Tool for Czech Pronunciation Generation Combining Fixed Rules with Pronunciation Lexicon and Lexicon Management Tool

Petr Pollák, Václav Hanžl

Czech Technical University in Prague
CVUT FEL K331 Technicka 2, 166 27 Praha 6 - Dejvice, Czech Republic
pollak@feld.cvut.cz, hanzl@feld.cvut.cz

Abstract

This paper presents two different tools which may be used as a support of speech recognition. The tool "*transc*" is the first one and it generates the phonetic transcription (pronunciation) of given utterance. It is based mainly on fixed rules which can be defined for Czech pronunciation but it can work also with specified list of exceptions which is defined on lexicon basis. It allows the usage of "*transc*" for unknown text with high probability of correct phonetic transcription generation. The second part is devoted to lexicon management tool "*lexedit*" which may be useful in the phase of generation of pronunciation lexicon for collected corpora. The presented tool allows editing of pronunciation, playing examples of pronunciation, comparison with reference lexicon, updating of reference lexicon, etc.

1. Introduction

One of the most important tasks within speech recognition is the generation of the pronunciation for given utterance. Both in training or in recognition phase, the information about the pronunciation of given or supposed utterance must be available. The pronunciation can be obtained either from a pronunciation lexicon or using a tool for automated conversion of orthographic transcription to the phonetic one. This paper is devoted to the solution of this problem for Czech language.

Czech has quite regular pronunciation, so the rules for automated generation of phonetic transcription (pronunciation) can be defined excepting some words, mainly the words of foreign origin. A special tool solving this problem "*transc*" was developed in Speech Processing Group at Czech Technical University in Prague. When the automated generation of phonetic transcription cannot be applied for a word with some pronunciation irregularities (due a conflict with the general rule), the special simple syntax of orthographic transcription can be used to re-define automatically generated regular pronunciation. This syntax is described in section 3.

The second approach of utterance pronunciation generation is based on pronunciation lexicon. This approach is not optimal for Czech due to high number of inflections in the language because it leads to growing size of the lexicon. On the other hand, irregular pronunciation can be easily involved in the lexicon and any special orthography transcription isn't required in this case. This is the main advantage of the lexicon approach. Because the creation of pronunciation lexicon is really hard work with necessary manual checks we have designed lexicon management tool "*lexedit*" with quite simple control which is described in the section 4.

Finally, we use small lexicon as a part of tool for automated pronunciation generation ("*transc*"), where the words with pronunciation exceptions are stored.

2. Phonetical alphabet for Czech

It is fact that any officially defined standard of Czech phoneme inventory has not been defined yet. We can find several attempts to its definition, e.g. in (Nouza et al., 1997) or (http://www.phon.ucl.ac.uk/home/sampa/czech-uni.htm), but any alphabet has not been accepted as the standard. The latest Czech SAMPA proposal is described in (http://noel.feld.cvut.cz/sampa/CZECH-SAMPA.html) That's reason why we must briefly describe the alphabets used for phonetic transcription in this work.

We work with two different phonetical alphabets which differ only in used symbols for defined inventory of phonemes with only one exception. It will be discussed later.

The first one is IPA CTU (Internal Phonetic Alphabet created at Czech Technical University in Prague) which is very close to the alphabet published in (Nouza et al., 1997). It is used historically by many tools written at our laboratory and it seems to be very convenient especially for the following reasons. Each phoneme is always represented by single character and the symbols are very close to the transcription of the pronunciation using standard Czech orthography. Consequently, it mainly respects standard Czech pronunciation rules, i.e. applying "*transc*" to the word in this phonetic transcription we obtain the same result. The exception from this rule can be find for special phonemes which are represented by upper case letters. These phonemes are quite rare and they may be represented usually by equivalents with two phonemes written by lower case letters ("E - eu", "A - au", "Z - dz", etc.), see tab. 1. Many times it can be done because the threshold between these two pronunciation is often quite soft, i.e. we can observe smooth passing of two phonemes "eu" into single phoneme "E".

As the first step to standardization, CZECH SPEECHDAT SAMPA was defined. It originates from IPA CTU but it uses already existing symbols in SAMPA alphabets for equivalent phonemes from other languages. This alphabet were defined within SpeechDat(E) project, see (Černocký et al., 2000). The correspondence between these two alphabets is in the tab. 1. There is only one small difference in pho-

neme inventory. CZECH SPEECHDAT SAMPA doesn't resolve voiced and unvoiced allophones of "ř".

CZECH SPEECHDAT SAMPA is the alphabet which is usually used at the international level. But the transcription of Czech pronunciations is very strange. That is reason why for our private purposes we use IPA CTU. The Czech reader may appreciate very convenient phonetical transcription which is very close to Czech orthography and which is generally easily understandable to Czech people, see tab 1. When we annotated Czech SpeechDat database on the pronunciation level we made the experience that people without any special education in this area could easily understand this transcription after short training period.

3. *transc* - tool for pronunciation generation

Czech language has generally regular pronunciation which corresponds closely with written form of the text. Some irregularities appear mainly within the words of foreign origin. The most frequent case are personal and company names, scientific terminology, technical terms, etc. Some of these irregularities are quite systematic and they can be handled by regular rules. Others can only be covered by exception lexicon lookup.

Czech, similarly as other Slavic languages, has highly inflective nature. Typically, nouns, adjectives, pronouns, and numerals are declined. They may have up to 7 different forms both in plural and singular. Verbs may be conjugated in 6 different forms for each tense, sometimes the forms differ according to gender. Using the pronunciation lexicon, its size rapidly increases. Moreover, many forms of the words differ only in terminations with regular pronunciation while their bases remain unchanged. That's the reason why the usage of fixed pronunciation rules in a conversion algorithm seems to be more efficient for the generation of written text pronunciation than the lexicon approach.

The most complicated part of "regular" pronunciation generation is proper handling of voiced/voiceless assimilations in consonant sequences. Consonant sequences may be quite long in Czech words and most often the sequence assimilates to be either voiced or unvoiced as a whole. However individual consonants behave in different ways during this assimilation – they may be more or less dominant and force its neighbor to share voiced/voiceless quality with them, they may be more or less likely to be dominated, these relations may differ in forward and backward directions and may depend on context. Chains of assimilations may cross word boundaries. On the other hand, certain consonants may split consonant sequences into parts which differ in voiced/voiceless quality. Overall number of combinations is immense and software realization of these "common and simple" assimilations gets rather complicated. See (Palková, 1984) for extensive list of details.

Most transformations leading from orthographic text to list of phonetic symbols are accomplished using sets of context-sensitive grammar rules. Each rule replaces certain sequence of symbols by other sequence of symbols and is applied as many times as possible. Once the rule can do no more replacements, next rule in set is applied. All rules in the set are applied in order from the first rule in the set to the last rule. This differs slightly from usual concept of context sensitive grammars (which permits any order of rules) but leads to much reduced complexity and also enables us to use smaller set of symbols – each 8-bit character may serve as both terminal and non-terminal symbol with slightly different meanings in various stages of processing.

3.1. Input format

The orthographic transcription at the input of this tool must obey conventions which can be summarized as follows:

- The convertible text must be written in regular words with standard Czech syntax. The transcription can contain punctuation.

- The words with irregular pronunciation are typed with following syntax: "(*orthography*/*pronunciation*)". The *pronunciation* is typed using standard Czech alphabet in any way which yields the correct result according to regular pronunciation rules, i.e. according to rule "write what you hear".

  Ex: `(monitor/monytor)`

- Known irregularities are included in internal lexicon of exceptions. This list can be extended by user defined external lexicon which includes transcriptions according to previously defined rule. The growing database of irregularities continuously decreases the probability of bad orthographic-to-phonetic conversion, i.e. it minimizes the necessity of special syntax in the orthography transcription of new utterances.

- The transcription is generally case insensitive. Whole text is in the beginning converted to lower case form. However sometimes it may be useful to distinguish between two words which differ just in the case of the letters. These words may be included in the external lexicon. Order of lexicon lookup and conversion to lowercase is problematic to decide, each possibility has some disadvantages – therefore we repeat the lookup both before and after the conversion to lowercase to enable case sensitive handling where needed and case insensitive handling otherwise.

  Ex:
  ```
  Detektor řeči značíme VAD.
  Detektor řeči značíme vé á dé.
  detektor řečy značýme vé á dé
  ```

  Ex:
  ```
  Výrobek má mnoho vad.
  Výrobek má mnoho vad.
  výrobek má mnoho vat
  ```

- All numbers should be transcribed in words.

- There are special characters and symbols with different meanings:

$xx	spelled letter
_	disjunctor
[yyy]	non-speech events and special noises
*, ~ , %	special marks for mispronunciation, word truncation, etc. (SpeechDat syntax)

CTU IPA		Czech SpeechDat SAMPA		Ortography
a	táta	a	ta:ta	*táta*
á	táta	a:	ta:ta	*táta*
A	Ato	a_u	a_uto	*auto*
b	bába	b	ba:ba	*bába*
c	cesta	t_s	t_sesta	*cesta*
č	čyHá	t_S	t_Sixa:	*čichá*
d	jeden	d	jeden	*jeden*
ď	ďelat	J\	J\elat	*dělat*
e	lef	e	lef	*lev*
é	méňe	e:	me:Je	*méně*
E	Ero	e_u	e_uro	*euro*
f	fAna	f	fa_una	*fauna*
g	guma	g	guma	*guma*
h	hat	h\	h\at	*had*
H	Hudý	x	xudi:	*chudý*
j	dojat	j	dojat	*dojat*
k	kupec	k	kupet_s	*kupec*
l	ďelá	l	J\ela:	*dělá*
m	máma	m	ma:ma	*máma*
M	traMvaj	F	traFvaj	*tramvaj*
n	výno	n	vi:no	*víno*
N	baNka	N	baNka	*banka*

CTU IPA		Czech SpeechDat SAMPA		Ortography
ň	koňe	J	koJe	*koně*
o	kolo	o	kolo	*kolo*
O	pOze	o_u	po_uze	*pouze*
p	pupen	p	pupen	*pupen*
ó	óda	o:	o:da	*óda*
r	bere	r	bere	*bere*
ř	moře	P\	moP\e	*moře*
Ř	keŘ	P\	keP\	*keř*
s	sut	s	sut	*sud*
š	duše	S	duSe	*duše*
t	dutý	t	duti:	*dutý*
ť	kuťyl	c	kucil	*kutil*
u	duše	u	duSe	*duše*
ú	kúl	u:	ku:l	*kůl*
v	láva	v	la:va	*láva*
y	byl, byl	i	bil	*bil, byl*
ý	výtr, lýko	i:	vi:tr, li:ko	*vítr, lýko*
z	koza	z	koza	*koza*
Z	leZgde	d_z	led_zgde	*leckde*
ž	rúže	Z	ru:Ze	*růže*
Ž	ráŽa	d_Z	ra:d_Za	*rádža*

Table 1: Correspondence between CTU IPA and SAMPA

3.2. Stages of processing

Individual transformations performed during conversion of orthographic text (with certain special marks) to pronunciation can be approximately summarized as the following steps (throughout these steps, CTU IPA is used as internal representation):

- Process marks for non-speech events, special noises, mispronunciation and word truncation ("[yyy]", "*", "~", "%". These marks just go through unchanged and they appear unchanged at corresponding places of output.
- Replace "(*orthography*/*pronunciation*)" by "*pronunciation*" only. Including this step rather early in processing makes possible formats of pronunciation much more readable – user does not have to care about tiny details like certain mandatory allophones; following stages will most likely do a better job than the user could do. It is also possible to change level of details (using or not using certain allophones) without changing the input text.
- Do some normalizations of text – delete interpunction, shrink sequences of spaces etc.
- First exceptions lexicon lookup and replacement.
- Conversion of text to lowercase.
- Second exceptions lexicon lookup and replacement. This stage covers capitalized words at the beginning of sentences.
- Conversion "ch" → "H". At this stage we started reuse of capital letters as additional phone symbols.
- Transcription of certain patterns often found in foreign word (internal lexicon). Foreign words not covered by dictionary lookup can still be processed right in this stage. (Conversion of "ch" should precede as some of these patterns ending in "c" would cause errors here.)
- Expansions "x" → "ks" and "q" → "kv".
- Assimilation exceptions – this stage handles marginal but yet regular patterns not covered by rules below.
- Replacement of "d", "t", "n" by "ď", "ť", "ň" when followed by one of "i", "í ", "ě". (This is not true assimilation, just quirk of Czech orthography.)
- Handling of "b", "p", "v", "f", "m" followed by "ě".
- Ends of words made voiceless (unless interword influence happens).
- Backward assimilation – certain phones made voiced, e.g. "sb" → "zb", "kd" → "gd".
- Backward assimilation – certain phones made voiceless. This is the most common type of assimilation which happens inside words. Dominant phones are "k", "č", "s", "š", "p", "t", "H" and they influence preceding phones "h", "v", "z", "d", "ž", "b".
- Forward assimilation – much less frequent, most notably "sh" → "sH" (but even this depends on dialect).
- "dž", "dz", "au" which survived till this point are converted to "Ž", "Z", "A". This replacement is in principal similar to replacement of "ch" made earlier but is much less sure and can be prevented by rules between.

- Handling of certain allophones, e.g. “M” (allophone of “m”) and “N” (allophone of “n”).
- Remove disjunctors “_”. Disjunctors may be used in dictionary and “(/)” to prevent false assimilations and false interpretation of letter couples, e.g. “d_ž” where “d” and “ž” should be separate phones.
- Conversion of CTU IPA symbols to SAMPA if required.

The above set of transcription rules explains the logic behind “*transc*” pronunciation generation but is in no means exhaustive; program itself covers much wider range of slight dependencies and irregularities than we can describe here.

3.3. Processing options available to user

An updated version of “*transc*” contains many options which allow effective usage of this tool for different orthographic transcriptions.

It reads input data from `stdin` stream and writes output to `stdout` stream so it can be easily included in a pipe for text processing during speech recognition.

These options can tailor “*transc*” output for particular use:

- The output can be either in CTU IPA alphabet or in SAMPA.
- Use of capital letters as additional phone symbols may be disabled, thereby making “*transc*” output suitable as “*transc*” input. This option may be used to generate example pronunciations usable in “(/)” format.
- External exceptions lexicon can be enabled (and selected) or disabled.
- Internal foreign words patterns can be disabled.
- Interword assimilations can be considered instead of simple word-end unvoicing.

4. Pronunciation lexicon - generation and management tools

Although fixed conversion rules seems to be more efficient to obtain phonetic transcription, many approaches require lexicon of pronunciation for given corpus. Typically, when we use the algorithms derived from English based recognition where the usage of pronunciation lexicon is the standard. From this point of view, pronunciation lexicon should be the part of internationally distributed speech corpora, e.g. databases of SpeechDat family (Pollák and Černocký, 1999).

Main advantages and disadvantages of lexicon solution for generation of phonetic transcription can be summarized in following points:

- the size of lexicon increases (due to repetition of very similar words),
- different pronunciation variants of one word cannot be resolved from text orthography,
+ it doesn’t need special orthography for irregular pronunciation,
+ it is standard approach in many working speech recognition systems for other languages.

4.1. Generation of lexicon

The creation of pronunciation lexicon is usually the work for an expert in phonetics or linguistics. Concerning the fact that many people don’t pronounce words correctly, real pronunciation of recorded speech data can be quite different from the regular one. That’s the reason why we hand-checked the difference between automatically generated and real pronunciation during the annotations. Observed differences were marked according to above described rule for orthographic transcription. When speech database is annotated this way, it allows us to generate lexicon of correct real pronunciations from orthographic transcription of recorded utterances.

This procedure seems to be quite efficient for generation of pronunciation variants of some words because the annotator can always compare automatically generated pronunciation by a tool (“*transc*” in our case) and real pronunciation uttered by speaker.

4.2. Lexicon structure

The information about word pronunciation is stored in the file which has very simple structure. It is simple text file where the information for single item is a block of lines separated by empty line. Each significant line begins with a keyword which resolve the kind of the information. The structure is similar to SAM label file format, used for annotation of SpeechDat family databases (Pollák and Černocký, 1999).

WRD: **word**
PRN: **pronun1;pronun3;pronun3**
OCW: 10
OCP: 3;5;2
SND: f11.wav,f12.wav,f13.wav;;f3.wav

WRD: **another_word**
..

The first two lines of the block for one item are mandatory and these two lines represents also the principle part of the lexicon. Other information are confidential. This structure allows easy work when some information is missing and also it is open to possible extensions of lexicon with other information. Finally, the file remains readable by simple view and also it can be processed by other general scripts.

4.3. Lexicon management tools

Updating pronunciation lexicon with new words is usually provided in several steps. However it requires some manual checks, some other steps may be done automatically.

At the early beginning, the first lexicon had to be always manually checked and corrected. Then the update of checked lexicon by new words can be slightly automatize, i.e. it may be done in following three steps:

1. automated comparison with checked lexicon (i.e. finding new items),
2. manual check of new items,
3. update of final lexicon by corrected new items.

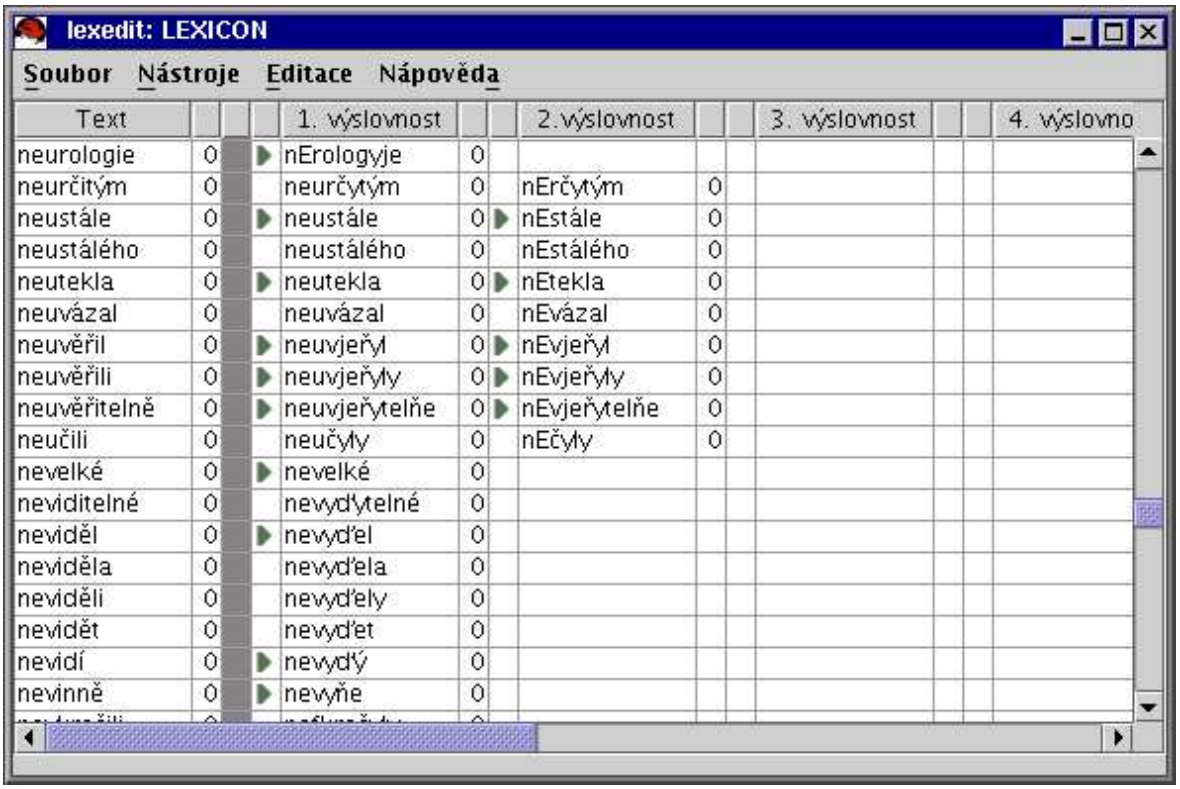

Figure 1: Lexicon management tool "*lexedit*" (pronunciation in IPA CTU)

This procedure is very efficient because after several updates the number of new words which must be checked is rapidly decreasing.

Concerning very simple structure of lexicon file, it can be processed by simple scripts for text processing. Our scripts representing particular tools are written in Perl. The most important tools for lexicon management are:

- *lexicon_differentiate.pl* - it finds new items or items with another unknown pronunciation,
- *lexicon_update.pl* - it updates final lexicon from corrected differential one, bad items in difference lexicon should be manually marked and these items are not involved to the final lexicon,
- *lexicon_count_occurence.pl* - computes the rates of occurrences above given corpus,
- *lexicon_find_word.pl*, *lexicon_find_pron.pl* - find given words (pronunciations) from lexicon in given source corpus,
- *lexicon_sampify.pl*, *lexicon_desampify.pl* - this pair makes conversion between IPA CTU and CZECH SPEECHDAT SAMPA.

The usage of above described tools is very simple. They can be processed on command-line basis and output is written to `stdout` stream. The redirection into specified file can be easily provided using operating system syntax.

4.4. User friendly interface

To increase user convenience, we present friendly interface which is written in Java programming language and which uses above described tools. The control is very intuitive and it is done by standard menus, hot keys, etc. Java programming language allows efficient implementation of structured lexicon management and it gives the support for visualisation and some additional functions as playing of associated acoustic files etc. The tools can be used independently on the platform, only standard Java virtual machine, runtime class libraries, and Java application launcher must be installed on the platform. These components are parts of Java Runtime Environment v1.4 (http://java.sun.com/j2se/1.4/download.html). The basic window of this tool with an example of one lexicon is on fig. 1.

The first version of lexicon management tool was realized under TCL/TK environment but especially complex editing and processing structured data was more complicated. That was the reason for final choice of Java Programming Language.

Very important property of our tool is the possibility of playing associated audio file which represents the utterance where the word with given pronunciation was spoken. It is very important especially during manual check of particular pronunciations. Items with associated audio file are marked with simple icon, see fig. 1.

5. Applications

5.1. Analysis of pronunciation variants of digits

As a by-product of this work we can present the rates of occurrence for pronunciation variants of digits $0 \div 9$ over Czech database of numerals (database FIXED2CS - "ČÍSLOVKY") which is available from ELRA (http://www.icp.grenet.fr/ELRA/home.html). The results

presented in tab. 2 are easily derived by computation of occurrences over given corpus by "*lexedit*" tool.

Digit	***Rate of variants***	*Ortho*
0	**nula - 1221 (≐ 100%)**	*nula*
1	**jedna - 1219 (≐ 100%)**	*jedna*
	jeden - 2	*jeden*
2	**dva - 1037 (≐ 85%)**	*dva*
	dvje - 186 (≐ 15%)	*dvě*
3	**tP\i - 1220 (≐ 100%)**	*tři*
4	**t_StiP\i - 965 (≐ 79%)**	*čtyři*
	t_Stiri - 47 (≐ 4%)	*čtyry*
	StiP\i - 137 (≐ 11%)	*štyři*
	Stiri - 72 (≐ 6%)	*štyry*
5	**pjet - 1224 (≐ 100%)**	*pět*
6	**Sest - 1222 (≐ 100%)**	*šest*
7	**sedm - 280 (≐ 23%)**	*sedm*
	sedum - 942 (≐ 77%)	*sedum*
8	**osm - 269 (≐ 22%)**	*osm*
	osum - 952 (≐ 78%)	*osum*
	vosm - 0 (0%)	*vosm*
	vosum - 1 (≐ 0%)	*vosum*
9	**devjet - 1220 (≐ 100%)**	*devět*

Table 2: Rates of pronunciation variants of digits $0 \div 9$ (pronunciation written in SAMPA)

5.2. Quality check of database annotation

Another application where we have used this tools was in annotation of large corpora. It seemed to be useful to create the pronunciation lexicon continuously by small steps after annotations of small blocks of data. Firstly, the check of smaller amount of new lexicon items is more pleasant. It brings consequently higher concentration on this work which yields to less amount of errors in the final lexicon. Secondly, many errors in orthographic transcription are easily detected which increase the quality of database annotation.

6. Conclusions

The most important part of this presentation can be summarized in following points:

- Two phonetic alphabets for Czech were described, IPA CTU and CZECH SPEECHDAT SAMPA.
- The updated version of tool "*transc*" for automated generation of phonetic transcription was described. Fixed rules of conversion are completed by extensible lexicon of irregularities. The usage of this lexicon increase the probability of correct pronunciation generation by this tool for new text.
- Lexicon management tool "*lexedit*" was created. It works with structured pronunciation lexicon file with possible additional information. It is written in Java programming language which allows user friendly editing or updating of given lexicon.

Acknowledgement

This paper is the part of the research supported by Czech Grant Agency as grant "Voice Technologies for Support of Information Society" with No. GACR-102/02/0124.

Also the work of Miloslav Brada, student of Czech Technical University in Prague, should be appreciated. He has created graphical user's interface for lexicon management tool.

7. References

J. Nouza, J. Psutka, and J. Uhlíř. 1997. Phonetic alphabet for speech recognition of Czech. *Radioengineering*, 6(4):16–20, December.

Z. Palková. 1984. *Fonetika a fonologie češtiny*. Univerzita Karlova, vydavatelství Karolinum.

P. Pollák and J. Černocký. 1999. Specification of speech database interchange format. Technical report, SpeechDat(E), August. Deliverable ED1.3, workpackage WP1.

J. Černocký, P. Pollák, and V. Hanžl. 2000. Czech recordings and annotations on CD's - Documentation on the Czech database and database access. Technical report, SpeechDat(E), November. Deliverable ED2.3.2, workpackage WP2.

http://noel.feld.cvut.cz/sampa/CZECH-SAMPA.html - Latest Czech SAMPA proposal.

http://www.icp.grenet.fr/ELRA/home.html - ELRA (European Languange Resources Association) home page.

http://www.phon.ucl.ac.uk/home/sampa/czech-uni.htm - Unofficial Czech SAMPA.