměnit tyto slovníky na vhodný druh lexikální databáze. Přeměnu tradičních slovníků na elektronické považujeme za přirozený vývoj.

WordNet 1.7.1					
	podstatná jména	**slovesa**	**přídavná jména**	**příslovce**	**celkem**
unikátní řetězce	109195	11088	21460	4607	146350
synsety	75804	13214	18576	3629	111223
WordNet 3.0					
unikátní řetězce	117798	11529	21479	4481	155287
synsety	82115	13767	18156	3621	117659

Tabulka 2.1: Tabulka s počtem slov a synsetů u verze WordNet 1.7.1 a 3.0 [7]

Využívání elektronických slovníků nám ukazuje, jak je neefektivní používat tyto výkonné stroje pouze k otáčení stránek. Důležité je pokusit se využít jejich výkon. WordNet přichází s efektivní kombinací tradičního výkladového slovníku a moderní vysokorychlostní komunikace.

2.2 Historie

2.2.1 Psycholexikologie

V roce 1985 se skupina psychologů a jazykovědců na Princetonské Univerzitě ujala vývoje lexikální databáze. První myšlenkou bylo poskytnout pomoc při vyhledávání ve slovníku spíše významově, než jen abecedně. Když práce pokročila, bylo potřeba stanovit ctižádostivější cíle založené na vlastních principech a řešeních. WordNet vznikl jako výsledek jejich snažení a zakládá se na psycholingvistických principech.

Počet slov u jednotlivých slovních druhů v lexikální databázi Wordnet uvádí tabulka 2.1. Tabulka obsahuje pro porovnání dvě verze slovníku, starší WordNet 1.7.1 a současnou WordNet 3.0.

2.2.2 Uspořádání slovníku

WordNet rozlišuje mezi semantickými a lexikálními vztahy. Největší rozdíl mezi WordNetem a standardními slovníky najdeme v tom, že WordNet dělí slovník na pět částí: podstatná jména, slovesa, přídavná jména, příslovce a funkční slova. Anglická funkční slova ukládá odděleně jako syntaktickou část jazyka. Z výsledků prvních studií slovních spojení vyšlo najevo, že realizace této syntaktické kategorie se liší osobním (subjektivním) uspořádáním.

2.3 Lexikální matice

Lexikální sémantika definuje, že slovo je tradičním spojením mezi lexikálním pojetím a mluveným projevem, který hraje syntaktickou roli. Tato definice slova vyvolává nejméně tři otázky pro vyhlédávání slov.

1. Které části mluveného projevu vložit do lexikálních vztahů?

2. Jaká je přirozená a uspořádaná forma lexikálního pojetí slova?

Významy slova	**Druhy slova**				
	F_1	F_2	F_3	$\ldots$	F_n
M_1	$E_{1,1}$	$E_{1,2}$			
M_2		$E_{2,2}$			
M_3			$E_{3,3}$		
$\vdots$				$\ddots$	
M_m					$E_{m,n}$

Tabulka 2.2: Názorný příklad uspořádání lexikální matice

3. Které syntaktické vztahy dokáží vyjádřit různé hry se slovy?

Přestože nemůžeme zanedbat ani jednu z těchto otázek, v této části se výrazně zaměříme na druhou z nich. Budeme se tedy zabývat uspořádáním slov anglického slovníku.

2.3.1 Příklad uspořádání

Tabulka 2.2 nabízí jednoduchý příklad, jak si udělat skutečný obrázek o lexikální matici. V záhlaví sloupců naleznete druh slova. Významy slova uvádí tabulka v řádcích. Záznam v buňce tabulky naznačuje, že druh uvedený ve sloupci může být použit (ve vhodné souvislosti) k vyjádření významu na řádku. Například $E_{1,1}$ uvádí, že druh slova F_1 může být využit k vyjádření významu M_1. Pokud tabulka obsahuje dva nebo více záznamů ve stejném sloupci, má slovo více významů. Pokud jsou dva nebo více záznamů na stejném řádku jedná se o synonyma (v závislosti na významu).

2.3.2 Vztah mezi druhy a významy slov

Zobrazení mezi druhy a významy slov je "many:many" - některé druhy slov mají několik významů a některé významy popisuje několik slov. Dva složité problémy lexikografie, slova s více významy a synonymičnost, uvádíme jako doplňkové hledisko toho zobrazení. Synonymičnost a vícevýznamovost jsou problémy, které nastávají i při běžné komunikaci:

- posluchač nebo čtenář, který přečte (uslyší) slovo, si musí správně vybrat mezi jeho více významy
- řečník nebo spisovatel, který popisuje význam slova, se musí rozhodnout, jaké slovo pro daný význam použije

2.4 Vztahy mezi slovy

V lexikálním slovníku WordNet existují různé vztahy mezi druhy slov a mezi významy slov. Patří mezi ně:

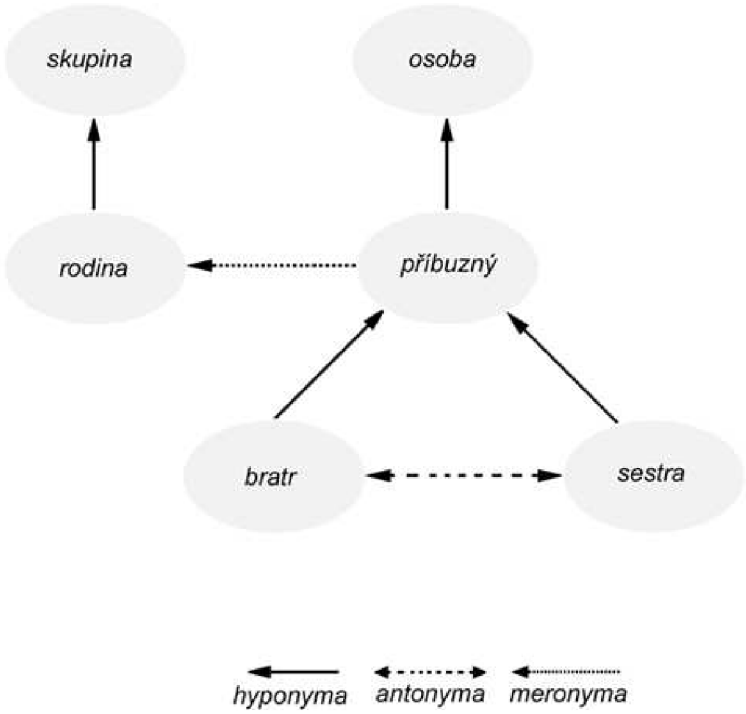
a) Synonyma

b) Antonyma

c) Hyponyma a hypernyma

d) Meronyma a holonyma

Ukázku těchto vztahů mezi slovy můžete vidět na obrázku 2.1.



Obrázek 2.1: Ukázka možných vztahů mezi slovy

2.4.1 Synonyma

Nejdůležitějšími vztahy pro WordNet jsou podobnosti ve významu slov. Podle jedné z definic (obvykle připisována Leibnizovi [11]) jsou dva výrazy synonymní, jestliže nahrazením jednoho z nich za druhý není nikdy změněn skutečný význam věty, v které byla záměna provedena. Podle této definice, pokud vůbec existují, jsou skutečná synonyma vzácná. Jednodušší verze této definice dělá synonymičnost relativnější k souvislostem: *dva výrazy jsou synonymní v lingvistickém kontextu C, jestliže náhrada jednoho výrazu za druhý v C nezmění skutečný význam.*

2.4.2 Antonyma

Dalším druhem vztahů jsou slova s opačným významem (antonyma). Tyto slova překvapivě těžko definujeme. Antonymum od slova x je často slovo záporné k x, ale nemusí tomu tak býti vždy. Například *bohatý* a *chudý* jsou antonyma. Ale pokud řekneme, že někdo není bohatý neznamená to, že musí být chudý. Mnoho lidí se necítí býti bohatými, ale ani chudými.

Slova s opačným významem mají lexikální vztah mezi druhy slov, nemají sémantický vztah mezi významy slov. Antonyma poskytují ve WordNetu základní uspořádání pro přídavná jména a příslovce.

2.4.3 Hyponyma a hypernyma

Na rozdíl od synonym a antonym, která mají lexikální vztahy mezi druhy slov, hyponyma a hypernyma jsou založena na sémantických vztazích mezi významy slov. Například *javor* je hyponymum pro *strom*, a *strom* je hyponymum pro *rostlinu*. Naopak *rostlina* je hypernymum pro *strom*, a *strom* je hypernymum pro *javor*.

Hyponimické a hypernymické vztahy mohou být také označovány jako *podřazenost a nadřazenost* nebo *podmnožina a nadmnožina*. Na základě těchto vztahů jsou ve WordNetu uspořádána podstatná jména.

2.4.4 Meronyma a holonyma

Jedná se o sémantický vztah vyjadřující vztahy mezi částí a celkem. Meronymum vyjadřuje část celku. Naopak holonymum je spojením celku s jeho částmi. Například *modrá* je meronymem k *barva*, *obličej* je holonymem pro *nos*. Ve WordNetu meronyma popisujeme vloženými skupinami slov nebo ukazateli (označeny *arcs*) z jednoho synsetu na druhý.

2.5 Další projekty

Ze zásad anglického lexikálního slovníku WordNet vycházejí i příbuzné projekty pro další jazyky. Největší úspěch slaví zejména dva z nich:

a) EuroWordNet [3]

b) Global WordNet Association [1]

2.5.1 EuroWordNet

Jedná se o skupinu sémantických slovníků pro několik evropských jazyků. Do této skupiny patří holandština, italština, španělština, němčina, francouzština, čeština a estonština. Slovníky jsou organizovány podobně jako WordNet. Navíc obsahují vztahy mezi jednotlivými jazyky. Shodné synsety jsou propojeny mezi jednotlivými slovníky, proto lze slovo porovnat s výrazy v dalších jazycích.

2.5.2 Global WordNet Association

Tato veřejná asociace těží ze základů projektů *Princeton WordNet* a *EuroWordNet*. Organizace podporuje vytváření slovníků pro další jazyky a poskytuje jim potřebné informace a vývojové prostředky.

Kapitola 3

Analýza

3.1 Program JWord 3.0

Program JWord [9], jehož inovací se tato bakalářská práce zabývá, slouží jako rozhraní pro databázi WordNet. Program byl vyvinut na univerzitě George Washingtona a nachází se ve verzi 3.0. Autorem programu je student Kunal Johar, který tento program vytvářel pod vedením profesora Rahula Simhy na katedře informatiky. Autoři programu na svém webu však přiznávají, že program má nedokonalé grafické uživatelské rozhraní. Tím také dali podnět ke vzniku této bakalářské práce.

3.2 Seznámení s programem JWord 3.0

Program JWord 3.0 byl napsán v jazyce Java. Data pro svou činnost získává ze souborů. Jeho grafické prostředí využívá knihovnu Javy AWT a výsledky hledání zobrazuje program v grafické komponentě JTree. Obsahuje v sobě několik slovníků:

- WordNet
- Rodget's Thesaurus
- Rodget's Thesaurus using WordNet Lemma
- Root Book

Uživatel si může nastavit, v kterých slovnících bude vyhledávat.
Uživatel aplikace v programu nalezne pouze následující informace:

- WordNet - významy slova, synonyma
- Rodget's Thesaurus - synonyma a jejich definice
- Rodget's Thesaurus using WordNet Lemma
- Root Book

3.2.1 WordNet

Program při vypisování významů zadaného slova, rozdělí nalezené informace podle slovního druhu (podstatná jména, slovesa, přídavná jména, příslovce). U každého slovního druhu vypíše jeho význam a pokud existuje i příklad použití. Dalším rozkliknutím větve stromu získáme i synonymum k tomuto případu použití zadaného slova.

3.2.2 Rotget's Thezaurus

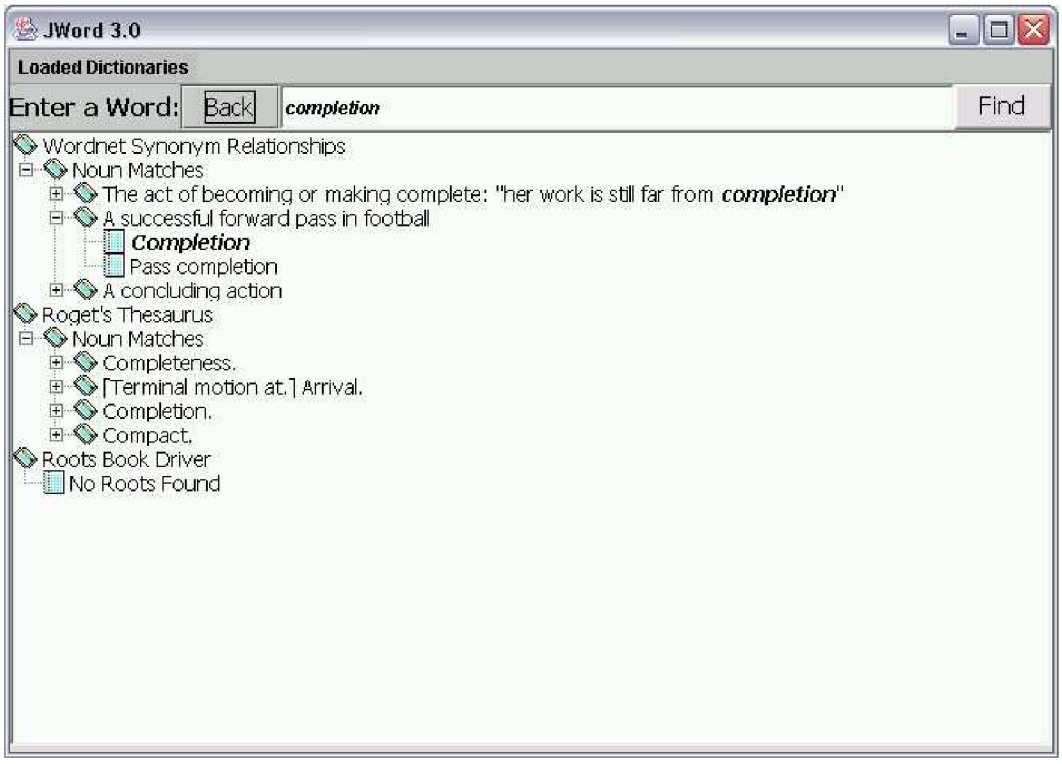
Rotget's Thezaurus také rozděluje výsledky podle slovního druhu. Vypisuje synonyma a k nim po rozkliknutí i definice.

3.2.3 Root Book

Tato část programu není funkční.

3.3 Funkčnost programu JWord 3.0

Uživatelské rozhraní programu JWord 3.0. má mnoho nedostatků, které ho dělají takřka nepoužitelným. Program je obtížně ovladatelný jelikož významy slov se přepisují a nelze tedy zadávat více slov po sobě. Sám uživatel může i informace, které zobrazuje komponenta JTree, přepisovat. Program se tak stává velmi nepřehledným. Na obrázku 3.1 můžete vidět ukázku uživatelského rozhraní.



Obrázek 3.1: JWord 3.0 a jeho uživatelské rozhraní

Kapitola 4

Návrh implementace

4.1 Možnosti uložení dat

Pro program JWord přicházely v úvahu dvě možnosti uložení dat pro slovník WordNet:

a) data uložená v souborech

b) data uložená v databázi

Oba způsoby uložení dat mají své výhody i nevýhody. Vystihovat by je měly následující odstavce.

4.1.1 Data uložená v souborech

Tohoto způsobu využívá program JWord 3.0. Data pro slovník WordNet jsou uložena v několika souborech. Každý slovní druh popisuje několik souborů. První soubor (nazvaný zkratkou slovního druhu s příponou idx) obsahuje abecední seznam slov. Na řádku za slovem se nachází několik značek včetně čísla synsetu. Pomocí tohoto čísla lze v dalším souboru (přípona dat) naleznout synsety hledaného slova i jejich definice a příklady použití.

Pomocí dalších operací se značkami uvedenými u hledaného slova v souboru můžeme nalézt i další informace. Mezi tyto informace patří například synonyma, antonyma, hypernyma a hyponyma zvoleného slova. Po zadání slova potřebujeme tedy projít soubory všech slovních druhů a pokusit se najít číslo jeho synsetu. Pokud může slovo nabývat významu více slovních druhů, pak musíme i synsety hledat ve více souborech.

Samotné vyhledávání v souboru lze realizovat několika způsoby. Prosté *sekvenční vyhledávání* není příliš vhodné pro svou časovou náročnost. Při použití vyhledávacího algoritmu lze dosáhnout lepších výsledků. Jelikož jsou data v souborech seřazena, můžeme využít například *binárního vyhledávání*.

Binární vyhledávání

Tento algoritmus najde medián (hodnotu, jež dělí řadu podle velikosti seřazených výsledků na dvě stejně početné poloviny), porovná ho s hledanou hodnotou a podle výsledku hledání pokračuje buď v horní nebo dolní polovině souboru. Tento algoritmus pokračuje rekurzivně od začátku dokud není nalezena hledaná hodnota nebo již není co půlit (hodnota se v souboru nevyskytuje). Složitost binárního vyhledávání je v nejhorším případě logaritmická.

	typ vyhledávání	
hledané slovo	**sekvenční vyhledávání**	**binární vyhledávání**
”make”	13 sec	3 sec
”game”	9 sec	1 sec

Tabulka 4.1: Tabulka s porovnáním sekvenčního a binárního vyhledávání

- Algoritmus pro binární vyhledávání (pro názornost v jazyce python)

```
def binarySearch(seznam, hodnota, vlevo, vpravo):
  while vlevo <= vpravo:
      stred = (vpravo + vlevo) / 2
      if seznam[střed] == hodnota:
          return True
      if hodnota < seznam[střed]:
          vpravo = střed - 1
      else:
          vlevo = střed + 1
  return False
```

V tabulce 4.1 lze vidět přibližnou dobu, za kterou algoritmus nalezne význam slova, jeho synonyma, hypernyma a hyponyma. Z tohoto hrubého porovnání vyplývá úspornost binárního vyhledávání. Čas zahrnuje pouze vyhledání v souborech pro jeden slovní druh, v tomto případě jde o údaje o slovesu *make* a podstatném jménu *game*.

4.1.2 Data uložená v databázi

Výhodou pro data uložená v databázi je jejich dostupnost pomocí SQL dotazů. Existuje několik databází lexikálního slovníku WordNet. Pro můj program jsem si vybral databázi MySQL WordNet 3.0. [2]. Tato databáze se skládá z několika tabulek. Pro náš program jsou užitečné zejména tabulky word, sense, synset, semlinkref, sample:

- word - tabulka má dva sloupce *wordid* a *lemma*. Sloupec lemma obsahuje seznam slov databáze WordNet a ve sloupci wordid lze najít jejich identifikační číslo. Databáze pracuje jen s těmito čísly, proto potřebujeme pro získání id slova vždy použít tabulku word.

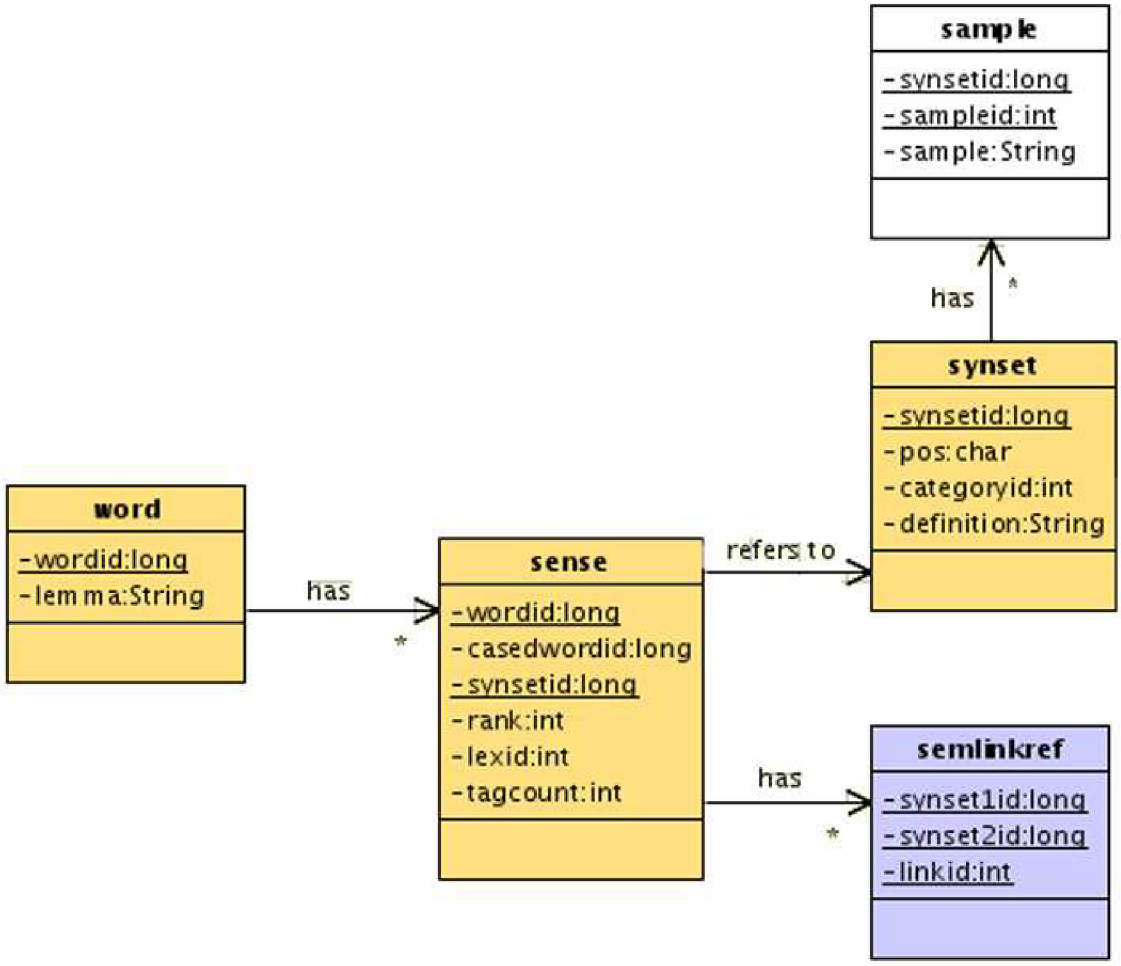
- sense - tabulku tvoří šest sloupců, nejdůležitější jsou dva z nich. Sloupce *wordid* a *synsetid* tvoří propojení mezi slovy a jejich synsety. Tuto tabulku lze využít při hledání id slov od zadaného synsetu, ale i naopak pokud hledáme synsety od zadaného id slova.

- synset - tabulka se skládá ze čtyř sloupců, za důležité považuji tři. Sloupec *synsetid* určuje číslo synsetu, pro které lze v tabulce zobrazit jeho význam (sloupec *definition*) a také jeho slovní druh (sloupec *pos*). Slovní druh je určen zkratkami (n - podstatné jméno, v - sloveso, a - přídavné jméno, r,s - příslovce).

- semlinkref - tabulka obsahuje tři sloupce a určuje vztah mezi synsety (sloupce *synset1id*, *synset2id*, *linkid*).

- sample - tabulka, která ke každému synsetu (sloupec *synsetid*) přiřazuje příklad použití daného slova (sloupec *sample*).

Dokumentace k databázi MySQL WordNet 3.0 obsahuje i několik hotových SQL dotazů. Tyto dotazy lze použít například pro získání synonym, antonym, hypernym a hyponym. Podstanou část schéma databáze můžeme vidět na obrázku 4.1. Celé schéma databáze MySQL WordNet 3.0 naleznete na konci práce v sekci přílohy na obrázku A.1.



Obrázek 4.1: Částečné chéma databáze MySQL WordNet 3.0 převzato z [2]

4.2 Porovnání rychlosti přístupu k datům

Dokud program nemusel hledat v souborech mnoho informací (vyhledal například jen významy slova), bylo vyhledávání v souborech uspokojivé. Při navyšujících se požadavcích

na program (hledání synonym, antonym, hypernym, hyponym a příkladů použití) není již vyhledávání v souborech dostačující. Pro naše účely se nabízí využití některé z databazí WordNetu. Výsledky použití databáze byly na tolik přívětivé, že pozitiva převýšila veškerá negativa, která toto řešení přináší.

4.3 Implementační jazyk

Pro implementaci výsledného programu byla navržena metodologie objektově orientovaného programování. Konkrétně byly doporučeny dva programovací jazyky:

a) programovací jazyk Java

b) rozhraní .NET

4.3.1 Programovací jazyk Java

Programovací jazyk Java [6] prošel od svého vzniku neuvěřitelným rozvojem. Z hlediska naší aplikace má Java řadu výhod. Do výhod Javy řadíme možnost využít knihovnu Swing pro grafické uživatelské prostředí nebo podporu Javy pro připojení do databáze. Komunikaci s databazí MySQL zajišťuje knihovna *mysql connector*. Největší výhodou Javy je však její přenositelnost mezi různými platformami.

Mezi jednu z nevýhod Javy patří její větší paměťová náročnost, která se projevuje hlavně u menších programů. Do nevýhod můžeme zařadit pomalejší start aplikace způsobený tím, že prostředí musí program nejprve přeložit a poté spustit.

Důležitou součásti volby programovacího jazyka je i volba vývojového prostředí. NetBeans 6.0 poskytuje naprostý komfort jak pro programování tak i pro navržení uživatelského rozhraní.

4.3.2 Rozhraní .NET

Programovací jazyk .NET je založen na .NET Framework, který zajišťuje překlad a běh .NET programů. Do výhod jazyka .NET patří rychlost programu. Rozhraní .NET program nezkompiluje přímo do strojového kódu, ale do mezikódu. Do strojového kódu se program zkompiluje až při spuštění na konrétním počítači. Výhodou je tedy optimalizace pro konkrétní procesor. Kompilace probíhá na konkrétním počítači pouze jednou, protože zkompilovaný program se uloží a při každém dalším použití se program pouze spustí. První spuštění programu může trvat několik sekund, ale každé další spuštění může být už velmi rychlé.

Mezi nevýhody jazyka .NET řadíme jeho orientaci na operační systém Windows a tedy jeho obtížnou přenositelnost.

4.4 Vyhodnocení návrhu implementace

Výhodou využití souborů pro uložení dat pro slovník WordNet je samostatnost programu bez nutnosti připojovat se k databázi. Nevýhodou pomalejší vyhledávání, větší velikost výsledného programu. Aktualizace programu je u obou způsobů uložení dat přibližně stejně obtížná. Pro svou práci jsem se rozhodl přejít od vyhledávání dat v souborech k využití databáze. To ovšem znamenalo najít vhodnou databázi pro uložení dat slovníku WordNet

a vytvořit úplně nové programové rozhraní pro komunikaci s touto databází. Odměnou za tuto snahu by měl být rychlejší a přehlednější program.

Za implementační jazyk byla vybrána Java. Tomuto rozhodnutí předcházelo zvážení všech výhod a nevýhod navržených programovacích jazyků. Java obsahuje množství využitelných knihoven a je přenositelná mezi platformami. Důležitou roli při rozhodování sehrál i fakt, že původní program JWord 3.0 byl také napsán v programovacím jazyce Java.

Kapitola 5

Imlplementace

Výsledný program se skládá ze dvou oddělených částí:

a) programové rozhraní - funkce pro komunikaci a vyhledávání v databázi využívají metody, které výsledná data zpracují.

b) grafické uživatelské rozhraní - poskytuje nadstavbu nad programovým rozhraním. Graficky zobrazuje data uživateli.

5.1 Programové rozhraní

Z návrhu implementace vyšlo najevo, že pro uchování dat lexikálního slovníku bude nejlepší využít databázi. Pro program JWord jsem vytvořil vlastní knihovnu, která zajišťuje komunikaci s databází. Zahrnuje funkce pro připojení k databázi a funkce, které vykonávají SQL dotazy. Tato knihovna také provadí veškeré potřebné operace se získanými daty a grafickému rozhraní předává vždy pouze informace, které budou zobrazeny uživateli.

Pro program JWord přicházelo v úvahu několik druhů databáze WordNet. Jako nejvhodnější byla vybrána databáze MySQL WordNet 3.0. [2] a to díky snadné instalaci i aktualizaci a své licenci. Tato databáze je přístupná ze sítě FIT a nachází se na serveru minerva2.

Knihovnu pro programové rozhraní tvoří tři provázané skupiny funkcí. Funkce pro komunikaci s databází využívají metody, které v databázi vyhledávají. Data těchto funkcí využívají metody pro zpracování dat.

a) funkce pro komunikaci s databází

b) funkce pro vyhledávání dat v databázi

c) funkce pro zpracování dat

5.1.1 Funkce pro komunikaci s databází

Níže uvedený seznam funkcí využívá knihovna pro programové rozhraní k připojení do databáze a pro provádění dotazů v databázi.

seznam prototypů funkcí pro komunikaci s databází

- public static Statement Connect(String server) - metoda, která zajišťuje připojení do databáze
- public static ResultSet MakeQuery(Statement mystmt, String query) - metoda, která vykoná databázový příkaz

5.1.2 Funkce pro vyhledávání dat v databázi

Základním stavebním kamenem knihovny pro programové rozhraní jsou funkce, které implementují SQL dotazy pro databázi. Všechny využívají připojení do databáze, následně vykonají svůj dotaz. Získaná data jsou uložena do kolekcí [4]. V případě jednodušších metod jsou získané informace vráceny v řetězci.

seznam prototypů funkcí pro vyhledávání dat v databázi

- public static String FindLemma(String wordid) - metoda, která najde slovo podle zadaného id
- public static String FindWordId(String word) - metoda, která najde id slova hledaného slova
- public static String FindSynsetId(String wordid) - metoda, která najde id synnsetu podle zadaného id hledaného slova
- public static List FindSynonyms(List synsets) - metoda, která najde všechny id synonym od zadané kolekce synsetů
- public static List FindSynsets(String wordid) - metoda, která najde id všech synsetů od zadaného slova
- public static List FindSynonymsIdFromSynsets(String synsetid) - metoda, která najde id všech synonym k zadanému id synsetu
- public static List FindSynonymsFromSynsets(String synsetid, String word) - metoda, která najde všechna synonyma k zadanému id synsetu
- public static List FindSense(List synsets) - metoda, která najde významy k zadaným id synsetů

Všechny tyto funkce slouží pouze k získávání dat. Implementují tedy SQL dotazy pro databázi MySQL WordNet 3.0. Získané informace využívají funkce pro zpracování dat.

5.1.3 Funkce pro zpracování dat

Soubor těchto funkcí využívá metod, pro vyhledávání v databázi. Data, která od těchto metod získávají, dále zpracovávají do podoby vhodné pro grafické uživatelské rozhraní.

seznam prototypů funkcí pro zpracování dat

- public static List GetSense(String word) - metoda, která najde všechny významy od zadaného slova
- public static List GetSynonyms(String word) - metoda, která najde synonyma od zadené kolekce synsetů
- public static List GetAntonyms(String word) - metoda, která najde antonyma od zadaného slova
- public static List GetTypes(String word) - metoda, která najde hyponyma od zadeného slova
- public static List GetTypeOf(String word) - metoda, která najde hypernyma od zadeného slova
- public static List GetWordClass(List Col, char wc) - metoda, která vybere z kolekce zadaný slovní druh

Ze všech uvedených funkcí, jsou nejzajímavější funkce pro zpracování dat. Tyto metody může bezpečně použít kdokoliv bez znalosti databáze MySQL WordNet 3.0.

5.1.4 Implementace SQL dotazů

Jednotlivé funkce programového rozhraní implementují i SQL dotazy pro vyhledávání v databázi MySQL WordNet 3.0.

SQL dotazy využité z dokumentace MySQL WordNet 3.0

Tyto SQL dotazy jsou součástí dokumentace k databázi MySQL WordNet 3.0. Uvádím nejdříve dotazy pro vyhledání hypernym a hyponym s příkladem pro slovo *horse*:

- hypernyma (slova s obecnějším významem)

```
select se1.rank,w2.lemma
from word w1
left join sense se1 on w1.wordid = se1.wordid
left join synset sy1 on se1.synsetid = sy1.synsetid
left join semlinkref on sy1.synsetid = semlinkref.synset1id
left join synset sy2 on semlinkref.synset2id = sy2.synsetid
left join sense se2 on sy2.synsetid = se2.synsetid
left join word w2 on se2.wordid = w2.wordid
where w1.lemma = 'horse'
and sy1.pos = 'n'
and semlinkref.linkid = 1
order by se1.rank asc;
```

- hyponyma (slova s konkrétnějším významem)

```
select se1.rank,w2.lemma
from word w1
left join sense se1 on w1.wordid = se1.wordid
left join synset sy1 on se1.synsetid = sy1.synsetid
left join semlinkref on sy1.synsetid = semlinkref.synset1id
left join synset sy2 on semlinkref.synset2id = sy2.synsetid
left join sense se2 on sy2.synsetid = se2.synsetid
left join word w2 on se2.wordid = w2.wordid
where w1.lemma = 'horse'
and sy1.pos = 'n'
and semlinkref.linkid = 2
order by se1.rank asc;
```

Dále byl implementován SQL dotaz pro vyhledání synonym, zde je uveden s příkladem pro slovo *house*:

- synonyma (slova se stejným významem)

```
select synsetid, w2.lemma from sense
left join word as w2 on w2.wordid=sense.wordid
where sense.synsetid in
(
select sense.synsetid from word as w1
left join sense on w1.wordid=sense.wordid
where w1.lemma='house'
)
and w2.lemma<>'house';
```

Provedení tohoto dotazu trvalo v databázi neúnosně dlouhou dobu, proto musel být nahrazen několika jednoduššími SQL dotazy. Tyto jednodušší SQL dotazy se však musí opakovat vícekrát pomocí cyklu a jsou uvedeny v sekci SQL dotazů, které již nejsou součástí dokumentace k databázi MySQL 3.0.

Posledním použitým SQL dotazem, je dotaz pro vyhledání antonym, zde je uveden příklad pro slovo *black*:

- antonyma (slova s opačným významem)

```
select se1.rank, w2.lemma, sy1.definition, sy2.definition from word
w1 left join sense se1 on w1.wordid = se1.wordid
left join synset sy1 on se1.synsetid = sy1.synsetid
left join lexlinkref on sy1.synsetid = lexlinkref.synset1id
and w1.wordid = lexlinkref.word1id
left join synset sy2 on lexlinkref.synset2id = sy2.synsetid
left join sense se2 on sy2.synsetid = se2.synsetid
left join word w2 on se2.wordid = w2.wordid
where w1.lemma = 'black' and sy1.pos = 'a'
```

```
and lexlinkref.linkid =30
order by se1.rank asc;
```

Vlastní SQL dotazy pro MySQL WordNet 3.0

Následující dotazy již nejsou součástí dokumentace k databázi MySQL WordNet 3.0.

Dotaz pro získání slova po zadání jeho id. Pro názornost uvedeno slovo *make*, které má v databázi identifikační číslo 80925.

- nalezení slova podle id

```
SELECT lemma FROM word WHERE wordid = '80925';
```

Opačným dotazem získáme id slova. Pro názornost uvedeno opět slovo *make*.

- získání id podle zadaného slova

```
SELECT wordid FROM word WHERE lemma = 'make';
```

Dalším dotazem, je dotaz pro nalezení synonym. Tento dotaz se skládá ze dvou částí. Pro získání synonym hledaného slova potřebujeme nejprve zjistit identifikační čísla všech jeho synsetů. Provádění toho dotazu zajišťuje cyklus, který se opakuje dokud nezjistíme id všech synsetů. Uvadím příklad pro slovo *make*. Jeho id už známe - 80925.

- a) synonyma - vyhledání synsetů

```
SELECT synsetid FROM sense WHERE wordid = '80925';
```

Po obdržení informací o id synsetů lze získat id synonym hledaného slova. Tento dotaz je prováděn opět v cyklu, dokud nejsou nalezena všechna synonyma od hledaného slova. Za proměnou *synsets* jsou postupně dosazovány čísla všech synsetů. Jako výsledek získáme id všech synonym hledaného slova.

- b) synonyma - vyhledání id synonym

```
SELECT wordid FROM sense WHERE synsetid = 'synsets';
```

Když známe id synonym můžeme již snadno zjistit slova samotná.

Následující dotaz vyhledá v databázi význam slova, tedy jeho definici. Hodnota pro jeden ze synsetů našeho příkladu (slova *make*) je 100340463.

- vyhledání definice slova

```
SELECT definition FROM synset WHERE synsetid = '100340463';
```

Podobným dotazem obdržíme i příklad použití ve větě. Opět stačí zadat hodnotu pro synset jehož příklad hledáme.

- vyhledání příkladu slova

```
SELECT sample FROM sample WHERE synsetid =  '100340463';
```

Všechny uvedené SQL dotazy implementují funkce knihovny pro programové rozhraní. Slouží spíše jako demonstrace struktury databáze WordNet MySQL 3.0.

5.2 Grafické uživatelské rozhranní

Grafické rozhraní umožňuje uživateli programu jednoduše a intuitivně ovládat program. Tvoří nadstavbu nad knihovnou pro programové rozhraní a využívá její funkce na získání potřebných dat. Tato data zobrazuje uživateli pomocí grafických komponent.

V programovacím jazyku Java je možné využít dvou grafických uživatelských prostředí [5]. Buďto staršího AWT (*Abstract Windowing Toolkit*) nebo novějšího JFC Swing (*Java Foundation Classes*) AWT je součástí jazyka Java od JDK 1.0, Swing od JDK 1.2. Pro program JWord jsem se rozhodl využít modernějšího prostředí Swing.

5.2.1 Prostředí Swing

Pro tvorbu grafického uživatelského rozhrání bylo využito několika komponent JFC Swing. Na základním okně (*JFrame*) jsou postupně využity tyto komponenty:

seznam využitých komponent prostředí Swing

- JTree
- JTabbedPane
- JComboBox
- JTextField
- JButton
- JLabbel

Hledané slovo se zadává do komponenty *JTextField*. Jeho vyhledání zajišťuje stisknutí tlačítka (komponenta *JButton*). Nejdůležitějším prvkem celého GUI jsou významy hledaného slova, které zobrazuje komponenta *JTree*. Uspořádání do stromu dostalo přednost před vypisováním do obyčejné *JTextArea* díky své přehlednosti a snažší manipulaci s daty. Pro znázornění příbuzných slov (synonym, antonym, hyponym a hypernym) používá program JWord komponenty *JTabbedPane*. Užvatel tak může pohodlně prohlížet jednotlivé typy slov. Pro uchování historie hledaných slov a připojení do databáze je v novém GUI programu JWord využito komponenty *JComboBox*. Jako popis některých událostí je použit *JLabbel*

5.2.2 Programová část GUI

Programová část GUI se skládá z několika bloků. Tyto části tvoří metody, které vkládají data do grafických komponent a obsluhují události vyvolané uživatelem.

- zobrazení významů slova
- příbuzná slova
- zadání hledaného slova
- historie
- připojení do databáze
- nastavení nového připojení

zobrazení významů slova

Hlavní část programu slouží k zobrazení významů hledaného slova. Získaná data zobrzazuje strom složený ze čtyř základních uzlů. Uzly obsahují významy slova podle toho v jakém se vyskytuje slovním druhu.

- *Nouns* - významy pro podstatné jméno
- *Verbs* - významy pro sloveso
- *Adverbs* - významy pro příslovce
- *Adjectives* - významy pro přídavné jméno

Tyto větve jsou naplněny daty ihned po zadání hledaného slova.

Metoda, nejprve získá data od knihovny pro programové rozhraní a poté data roztřídí podle slovního druhu. Na závěr přiřadí do každé větve odpovídající významy.

příbuzná slova

Podstatnou část programu tvoří panel pro zobrazení příbuzných slov. Tento panel má čtyři seznamy, do kterých vypisuje synonyma, antonyma, hyponyma a hypernma. K tomuto účelu je využito čtyř funkcí.

- static void printSynonym(String word)
- static void printAntonym(String word)
- static void printHyponym(String word)
- static void printHypernym(String word)

Tyto funkce získávají data z knihovny pro programové rozhraní a vkládají obdržená data do svých seznamů.

Pro urychlení programu se seznam zobrazující synonyma naplní zároveň s významy slov. Další kontaktování databáze proběhne až po kliknutí uživatele na některý z dalších seznamů

pro antonyma, hypernyma nebo hyponyma. Tuto situaci zajišťuje zaregistrováná událost *ChangeListener*.

Každý ze čtyř seznamů implementuje metodu *MouseListener*. Tato skutečnost umožňuje seznamům reagovat na kliknutí myši. Při dvojkliku tak program automaticky vyhledá označenné slovo.

Seznam obsahující synonyma hledaného slova má zaregistrovanou událost *ListSelectionListener*. Proto reaguje na označení položky seznamu. Pokud tedy uživatel zvolí některé ze synonym, strom obsahující významy slov se rozbalí až po větev, která je významem označeného synonyma.

zadání hledaného slova

Panel pro zadání slova, které bude zpracováváno se skládá ze tří částí. Tvoří je popisek, textové pole, kam uživatel napíše slovo určené ke zpracování a tlačítko. Uživatel může spustit vyhledávání buď stiskem tlačítka *Search* nebo stiskem klávesy *Enter*. To zaručuje obsluhovaná událost textového pole *ActionPerformed*.

Po udání povelu ke zpracování zadaného slova provede program metodu, která obslouží jak funkce pro grafickou komponentu zobrazení významů slova, tak i funkce pro grafickou komponentu zobrazující příbuzná slova. Po vykonání těchto funkcí obsahují grafické komponenty aktualizovaná data odpovídající pro nově zadané slovo.

historie

Historie zadaných slov realizuje komponenta *JComboBox*. Každé slovo, o kterém vyhledávme informace, program automaticky vkládá i do historie (pokud se však již v historii nenalézá). Pokud uživatel vybere některé slovo z historie, program se zachová stejně jako při zadání nového slova. Tuto službu poskytuje zaregistrovaná událost *ActionPerformed*.

připojení do databáze

Připojení do databáze také vyyužívá komponenty *JComboBox*. Po spuštění program načte všechna existující připojení ze souboru *connect.dat*. Vybráním jedné položky *JComboBox*u se program začne připojovat ke zvolené databázi. Tuto vlastnost zaručuje opět událost *ActionPerformed*. Stav připojení k databázi udává popisek v levém dolním rohu programu.

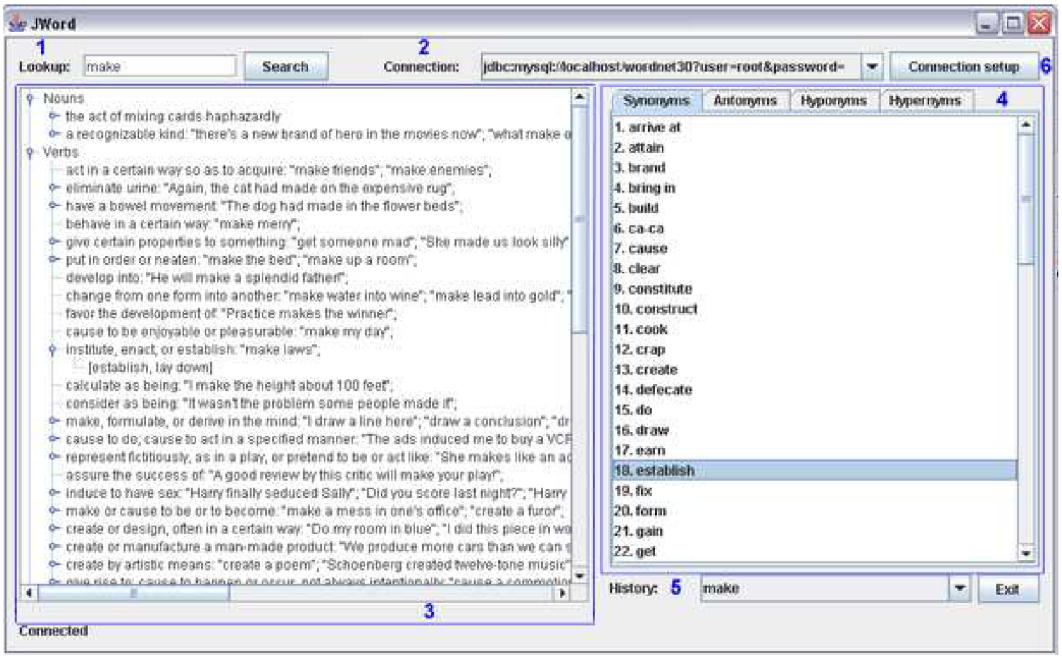
nastavení nového připojení

Program uživateli umožňuje i vytvoření svého vlastního připojení k databázi WordNet 3.0. Tuto možnost vyvolá uživatel stiskem tlačítka *Connection setup*. V novém okně pak vyplní potřebné údaje. Tyto údaje program uloží do souboru *connect.dat* ve formátu vhodném pro připojení knihovnou *mysql connector*.

5.2.3 Manuál ke GUI

Používání programu JWord je velmi intuitivní, přesto zde popíši základní ovládání tohoto programu. Úvodní okno programu, jak je vidět na obrázku 5.1, je rozděleno do několika částí.

1. Textové pole, do kterého uživatel vepisuje hledaný výraz. Pro vyhledání slova stiskněte tlačítko *Search*, které se nachází na pravo od textového pole nebo stiskněte *Enter*.



Obrázek 5.1: Nové GUI vytvořené pomocí Swing

2. Připojení k databázi MySQL Wordnet 3.0. Po spuštění programu vyberte pomocí rozbalovacího menu připojení ke své databázi nebo připojení vytvořte (6).

3. Strom, který zobrazuje nalezené výrazy hledaného slova. Obsahuje čtyři větve:

   - *Nouns* - výrazy, kde má hledané slovo význam podstatného jména
   - *Verbs* - hledané slovo je v těchto významech slovesem
   - *Adverbs* - významy pro příslovce
   - *Adjectives* - hledané slovo je ve tvaru přídavného jména

   Po rozkliknutí některé z větví, zobrazí program další větve s významy hledaného slova. U každého významu je uveden i příklad použití ve větě. Ten je vepsán do uvozovek. Synonyma hledaného slova, uživatel zjistí rozkliknutím větve s významem slova.

4. Panel se slovy příbuznými k hledanému slovu. Uživateli jsou zde v samostaných seznamech zobrazena synonyma (slova se stejným významem), antonyma (slova s opačným významem), hyponyma (slova s konkrétnějším významem), hypernyma (slova s obecnějším významem). Pokud uživatel poklepe na některé ze zobrazených slov, program toto slovo okamžitě vyhledá. U synonym pak po označení slova, program vyhledá jeho význam v levém okně a rozbalí strom až k tomuto významu.

5. Historie, do tohoto rozbalovacího menu program ukládá všechna hledaná slova. Pokud se chce uživatel k některému slovu vrátit stačí jej vybrat.

6. Připojení do databáze, po stisku tlačítka *Connection setup* se otevře nové okno. V tomto okně uživatel může nastavit vlastní připojení do databáze vyplněním základních informací do formuláře. To bude i po restartu programu uloženo v roletce připojení (2).

číslo pokusu	čas vyhledávání (sec)
1. pokus	0.9 sec
2. pokus	0.7 sec
3. pokus	0.7 sec
4. pokus	0.7 sec
5. pokus	0.6 sec

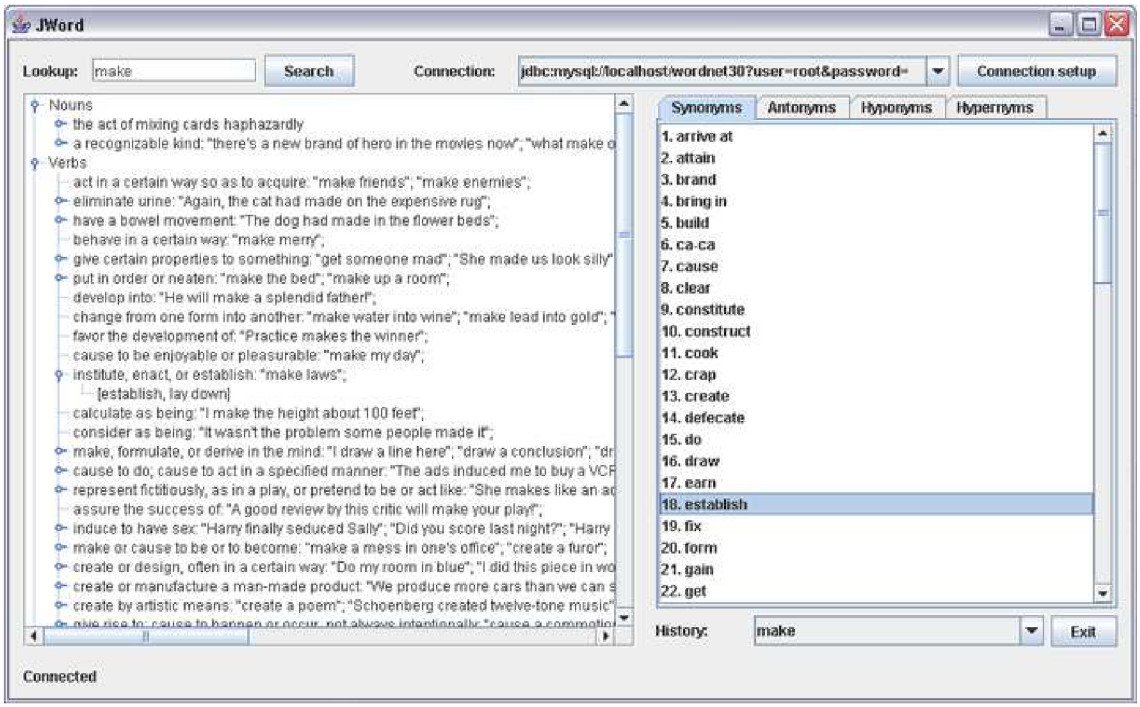
Tabulka 5.1: Čas vyhledání dat o slovu *make*

5.3 Testování

Každý program musí před svým dokončením projít testovací fází. Na testovaných příkladech je demonstrována i funkčnost programu.

5.3.1 Vyhledání informací o slovu *make*

Program zobrazil 2 významy pro podstatné jméno a 49 významů pro sloveso, jiné slovní druhy databáze pro toto slovo neobsahuje. Dále vypsal 69 synonym, 3 antonyma, 266 hyponym a 88 hypernym. V tabulce 5.1 uvádím čas, za který program zobrazil informace o slovu *make*. Průměrně trvalo vyhledání informací ke slovu *make* 0.7 sekund. Do toho času je zahrnuto vyhledání významů slova, synonym a přiřazení synonym k významům slova. Další vztahy (antonyma, hypernyma, hyponyma) se začnou hledat až po kliknutí uživatelem na položku seznamu pro daný vztah. Je tak učiněno právě kvůli úspoře času. Na obrázku 5.2 můžete vidět výsledek vyhledání informací pro zadané slovo.



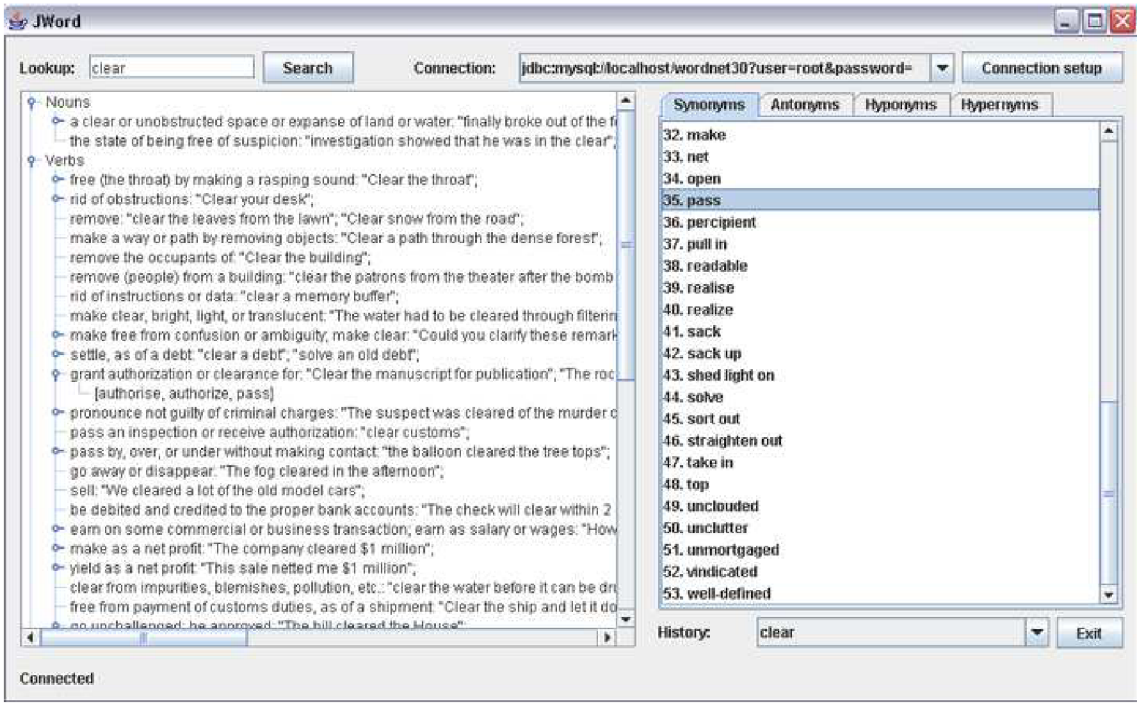
Obrázek 5.2: Vyhledání informací o slovu *make*

číslo pokusu	čas vyhledávání (sec)
1. pokus	0.8 sec
2. pokus	0.7 sec
3. pokus	0.7 sec
4. pokus	0.6 sec
5. pokus	0.5 sec

Tabulka 5.2: Čas vyhledání dat o slovu *clear*

5.3.2 Vyhledání informací o slovu *clear*

Program zobrazil 2 významy pro podstatné jméno, 24 významů pro sloveso, 15 pro příslovce a 4 pro přídavné jméno. Dále vypsal 53 synonym, 4 antonyma, 41 hyponym a 54 hypernym. V tabulce 5.2 uvádím čas, za který program zobrazil informace o slovu *clear*. Průměrně trvalo vyhledání informací ke slovu *clear* 0.667 sekund. Na obrázku 5.3 můžete vidět výsledek vyhledání informací pro zadané slovo.



Obrázek 5.3: Vyhledání informací o slovu *clear*

Kapitola 6

Závěr

Cílem práce byla inovace stávajícího grafického uživatelského rozhraní programu JWord 3.0. Již v průběhu analýzy vyšlo najevo, že vytvoření pouze GUI nebude optimální. Při analýze a návrhu implementace byl kladen důraz zejména na rychlost vyhledávání vlastností od zadaného slova. S přibývajícími požadavky na získávání informací nebylo původní vyhledávání v souborech již dostačující. Pro uložení dat lexikálního slovníku WordNet bylo, po zvážení všech možností, využito databáze MySQL WordNet 3.0. Značnou část vývoje aplikace zabrala tedy implementace nového programového rozhraní, které využívá tuto databázi. Ve druhé fázi proběhla implementace GUI, které je nadstavbou nad tímto programovým rozhraním.

Program pro svůj bezproblémový běh potřebuje připojení k databázi MySQL WordNet 3.0. Aplikace by měla sloužit hlavně v síti FIT, kde je databáze pro WordNet přístupná na serveru *minerva2*. Protože se připojení k databázi nastavuje přes uživatelské rozhraní, není problém připojit se k jákemukoliv jinému serveru s databází MySQL WordNet 3.0.

Nejdůležitější částí aplikace je vyhledání významů od zadaného slova. Aplikace dále zobrazuje kromě seznamu všech synonym i synonyma pro jednotlivé významy zadaného slova. Program uživateli vypisuje i další vztahy k hledanému slovu a těmi jsou seznamy všech jeho antonym, hyponym a hypernym. Jako doplněk aplikace lze úvést historii hledaných slov.

6.1 Možná rozšíření

Jako rozšíření práce lze navrhnout přizpůsobení programového rozhraní tak, aby mohlo obsáhnou i data pro další jazyky. Přímo se nabízí začlenit databáze z projektů jako je *EuroWordNet* a *Global WordNet Association*. Pro tento účel pak inovovat i grafické uživatelské rozhraní.

Jedno z dalších možných vylepšeních programu může být editační mód. Ten by mohl sloužit pro procházení, editaci a vkládání nových řetězců či celých synsetů do databáze. Toto vylepšení by však již vyžadovalo náročnější rozšíření programového rozhraní.

Literatura

[1] The Global WordNet Asoccation. [online]. [cit. 2008-04-12]. Dostupný z WWW: `http://www.globalwordnet.org/`.

[2] Bernard Bou. *A ready-to-use SQL database* [online]. [cit. 2008-04-14]. Dostupný z WWW: `http://wordnet.princeton.edu/links#SQL`.

[3] EuroWordNet. [online]. [cit. 2008-04-12]. Dostupný z WWW: `http://www.illc.uva.nl/EuroWordNet/`.

[4] P. Herout. *Java - bohatství knihoven.* České Budějovice: Kopp, 2006. 244 s. ISBN 978-80-7232-288-5.

[5] P. Herout. *JAVA - grafické uživatelské prostředí a čeština.* České Budějovice: Kopp, 2007. 352 s. ISBN 978-80-7232-328-9.

[6] P. Herout. *Učebnice jazyka Java.* České Budějovice: Kopp, 2007. 320 s. ISBN 978-80-7232-323-4.

[7] P. U. C. S. Laboratory. *Wordnet 3.0 database statistics* [online]. [cit. 2008-04-12]. Dostupný z WWW: `http://wordnet.princeton.edu/man/wnstats.7WN/`.

[8] J. Miller. *Introduction to WordNet: An on-line lexical database* [cit. 2007-10-02]. 1993. Dostupný z WWW: `http://wordnet.princeton.edu/5papers.pdf`.

[9] The George Washington University. *JWord 3.0* [online]. [cit. 2007-10-02]. Dostupný z WWW: `http://www.seas.gwu.edu/~simhaweb/software/jword/`.

[10] Semantic Web. [online]. [cit. 2008-04-05]. `http://www.w3.org/2001/sw`.

[11] Wikipedia. *Gottfried Wilhelm Leibniz* [online]. [cit. 2008-04-12]. Dostupný z WWW: `http://cs.wikipedia.org/wiki/Leibniz/`.

Dodatek A

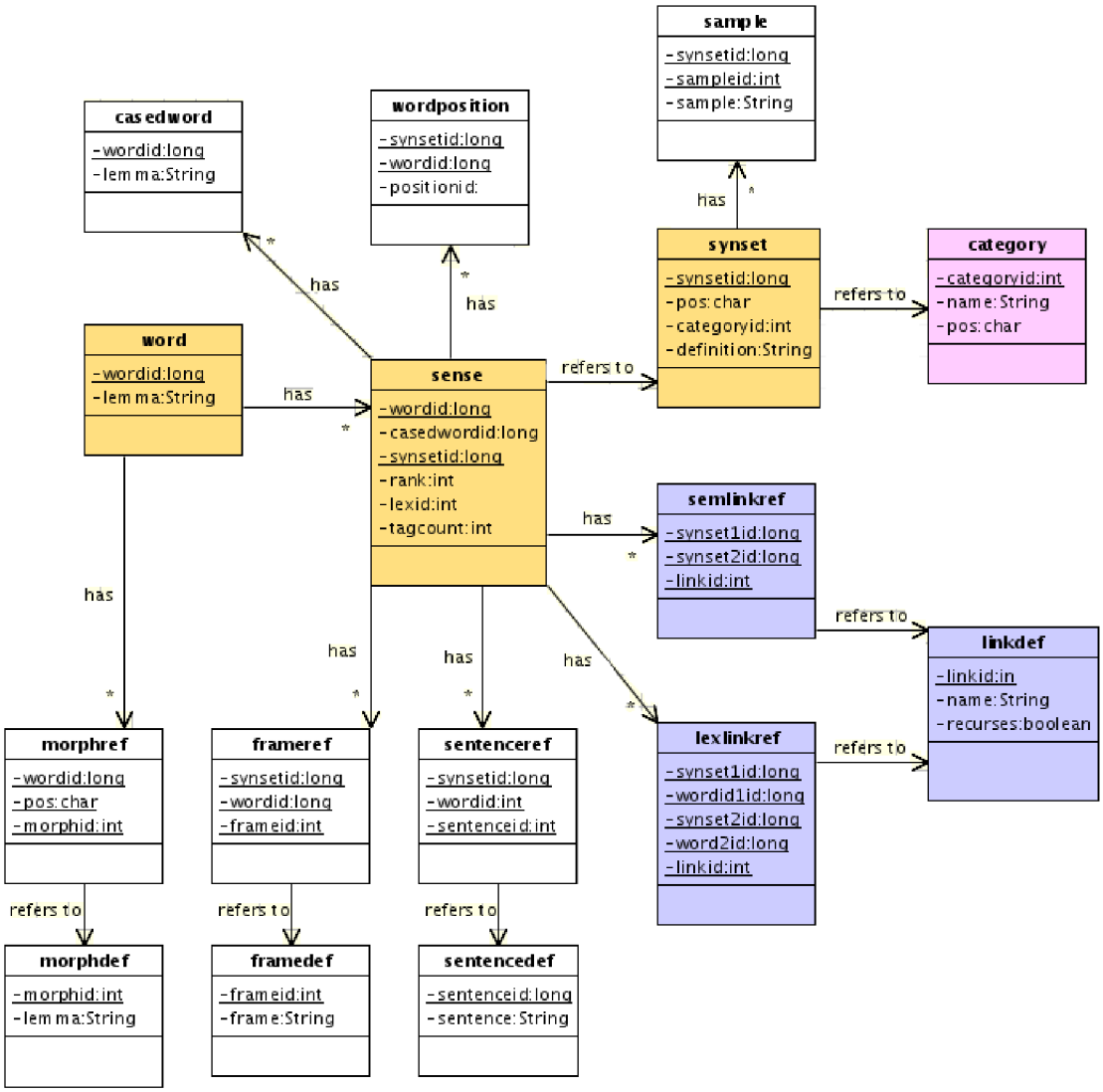
Přílohy

A.1 Obsah přiloženého CD

Obsah jednotlivých adresářů:

- /help - nápověda k programu ve formátu html
- /src - zdrojové kódy s dokumentací ve formátu JavaDoc
- /dist - distribuce projektu ve formě JAR archivu včetně potřebných externích knihoven
- /text - text této bakalářské práce ve formátu pdf

A.2 Schéma databáze



Obrázek A.1: Schéma databáze MySQL WordNet 3.0 převzato z [2]