

Development of LottaPron
-
an automatic pronunciation generator

Charlotta Duit
lottad@stp.ling.uu.se

Master's thesis in Computational Linguistics
Språkteknologiprogrammet
(Language Engineering Program)
Uppsala University – Department of Linguistics and Philology

17th December 2004

Supervisors:
Bertil Lyberg, Uppsala University
Rich Schulman & Mathias Johansson, SpeechCraft

Abstract

This master's thesis has been conducted at SpeechCraft, a company that mainly works with the product Prego which is an automatic switchboard based upon speech recognition. Prego is developed with speech recognition services provided by Nuance Communications Inc. One of these services is called Autopron which is an automatic pronunciation generator and which supplies phonetic transcriptions from graphically written input.

However, the transcriptions generated by Autopron do not represent proper names well; especially not foreign proper names. Therefore the purpose of this thesis has been to develop a "new" Autopron based upon Swedish subjects' pronunciation variations of foreign names. The foreign languages that were selected for this thesis are Finnish and Serbo-Croatian. Ten subjects were chosen to record a set of Finnish and Serbo-Croatian names. Pronunciation rules were developed based upon the subjects' pronunciation variations. These rules were then used to create the LottaPron program implemented in Perl. The program that has been developed takes a foreign name as input and produces several different pronunciation variations as output. This automatic pronunciation generator is, in the long run, intended to improve the performance of the product Prego.

The LottaPron program has been evaluated with the Nuance Batchrec utility which tests the recognition performance of an application. The results from the Batchrec evaluation shows a slight improvement in recognition performance for the LottaPron program compared to the Autopron utility.

Contents

List of Figures	iii
Acknowledgements	iv
1 Introduction	1
1.1 Purpose.....	1
1.2 Outline of this thesis	2
2 Background	3
2.1 Nuance speech recognition	3
2.1.1 Introduction to the Nuance speech recognition process	3
2.1.2 Autopron.....	5
2.1.3 Batchrec.....	6
2.1.4 The Nuance phoneme set and the Computer Phonetic Alphabet	6
2.2 Xenophones.....	7
2.3 Finnish and Serbo-Croatian	7
2.3.1 Characteristics of Finnish.....	8
2.3.2 Characteristics of Serbo-Croatian.....	8
2.3.3 Comparison with Swedish phonetics.....	9
2.4 Generative grammar and phonology.....	10
2.4.1 Generative rules in phonology.....	11
2.4.2 Feeding and bleeding and linear rule ordering	12
2.4.3 Generative rules in Perl	13
3 Methods for gathering pronunciation variation	14
3.1 Gathering of survey material	14
3.2 The subjects	15
3.3 The first name collection	15
3.4 The second name collection.....	16
3.5 Transcribing the names	16
3.6 Results of the surveys	16
4 Implementation of the LottaPron program	18
4.1 From subject utterance to a complete rule	18
4.2 Exposition of the LottaPron program flow	19
4.3 The need for a specific file and rule ordering	21
4.4 Description of the rules and the code.....	21

4.5 Example run of the LottaPron program	23
5 Evaluation.....	25
5.1 Initial evaluation of the LottaPron program.....	25
5.2 Batchrec evaluation.....	25
5.2.1 Recapitulation of the Batchrec utility	26
5.3 Results from transcriptions produced by Autopron	28
5.4 Results from manually produced transcriptions	29
5.5 Results from transcriptions produced by LottaPron	30
5.6 Comparison of the results	31
5.6.1 Substitutions	31
5.6.2 Confidence scores.....	31
5.7 Discussion of the results	32
6 Concluding remarks and future developments.....	35
Bibliography	37
Appendices	
A. Swedish phoneme set.....	39
B. The first name collection	41
C. The second name collection	43

List of Figures

Figure 1 Nuance recognition process overview 3
Figure 3 Flow chart of the LottaPron program20
Figure 4 Table of name changes23-24

Acknowledgements

I would like to thank a few people for making this thesis possible. First of all I would like to thank everybody at SpeechCraft. I am especially grateful to my supervisors Rich Schulman for presenting the topic for this thesis and to Mathias Johansson for always finding the time to straightening out the problems.

Also, I would like to thank my supervisor at the Department of Linguistics and Philology, Bertil Lyberg, for the help with the formalities.

Furthermore I am eternally grateful to my ten trustworthy subjects who diligently read all the names as I asked them to. Without these recordings there would be no thesis.

Last, but not least, I want to thank my excellent Scottish proofreader, Brian McGill, for pointing out my flaws.

1 Introduction

The internationalized society we live in today has caused our languages to change in many different ways. The increasing influence from other cultures and languages has affected the Swedish language in many ways. One way this has influenced the Swedish language is the appearance of xenophones, i.e. foreign speech sounds in a language. These xenophones are often encountered in foreign names. This is an important and highly current question in Sweden today since many of us no longer are called Andersson. Not only our names are foreign; we also continue to borrow words from other languages.

The presence of xenophones are interesting from a technological standpoint since increasingly many of our devices are governed by our speech. In order to create speech recognition systems with high quality for the Swedish language these xenophones need to be included. This thesis is an attempt to further develop such a speech recognition system by integrating foreign speech sounds in the underlying language description of the system.

1.1 Purpose

This thesis has been conducted at SpeechCraft, a company that mainly works with the product Prego which is an automatic switchboard based upon speech recognition services provided by Nuance Communications Inc. When you place a call to the Prego system you simply say the name of the person you are seeking and you are automatically connected to that person without ever talking to a human telephone operator.

One component in the Nuance speech recognition system is called Autopron which automatically supplies phonetic transcriptions from graphically written input. However, these automatically generated transcriptions do not represent proper names well and particularly not foreign proper names. Therefore the purpose of this thesis is to develop a “new” version of the Nuance utility Autopron based upon Swedish subjects’ pronunciation variations of foreign names. To limit the thesis, only two foreign languages, Finnish and Serbo-Croatian, have been used as a basis for the gathering of the pronunciation variations. The “new” Autopron program, which for selfish reasons is called LottaPron throughout this thesis, is intended to improve the performance of the speech recognition for the product Prego.

1.2 Outline of this thesis

This thesis is divided into six different parts including chapter 1 which consists of this introduction. Chapter 2 supplies the basic backgrounds for all areas connected to the purpose of this thesis. The chapter initially gives a brief summary of the Nuance speech recognition process followed by a recapitulation of the xenophone notion. Chapter 2 also includes the basic knowledge in Finnish and Serbo-Croatian phonetics as well as the fundamentals of generative phonetics. Chapter 3 concerns the methods used to gather the pronunciation variations. The surveys and the subjects are accounted for in this chapter. In chapter 4 the program, LottaPron, is described. The process from a subject's pronunciation to the development of actual functional code is accounted for and the structure of the program is explained. Chapter 5 concerns the evaluation of the LottaPron program with the help of the Nuance evaluation utility Batchrec. The results from the evaluation are also discussed in this chapter. Finally, chapter 6 concludes and summarizes the thesis as well as discussing further developments of the LottaPron program.

2 Background

Nuance Communications Inc. provides solutions for different automatic telephony services and has developed speech recognition systems for a number of languages including Swedish. The following four sections will describe the Nuance speech recognition process and some of the Nuance system utilities of interest for this thesis. These Nuance sections are then followed by a background to the essential phonetic knowledge for this thesis.

2.1 Nuance speech recognition

2.1.1 Introduction to the Nuance speech recognition process

The Nuance Speech Recognition System is based on a complex client/server architecture which will not be explored deeper than necessary for this thesis. Therefore, only a brief survey of the most essential components will be accounted for in this section. As a starting point for this survey a flow chart over the Nuance speech recognition process (Figure 1) is shown.

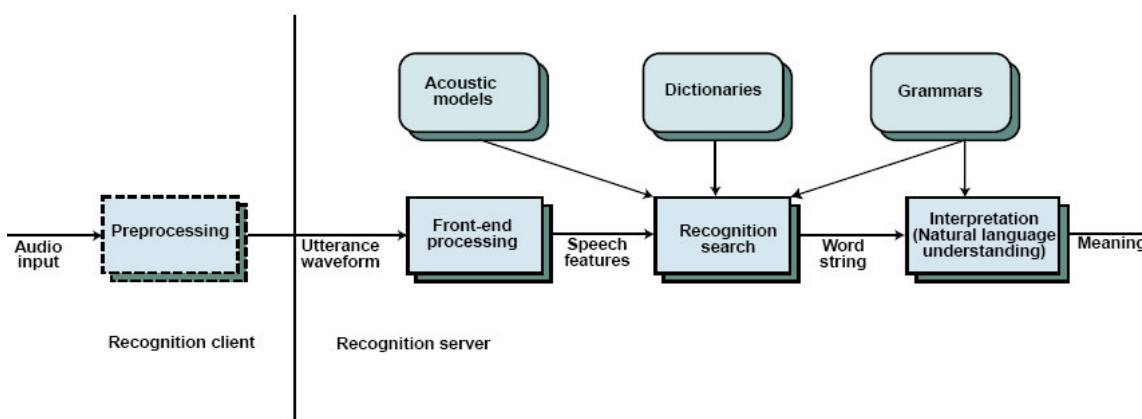


Figure 1 Nuance recognition process overview (Introduction to the Nuance System Version 8.0 1996-2001, p. 14)

Before the actual recognition process is performed by the **recognition server** the **preprocessing** phase is handled by the **recognition client**. In this phase, the speech signal undergoes echo cancellation and endpointing. The echo cancellation removes echo from the speech signal whereas endpointing is the process where background noise or silence is distinguished from the actual speech signal. The endpointing process also detects where the speech signal begins and ends (*Introduction to the Nuance System Version 8.0* 1996-2001, p. 14-15).

The input to the **front-end processing** phase consists of an utterance waveform. The front-end processing filters out some of the background noise and at the same time the front-end processing extracts a feature set from the original audio sample. The recognition processing is then performed on the feature set and not on the audio sample (*Introduction to the Nuance System Version 8.0* 1996-2001, p. 15).

The recognition server performs the speech recognition in the **recognition search** phase with the help of three different components: *Acoustic models*, *Dictionary files* and *Recognition grammars* (*Introduction to the Nuance System Version 8.0* 1996-2001, p. 14).

The dictionary file contains a phonetic transcription for every entered word in the grammar.

For instance the name `Emina` can have the following transcriptions:

`emina` `e m i n a`

`emina` `e m I n a`

`emina` `E m i n a`

`emina` `E m I n a`

Nuance provides extensive dictionaries for every language that they support and when developing an application, if necessary, you can chose to add extra dictionaries containing special entries such as uncommon names (*Introduction to the Nuance System Version 8.0* 1996-2001, p. 14).

The recognition grammar defines all the accepted lexical entries and at the same time it defines the corresponding interpretation for these lexical entries. That is, the name `Emina` is listed in the grammar which enables it to be recognized (*Introduction to the Nuance System Version 8.0* 1996-2001, p. 14).

The acoustic model contains Hidden Markov Models (HMM) and they are used for the phonetic recognition (*Introduction to the Nuance System Version 8.0* 1996-2001, p. 17). (HMMs are complex statistical models which will not be explained further in this thesis, only their function in the recognition process will be briefly described.) The HMMs interpret the meaning of the sampled speech signal by mapping the signal to a series of phonetic units. At the same time HMMs are also used to map the phonetic sequence to the word series that

represents the transcription (*Application Developer's Guide Version 8.0* 1996-2001, p. 204-205).

During the recognition search phase, the recognition process uses all of the three components accounted for above to analyze the speech features and to be able to produce a transcription of the original utterance. The acoustic model is used to recognize the individual phonemes. The dictionary, combined with the acoustic model, maps the phoneme sequences to words. And finally the grammar defines the set of word sequences that can be recognized (*Introduction to the Nuance System Version 8.0* 1996-2001, p. 16-17).

The final phase of the Nuance recognition process is the **interpretation phase** which enables the application to produce a suitable response to the original utterance. This phase in the process also uses the grammar to be able to produce a response since only utterances that are listed in the grammar can be successfully recognized. The input to the interpretation phase is the recognized utterance and the output is an interpretation of the meaning of the utterance. The interpretation phase then uses the interpreted meaning to produce a response in the form of, for instance, playing a prompt (*Introduction to the Nuance System Version 8.0* 1996-2001, p. 19).

2.1.2 Autopron

When developing a new application, an application dependent grammar has to be produced. For each word in the grammar there has to be a defined pronunciation in the dictionary as described above in section 2.1.1. During the compilation of the grammar, a search is done in the standard dictionary files (which are provided by Nuance) and in the possible application dependent dictionary for the correct pronunciation of the word. If the system fails to find a pronunciation for the word in the available dictionaries an error is displayed and a file is created containing a list of the words that are missing pronunciations (*Grammar Developer's Guide Version 8.0* 1996-2001, p. 105).

To create pronunciations for these missing words there are two options. The transcriptions can be added manually to the dictionary or the Autopron function can be used (*Grammar Developer's Guide Version 8.0* 1996-2001, p. 106). Autopron is the automatic pronunciation generator function in the Nuance System. The Autopron function creates pronunciation for the

missing words if the command *auto_pron* is specified during grammar compilation. Autopron produces a maximum of ten transcriptions for every input word. The pronunciations that Autopron generates are automatically written to a file with the file extension *.autopron* and the system can search for the pronunciations in this file as well as in the available dictionaries. Of course these automatically generated pronunciations can then also be added to the available dictionaries (*Grammar Developer's Guide Version 8.0* 1996-2001, p. 107-108).

However, the automatic pronunciations generated by Autopron are not as accurate as manually created transcriptions (*Grammar Developer's Guide Version 8.0* 1996-2001, p 133). Specifically when concerning the pronunciation of proper names the connection between spelling and pronunciation appears to be weaker which makes it more difficult for Autopron to generate the proper pronunciations. In particular, foreign names are often not pronounced as would be expected but in a completely different way.

The Autopron function is used twice during the work with this thesis. Autopron was first used to choose which languages to concentrate on in the development of the LottaPron program (See section 3.1). Autopron was also used in the evaluation process to generate data to compare with the data generated by the LottaPron program (See chapter 5).

2.1.3 Batchrec

The Nuance System contains an evaluation program called Batchrec. Batchrec can be used to test the recognition performance of an application. Batchrec requires pre-recorded audio files containing the words that are to be tested. If utterance transcriptions and/or natural language interpretations are provided, the Batchrec utility then evaluates the recognition accuracy scores for each of the recorded utterances (*Grammar Developer's Guide Version 8.0* 1996-2001, p. 119). Batchrec is used in the evaluation phase of this thesis and will be explained further in section 5.2.1.

2.1.4 The Nuance phoneme set and the Computer Phonetic Alphabet

In the Nuance System, a set of symbols are used to represent the different phonemes of a language. (The phoneme set for Swedish can be viewed in appendix A). The Computer Phonetic Alphabet (CPA) is used to specify the Nuance phoneme set. CPA is a phonetic

alphabet defined by IPA (International Phonetic Alphabet) which is adapted to work using the computer keyboard (*Grammar Developer's Guide Version 8.0* 1996-2001, p.133-134).

However, CPA is not as detailed as IPA and certain specifics are left out. For example, in the Swedish phoneme set there are no distinctions between the different allophones for *ö* before *r* and *ö* before any other letter. Furthermore, retroflexion and nasalisation sounds are not part of the Swedish phoneme set.

Since the application that was developed for this thesis concerned Swedish people's pronunciation of foreign proper names, the subjects sometimes produced phonemes that were not a part of the Swedish phoneme set as defined by the Nuance System. To be able to transcribe these names properly, two phonemes that were produced often by the subjects were added to the phoneme set when developing the application. These phonemes are defined as *a* : (long front *a*) and *s* : (voiced velar fricative).

2.2 Xenophones

Due to the growing influence from other languages and countries, the number of phones in Swedish has increased. A xenophone is a foreign phone in a language. That is, the xenophone is a part of the phone sets of the speakers of a language but it is not considered to be a phoneme or an allophone in the language. An example of a xenophone in Swedish is the voiceless dental fricative [θ] (as in 'thank'). The voiceless dental fricative is not a phoneme in Swedish but is produced by nearly half of the Swedish population when English names or words are encountered in a Swedish text. This xenophonia phenomenon will have consequences within speech recognition systems which will need to increase the language description in order to work properly (Eklund & Lindström 2001).

2.3 Finnish and Serbo-Croatian

The two languages that were chosen as basis for this thesis are Finnish and Serbo-Croatian. Why these two languages were picked will be discussed in section 3.1 below. In this section, a presentation is given of the phonetics of the both languages.

2.3.1 Characteristics of Finnish

Finnish is a member of the Finno-Ugric language family and is therefore not closely related to Swedish which is a member of the Indo-European language family (*Ethnologue language family index*). However, the Finnish pronunciation does not differ greatly from the Swedish pronunciation. There are 8 vowels in Finnish - i, e, ä, y, ö, u, o, a - and 13 consonants - p, t, k, d, s, l, r, v, j, h, m, n, ng (*Nationalencyklopedin* 1991, p. 298). The Finnish pronunciation is highly consistent. The primary stress is always on the first syllable and each graph can only be pronounced in one way. Long vowels are double signed and long consonants are equally double signed (Maarit, Jukka). A single vowel or consonant is always short. Thus, the visible differences between Swedish and Finnish are few if you disregard the double signed vowels and should therefore not create too much difficulty for a Swede to pronounce.

2.3.2 Characteristics of Serbo-Croatian

Serbo-Croatian belongs to the Indo-European language family and more specifically to the South Slavic branch (*Ethnologue language family index*). Today the denomination 'Serbo-Croatian' is abolished and Serbian and Croatian are considered independent languages (*Nationalencyklopedin* 1995, p. 377). But the differences between the two languages are very few and therefore the languages have been considered in this thesis as one language. Serbian is written with the Cyrillic alphabet while Croatian is written with the Latin alphabet. Serbo-Croatian also applies spelling that corresponds to the pronunciation (*Nationalencyklopedin* 1995, p. 376) but the connection to Swedish spelling is much weaker than between Finnish and Swedish. Serbo-Croatian has several phones that do not exist in Swedish and thus the spelling in many cases is very different from Swedish spelling. There are 5 vowels - a, e, i, o, u, - and 25 consonants - p, b, t, d, k, g, m, n, nj, r, l, lj, f, v, s, z, š, ž, j, h, ģ, č, đ, đ - in Serbo-Croatian (Dahlstedt 1971, p. 20 and 32). The following are some examples of the pronunciations of the, for Swedish, foreign consonants:

č - c as in cello

ć - tch as in latch

đ - g as in germ

dž - dg as in edge

h - ch (gutteral kh) as in loch
j - y as in yes
lj - ly as in halyard
nj - ny as in canyon
r - slightly rolled; when used as a vowel as er in pert
š - s as in sugar
ž - z as in azure

(Serbo-Croatian Pronunciation Guide)

2.3.3 Comparison with Swedish phonetics

Finnish has, as mentioned above, 8 vowels: i, e, ä, y, ö, u, o, a. U is pronounced as the Swedish o as in /bo/ and o is pronounced as the Swedish å. In other words, the Swedish u is missing from the Finnish vowel system (*Nationalencyklopedin* 1991, p. 298). A problem that can arise from this vowel system for a Swedish speaker is probably the double signed vowels mentioned above, especially if there is a double signed consonant following the double signed vowel. This, in Swedish, indicates that the vowel should be short. Also it is possible that u and o is pronounced according to Swedish rules.

The 13 Finnish consonants - p, t, k, d, s, l, r, v, j, h, m, n, ng - are not very different from their Swedish correlates. The consonants that are missing in Finnish are three fricatives (f, sj, and tj) and two stops (b, g) (Dahlstedt 1971, p.32). This consonant system will not cause any problems for a Swedish speaker since the consonants are all familiar.

Serbo-Croatian has 5 vowels as mentioned above. Among these there are no front rounded ones as Swedish y and ö. Furthermore, there are no allophonic variation among the vowels as is the case with Swedish ä and ö when placed before an r (Dahlstedt 1971, p.29). The Serbo-Croatian's, compared to Swedish, decreased vowel system should not bring any problems for Swedish speakers since all the Serbo-Croatian vowels exist in Swedish.

Serbo-Croatian has, as already mentioned above, 25 consonants. 9 of these are not present in the Swedish phonetic system. One of the 9 is a nasal consonant which is foreign for Swedish

speakers: n.j. Also a liquid, l.j, is foreign to Swedish speakers. The rest of the 9 are either fricatives: z, ž or affricates: c, č, č̣, ć, đ (Dahlstedt 1971, p.32). These foreign consonants should be difficult for Swedish speakers to pronounce. For example c is mostly pronounced in Swedish as s or k, depending on the context, not as an affricate and z is mostly pronounced as s in Swedish. The different variants of c: ć, c̣, č will probably, to a large extent, all be pronounced as a c would be in Swedish. The diacritic signs that distinguish the different c's are seldom printed in Swedish. The nasal consonants will probably not cause problems for a Swedish speaker since they are pronounced exactly as in Swedish only they are not considered as phonemes in Swedish.

2.4 Generative grammar and phonology

The use of formal languages in linguistics to model natural languages is called generative grammar (Jurafsky & Martin 2000, p. 331). The generative grammar theory was introduced by Noam Chomsky and Morris Halle in *Syntactic Structures* 1957. Generative grammar is focused on the nature of linguistic knowledge. The basic questions in generative grammar concern what it means to know a language, how you learn a language and if there are any universal properties of languages which could be categorized as an Universal Grammar (Katamba 1989, p.X).

A generative grammar consists of a collection of rules which indicate how all grammatically correct sentences in a language are produced. The grammar is therefore called generative since the language is defined by the set of possible sentences that are *generated* by the grammar (Elert 1970, p. 20).

Different components together constitute the generative grammar. The most essential component is the syntax but also among other a phonological component exists (Elert 1970, p. 20-21). These components all play different parts in generating the utterances of the language (McAllister 1998, p. 168).

The Generative Phonology theory was developed by Chomsky and Halle in *The Sound Pattern of English* in 1968. The purpose of the generative phonology is to relate the product

of the syntactic component, which is called the surface structure, to the phonetic structure with the help of the phonetic rules (McAllister 1998, p. 168).

2.4.1 Generative rules in phonology

During the process of formulating pronunciation rules, the basic formal conventions of generative phonology have been used. A basic formal rule is often portrayed like this:

$$a \rightarrow A / _C$$

This stands for ‘a becoming A if found before a consonant’ (Katamba 1989, p. 118-119). This is an example of a context dependent rule since it will only apply if the correct context is found. There are also context free rules that have no specified environment and will thus work in any surrounding.

There are different kinds of operations that can be carried out with the help of a generative rule. The rule already mentioned above is an example of a specification rule. There are also deletion rules:

$$A \rightarrow \emptyset / x_y$$

This rule reads ‘A becomes nothing (\emptyset) in the context between x and y’. Correspondingly there are also addition rules:

$$\emptyset \rightarrow A / x_y$$

where ‘nothing becomes A in the context between x and y’.

There are of course several further different rules in generative phonology but those mentioned above are the most basic ones. Also, a set of different symbols are available to further modify the rules. One of the most basic of these symbols is the symbol for morpheme boundary: # which is often used when building rules¹ (McAllister 1998, p. 177-178).

¹ Notice that there are different notational conventions for the morpheme boundary. Sometimes # can symbolize a word boundary while + stands for the morpheme boundary.

2.4.2 Feeding and bleeding and linear rule ordering

Problems concerning the ordering of the generative rules can arise. The rules can interfere with each other causing faulty or no output at all. It is therefore important to consider and determine the proper ordering of the rules. Chomsky and Halle projected in *Sound Pattern of English* that rules should be linearly ordered. When rules are linearly ordered each rule has its own place and each rule can not be applied more than once. That is, an order is established where each rule is ascribed to a place and then the rules are carried out in order. So if a rule is number five on a list of rules, all rules from one to four have to be carried out before rule number five and all rules with numbers higher than five have to be carried out afterwards (Katamba 1989, p. 122-123).

The ordering relation between different rules can be described on the basis of the potential effect (positive or negative) that the application of one rule has on the application of another rule. This effect can be described to be either *feeding* or *bleeding*. Two rules are in a *feeding* relation if the application of rule A produces new input to rule B. Correspondingly, two rules are in a *bleeding* relation if the application of rule A eliminates input to rule B (Kenstowicz 1994, p. 94).

An example of the importance of rule ordering is taken from Katamba's *An Introduction to Phonology* (1989) p. 122 and concerns the French word *an* [ã]. This word is both subjected to the rule that deletes final consonants that are not followed by a vowel and to the nasalisation rule. It is assumed that the underlying representation is /an/ and the two rules are then applied in different orders to see which order is the correct one.

1. Final consonant deletion rule:

[+cons] → ∅ / ___ {C or #}

2. Vowel nasalisation rule:

[-cons] → [+nasal] / ___ [+nasal] {C or #}

In the first try, rule 1 is first applied to /an/. This results in the deletion of the final nasal consonant n. When rule 2 then is applied, there is no longer a nasal to trigger off the nasalisation process on the vowel. The output is then merely an [a] with no nasal marking. If the rules are applied the other way around, rule 2 then rule 1, the vowel first gets nasalised by the n and then the n is deleted by the final consonant deletion rule leaving the output [ã].

Rule ordering is therefore, as shown in the example above, necessary to decrease the possible output you get from phonological rules and to make sure that it is a correct output (Katamba 1989, p. 132).

2.4.3 Generative rules in Perl

There are several tools for manipulating text strings in the programming language Perl which has been used for the development of the LottaPron program (Hammond 2003, p. 94). The function that most frequently has been used to develop the LottaPron program is `s///`. In principle, this function is the same as a generative rule. The `s///` function is used to replace a string in a defined context with another string. To illustrate this, a rule from the program is shown:

```
$longvoka =~s/[a] ([bcdfghjklmnpqrstvwxz])/A$1/g;
```

This also reads “a becoming A if found before a consonant” as in the example above in section 2.4.1. The details of the rules will be explained later in the thesis (see section 4.4).

3 Methods for gathering pronunciation variation

In this chapter, the work methods for gathering and limiting the data for the developing of the LottaPron program are described. The subjects and the collection of their pronunciation variations are accounted for here as well as the reasoning behind the choice of languages for the survey.

3.1 Gathering of survey material

To be able to write pronunciation rules for Swedish pronunciation of foreign names, two surveys had to be conducted to gather pronunciation variation. 10 subjects were collected for the purpose of recording their pronunciations of two sets of foreign names. The material for these surveys was selected from one of SpeechCraft's name databases. There were approximately 2000 names to search through. In the first round of selections, every name that for some reason did not feel Swedish was picked out. These selections were merely based upon an intuitive knowledge as a native Swedish speaker. The selected names were then grouped according to possible origin. In cases where the origin was not clear (which were most cases except the Finnish names), the name was googled. The reasoning behind this action was that the more hits from one country, the more likely this name has this origin. For some names, homepages was even encountered that specifically described the origin of the name. After this rather unscientific classification, three major groups of names were collected: Finnish, Serbo-Croatian and Middle Eastern names.

The Nuance utility Autopron was run on the different name groups to determine whether or not Autopron would produce approvable results for the different groups. After running the Autopron on all of the names, it was finally decided that the two language groups to be investigated further were Finnish and Serbo-Croatian. The Finnish names were selected when the results from the Autopron test showed that Finnish did not do well. The Middle Eastern names were ruled out simply because there were too few of them with which to conduct a proper investigation. Also the Serbo-Croatian names were too few in number (although not as few as the Middle Eastern names) and had to be supplemented with names found on the Internet. These names were taken from the homepage of the Serbian Confederation in Sweden (*Serbernas Riksförbund i Sverige*).

3.2 The subjects

10 subjects were collected for the experiment; 4 male and 6 female from different age groups spanning from 20 to 50 years of age. These subjects had no prior specific linguistic knowledge about the foreign languages in the study. Nor did they have any insight into the purpose of the survey. It is important that the subjects have no education in, or otherwise close connection to, the two languages in the survey since it is the Swedish speakers' pronunciation of foreign names that is under investigation. It is also important to prevent the subjects from knowing the purpose of the survey in order to hinder them from enunciating or talking extra slowly. The results that are interesting are not the correct pronunciations but the wrong ones since the problems with speech recognition lie within the variations. The subjects were sent an email with an attached instruction along with the name list (see appendix B). The subjects used the telephone to make their recordings and for this purpose a program was constructed which recorded the subjects over the phone line. This was done with Nuance applications that are specially developed for recordings over the phone.

3.3 The first name collection

All together, 80 names were selected for the survey. 40 names from each language. 20 of the Finnish and 20 of the Serbo-Croatian were set aside for the evaluation recording. The remaining 20 Finnish and 20 Serbo-Croatian names were mixed with 80 other names randomly taken from the company's databases. The results from this recording were used for developing the different pronunciation rules in the LottaPron program. In the surveys for both of the recordings, the names were grouped together as a first name and a last name. This means that the subjects in the two recording sessions read 40 units of names, each unit containing a first name followed by a last name. Half of these 40 units were mere padding which leaves 20 units containing the names of interest for the survey. The name units selected as padding were semi foreign; that is, they had either a foreign first name or a foreign last name. This mixing was done to prevent the subjects from figuring out the focus on the Finnish and the Serbo-Croatian names. The reason that semi foreign names were picked for the mixing was to give the subjects some breathing room during the reading since it can be a real tongue-twister to read many foreign names in a row. Also strictly Swedish names could not be picked to fill out the list since that would give the purpose of the survey away completely.

3.4 The second name collection

The second name recording was conducted exactly as the first one. The purpose of this survey was to collect pronunciations to test the LottaPron program with. The same subjects were used (apart from one female subject who was unable to participate) and the same amount of names were sent out to the subjects. That is 9 subjects and 40 names. The names used for this second recording can be viewed in appendix C. This survey was conducted 13 weeks from the first one. This was done to prevent the results from the second survey from influencing the development of the pronunciation rules. Otherwise it would be impossible to see if the pronunciation rules are general enough to work on a larger set of Finnish and Serbo-Croatian names than the set used for the development of the LottaPron program².

3.5 Transcribing the names

When the recordings were finished, the Finnish and the Serbo-Croatian names were transcribed with the Nuance phoneme set in CPA. The transcribing was done in CPA to be able to use the Nuance tools in the evaluation of the program. CPA was also used since an ascii-adapted phonetic alphabet is the easiest to use when developing a program that is to produce phonetic output.

The transcribing of the names was done as a basis for the developing of the pronunciation rules. A rule in the program consists of a letter (or several letters) that is transformed into a phoneme which is represented by a sign in CPA. The CPA signs will eventually be the output from the LottaPron program when all the letters in a name have been transformed into CPA.

3.6 Results of the surveys

Finnish seems to be easier than Serbo-Croatian for Swedish speakers to pronounce. The Serbo-Croatian names suffer from more stuttering, hesitation and pronunciation far from the original spelling than the Finnish names suffer from. Finnish names also generate fewer numbers of variants. The subjects have a harder time getting the Serbo-Croatian names right at the first try. The result is a larger number of stutterings and “dyslectic” variants and a larger number of variants.

² Observe that the rules will only work properly on Finnish and Serbo-Croatian names since the development of the rules were merely based on these two languages.

In the first recording, the subjects produced 69 different Finnish variants from the 20 Finnish names and 125 Serbo-Croatian variants from the 20 Serbo-Croatian names. In the second recording, 63 different Finnish and 99 Serbo-Croatian variants were produced. These numbers indicate that the subjects do not have as unambiguous an understanding of the Serbo-Croatian pronunciation as of the Finnish pronunciation. This may depend on the Serbo-Croatian spelling that is more foreign to Swedish people and thus causes more problems than the Finnish spelling does. Swedish people are also more used to the Finnish language since it is a neighboring country.

4 Implementation of the LottaPron program

The program that has been developed for this thesis takes a name in plain text as input and generates all possible phonetic pronunciations of the name as output. The output is presented in CPA (Computer Phonetic Alphabet) and the purpose of the program is to be used instead of, or as a supplement to, the Nuance utility Autopron. The following sections describe the development of the LottaPron program.

4.1 From subject utterance to a complete rule

The pronunciation rules that have been developed for this program are based upon the first recording of the subjects. Almost all of the variations that the subjects made are used as a basis for different pronunciation rules. Some of the subject variations were not appropriate for building rules upon since they were very far from the original spelling. Therefore, these variations were not considered when the rules were developed. As an example, the name *Radojčić* was pronounced by one subject as */radObjEk/*. If this pronunciation was to be the basis of a rule for i.e. names ending in *-jčić* it would generate several variants that most people probably never would produce. Therefore “mispronunciations” like these were not taken into account when the rules were developed. It is, of course, difficult to decide when a name is “mispronounced” enough not to be considered but the exclusion has taken place when it is decided that the “mispronounced” variation would cause damage to other more plausible variants.

The pronunciation rules are developed from graph to phone. That is, the input to the LottaPron program is a graph and the output will be a phone. When a name is sent in to the system it is represented by graphs (*erkki*) and the different pronunciation variants produced by the system will then be represented by CPA (*erki, erkI, Erki, ErkI*).

All the variants that are finally generated by the program are not always variants that have been produced by the subjects. This arises from two different reasons. Firstly, pronunciation variants that were considered to be missing have been added. Secondly, since a rule affects all names that share the same spelling in some way, the change will occur in all names that match the rule even if the subjects only produced the variation in one name. For instance the name

Kataja was pronounced in three different ways by the subjects: /kataja/, /kAtaja/ and /katAja/ but the program generates 9 different versions:

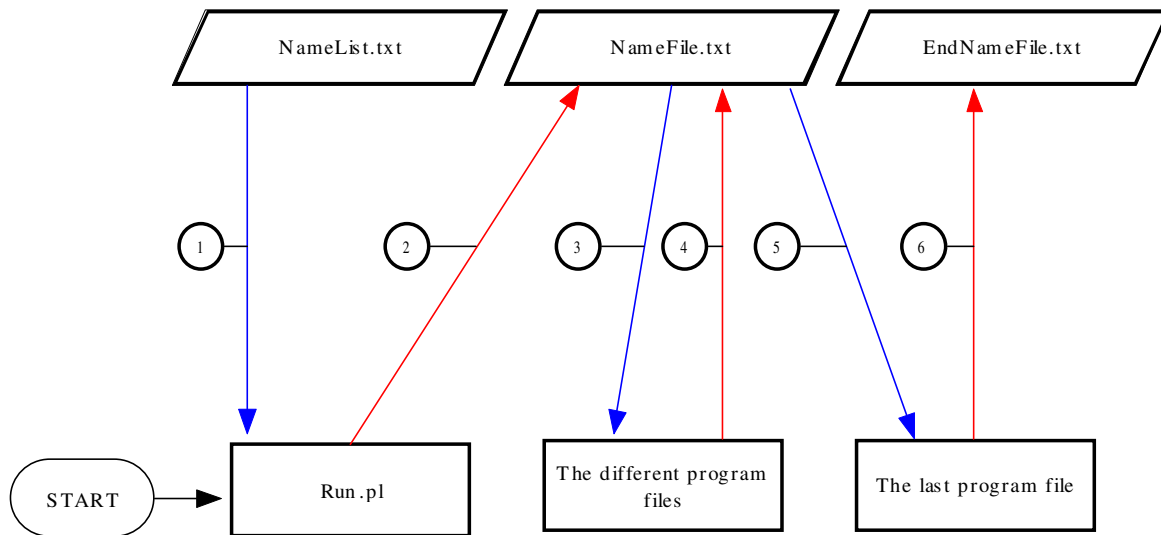
```
kAtAja
kAta:ja
kAtaja
katAja
kata:ja
kataja
ka:tAja
ka:ta:ja
ka:taja
```

These are all feasible variants but they were not all produced by the subjects. The program generates all these variants since other names with the same structure i.e. either an a on second place or an a on fourth place or an a in both positions have been pronounced by subjects in this way. And since the rules match similar structures, all 9 different versions of Kataja are generated.

For some isolated names, the subjects have produced variations that were not general enough upon which to build rules that would be applicable to the rest of the names. For these names, special rules were constructed that will only work on the name in question. These rules were also put in separate files to prevent them from interfering with other rules.

4.2 Exposition of the LottaPron program flow

This program has been developed with the programming language Perl which is well suited for working with string manipulation (Hammond 2003, p.94). The program that has been developed consists of 39 different files. Three of these files are text files while the rest are Perl files. The 36 Perl files all contain different pronunciation rules while the text files are used for temporary storage during the program process and for final output storage at the end of the process. Below, a flow chart of the LottaPron program is shown:



- 1) Run.pl reads a name at a time from NameList.txt
- 2) Run.pl writes a name at a time to NameFile.txt
- 3) The different program files each reads a name at a time from NameFile.txt
- 4) The different program files each writes the different name variants to NameFile.txt
- 5) The last program file reads a name variant at a time from NameFile.txt
- 6) Finally the last program file appends all the variants to EndNameFile.txt

Figure 3 Flow chart of the LottaPron program

All the names that are to be processed are placed in the first text file, NameList.txt. The second text file, NameFile.txt, is a temporary storage file that changes content throughout the program process. The final text file, EndNameFile.txt, stores the complete output from the program.

The program is started with the file Run.pl which contains commands to start the process. The file Run.pl also reads the names in NameList.txt and sends them one at a time to the file NameFile.txt. All the other Perl files will then, in order, process whatever name that is present in NameFile.txt.

The first Perl file that contains rules then reads the name that is present in NameFile.txt and, if the rules in the file are applicable to the name in question, the rules are carried out. The name, now changed by the rules in the first Perl file, is then again stored in NameFile.txt for the next Perl rule file to read and apply its rules on. This line of action is then repeated until the last Perl file is reached. In the last Perl file, all the variants that have been generated throughout the program are stored in the EndNameFile.txt.

This procedure is repeated until there are no more names to read in `NameFile.txt`. The loop in `Run.pl` that reads the names in `NameFile.txt` is then exited and the program is finished.

4.3 The need for a specific file and rule ordering

All the Perl files have to be placed in a special order. This has to be done to prevent the rules from interfering with each other according to the feeding and bleeding discussion above (see section 2.4.2). For example, the rules concerning the double signed vowels in Finnish have to be carried out before the single signed vowel rules. This has to be done since the double signed vowels have to be reduced to a single character first. Otherwise the following vowel rules would work on any found vowel creating unwanted variants like `maArIt` or `mAArIt`.

Another situation that arises due to the feeding and bleeding problem is the need for placeholders. Some graphs are the same as the CPA signs and, when changed for the CPA, the change is not visible. But the change needs to be visible since no other rules should change the sign later on. This problem calls for placeholders. A placeholder is a sign that is not a part of the Latin alphabet or the CPA alphabet and is used whenever a change needs to be visible in order *not* to get changed later on in the process.

4.4 Description of the rules and the code

The purpose of the system is to generate a varying number of pronunciation variants of the input name in question. This means that the most important feature of this system is the ability to temporarily store all the different variants of the current name throughout the complete process of the program. As soon as a new pronunciation variant has been created, it has to be stored somewhere where the following rules can continue to affect all of the different variants. The storage of the different variants has been solved using text files for temporary storage between the different Perl rule files and with arrays in each of the Perl rule files.

Basically all the rules in the program are written in the same way; a letter is exchanged for another sign. This is easily done in Perl with the function `s///`. For example, to exchange an upper case `J` for a lower case `j` in a string you write `$variable=~s/J/j/g`; and all

instances of J 's in whatever string that is present in `$variable` is exchanged for `j`. The flag `g` makes sure that all instances of J , and not just the first encountered J , is changed for a `j` (Hammond 2003, p. 94).

Each file in the program starts by reading the variants that are present in `NameFile.txt`. These variants are stored in an array, `@namearray`, in the beginning of each Perl file. From this `@namearray` the current Perl file reads a variant at a time. Each name variant in `@namearray` is thus one at a time exposed to the rules in the Perl file.

Every rule in the different Perl files works in the same way. The variable which contains the current name is renamed for every new rule. The rule is then applied to this temporary variable. When the rule has been applied to the temporary variable, a check is run to see if the temporary variable is identical to the original variable. If they are not identical, the rule has matched the name and a change has taken place. The content of the temporary variable is then saved instead of the original name in a second array, `@namearray2`. The next rule in the Perl file is then carried out in the same manner but this time the rule is also applied to each position in `@namearray2`. This is done on every rule since there can be variants saved in the array that also should be affected by this particular rule. As a precaution, this is done on every rule to be sure that all variants that are present in `@namearray2` always are exposed to every existing rule. Below is an example of a rule:

```
$variableName = $name; #Renames $name to current rule variable
$variableName =~/J/j/g; #Runs the rule on the content of the variable
for (each position in the array)
    $variableName =~/J/j/g; #Runs the rule on the entire array
if ($variableName is not equal to $name)
    push $variableName on to the array; #Pushes the
    variable onto the array
```

At the end of every file, a check is done of `@namearray2` that removes any duplicates of the variants if any such have been generated during the process. After this check, all the unique variants in `@namearray2` are written to `NameFile.txt` to be processed by the next Perl file in the system. If the `@namearray2` is empty and thus no changes took place in

this rule file, the original name is written to NameFile.txt instead and will then be processed by the next Perl rule file.

4.5 Example run of the LottaPron program

In this section, an exposition of how a name goes through all the changes in the different Perl rule files will follow. The name Dragan generates 9 different variations:

```
dra:gAn
dra:ga:n
dra:gan
dragAn
draga:n
dragan
drAgAn
drAga:n
drAgan
```

Name	File
Dragan	Run.pl
Dragan	Fil01.pl
dr@:gan dr@gan drAgan	Fil12.pl
dr@:gAn dr@:g@:n dr@:g@n dr@gAn dr@g@:n dr@g@n drAgAn drAg@:n drag@n	Fil16.pl

Dra:gAn	Fil26.pl
Dr:ga:n	
Dra:gan	
DragAn	
Draga:n	
Dragan	
DrAgAn	
DrAga:n	
DrAgan	
All variants are written to EndNameFile.txt	

Figure 4 Table of name changes

The first file, `Run.pl`, reads the name from the file `NameList.txt` and then sends the name to `NameFile.txt`. In `Fil01.pl`, all upper case is switched to lower case and `Dragan` becomes `dragan`. Then nothing happens until `Fil12.pl`, where the name encounters rules for vowels in the beginning of a name and the rules for initial a's are carried out. An a in the beginning of a name can become a, a: and A. There are now three variants of the name: `dr@:gan` `dr@gan` and `drAgan`. The a's have been exchanged for the placeholders for lower case a's in two instances. The next file that affects the variants is `Fil16.pl`. This file contains rules regarding final vowels. Also the final a's can become a, a: and A. This means that the three variants are increased to nine since, for each variant of initial vowels, there are three variants of final vowels. The last file that influences the name variants is `Fil26.pl`. This file contains a rule for converting the place holder for a, @, into a again. This leaves nine variants of the name `Dragan` in CPA which are written to the final storing file `EndNameFile.txt` in the last file of the program, `Fil35.pl`.

5 Evaluation

The evaluation phase consists of two separate evaluations. First, an evaluation of the LottaPron program has been conducted which only tests the ability of the LottaPron application. Then, the Batchrec utility has been used to evaluate the performance of the LottaPron application in comparison with the Autopron utility.

5.1 Initial evaluation of the LottaPron program

When all the names from the first collection worked sufficiently in the program, i.e. when the names generated all the pronunciation variations that were made by the subjects, the program was considered to be finished. To test the program, the names that were recorded in the second collection were run through the program. The results from this evaluation run are as follows: 24 out of the 40 names worked perfectly, 3 names did not work at all, and the remaining 13 names suffered from partial problems, that is the rules did not generate all the different variations that the subjects had produced. The variations for these 13 names were correct but just not exhaustive.

Some additional work was then done on the rules in the LottaPron program in order to make also the evaluation names work properly when run through the program. It is this final version of LottaPron that is evaluated with the Batchrec utility as described below in the next section.

5.2 Batchrec evaluation

When the LottaPron program was considered to be finished, the Batchrec evaluation phase was begun. To evaluate the performance of the application, the Nuance Batchrec utility was used. Before the actual evaluation could start, it was discovered that all the different pronunciation variations that contained any of the added phonemes *a:* or *s:* had to be removed. Unfortunately, Nuance does not support these phonemes and they had to be eliminated. All the results from the Batchrec run thus do not include any of the unsupported phonemes.

The names that were used for the evaluation Batchrec run are taken from the second name gathering. These are the names that were gathered solely for the purpose of evaluation of the application.

Also, it was discovered that some of the evaluation names unfortunately already existed in the standard Nuance dictionary. This meant that these names would be recognized without the pronunciation variations produced by the LottaPron and the recognition accuracy could therefore not be measured for these names. Since it is impossible to completely disconnect the standard dictionary and since the “existing” names sometimes were grouped together with a “non existing” name in the audio files, it was decided that all names would be tested with Batchrec anyway. Altogether there were 22 separate names left that were not in the standard Nuance dictionary and that would generate recognition accuracy with Batchrec. The results from the Batchrec runs will therefore have to be seen more as a pointer than actual statistics of the performance accuracy. However, the Batchrec runs make it possible, as will be shown later on in the coming sections, to check the recognition accuracy for every entered audio file. This enables introspection into the performance of the transcriptions of Autopron and LottaPron.

Altogether 248 audio files, each containing a first and a second name, were used in the evaluation. These files are the 10 subjects’ different attempts at pronouncing the names. The subjects were allowed to produce several different pronunciations for each name and every attempt was saved in a separate audio file. All these 248 audio files therefore contain a large set of pronunciation variations to test the LottaPron program with.

To be able to compare the difference in performance of Autopron and LottaPron, Batchrec had to be run on the output of both of the applications. The names in question were run through both Autopron and LottaPron. The phonetic transcription output from both applications was then tested with the Batchrec utility by putting the phonetic transcriptions from the different applications in to the dictionary one at a time.

There is also a feature which uses manually produced transcriptions for all the entries of the dictionary in question. Results from a Batchrec run with manually produced transcriptions will also be compared to the results from the evaluation runs for Autopron and LottaPron.

5.2.1 Recapitulation of the Batchrec utility

To perform a Batchrec evaluation, prerecorded audio files are needed (*Grammar Developer’s Guide Version 8.0* 1996-2001, p. 118). The recordings made by the subjects were used for this

purpose. The names were grouped together as a first name followed by a last name in the recordings and this is how the audio files looked when they were used in the evaluation.

Furthermore, three different files have been used for the Batchrec procedure: a *grammar file*, a *dictionary file* and a *transcription file* which contains a written transcription (*not* a phonetic transcription) of the audio files.

The *grammar file* is needed to list all the names in the dictionary (*Introduction to the Nuance System Version 8.0 1996-2001*, p. 14). For example, a grammar file looks like this:

```
PERSON
[
(nina johansson)
(bo andersson)
(hamdiya lepenica)
(emina semovic)
(djordje brstina)
(pia andersson)
]
```

The *dictionary file* contains all the different phonetic transcriptions of the names (*Introduction to the Nuance System Version 8.0 1996-2001*, p. 14), for example like this:

```
hamdiya          h a m d i dZ a
hamdiya          h a m d I dZ a
hamdiya          h A m d i dZ a
hamdiya          h A m d I dZ a
hamdiya          h a m d i j a
hamdiya          h a m d I j a
hamdiya          h A m d i j a
hamdiya          h A m d I j a
```

And finally, the *transcription file* holds all the written transcriptions of the audio files. By doing this Batchrec calculates accuracy statistics, per sentence and cumulative. The transcriptions must be spelled exactly as in the grammar, letter case and all as shown below (*Grammar Developer's Guide Version 8.0* 1996-2001, p. 125).

```
prompts\names\O_hamdiya_lepenica.wav hamdiya lepenica
```

Batchrec also calculates confidence scores which are a measure of the recognizer's confidence in its own answer. The confidence scores are used to decide whether or not the utterance should be rejected. A low confidence score leads to a rejection of the utterance. The confidence threshold is by default 45 on a scale from 0 to 100, which means that the system automatically rejects every utterance that generates a confidence score below 45. However, in this Batchrec run, the confidence threshold is set to 0 which forces the system to choose an interpretation at all times. This enables the system to show exactly where the mistakes took place and what kinds of mistakes they were (*Grammar Developer's Guide Version 8.0* 1996-2001, p. 126).

5.3 Results from transcriptions produced by Autopron

The Batchrec utility prints out statistics for every audio file that the utility has evaluated. The following are the statistics for the last checked audio file in the Batchrec run of the phonetic transcriptions produced by Autopron.

```
1.      File 248: prompts\names\N_nikola_vojnovic2.wav
2.      Grammar: .TOP
3.      Transcription: nikola vojnovic
4.      Result #0:      nikola vojnovic (conf: 83, NL conf: 82)
5.      NL Res.#0:      <pid "314">
6.      Total Audio: 3.63 sec
7.      Utt. Times: 0.65 secs 0.179xRT (0.66 usr 0.00 sys 0.181xcpuRT) 100%cpu
8.      Ave. Times: 0.59 secs 0.220xRT (0.59 usr 0.00 sys 0.2196xcpuRT) 99%cpu
9.      Rec Errors: 0 ins, 0 del, 0 sub = 0.00% of 2 words.
10.     Rec Total:  0 ins, 0 del, 14 sub = 2.82% of 496 words (2.82% of 248
      files).
11.     Rec Final Total: 0 ins, 0 del, 14 sub = 2.82% of 496 words (2.82% of
      248 files).
```

- 1) Shows which audio file is being analyzed.
- 2) Refers to the grammar that has been used to recognize this file.
- 3) Prints the transcription of the audio file as provided by the transcription file.
- 4) Shows the recognition result returned by the recognizer and the confidence scores of that result.
- 5) Refers to the natural language results that have not been used during this test.
- 6) The length of the recorded audio file.
- 7) The time it took to process the audio file (0.65 secs).
- 8) The average time it took to process all the audio files to this point (0.59 secs).
- 9) Shows the statistics for this file; number of incorrect words inserted by the recognizer, number of deleted words, number of misrecognized and substituted words and, finally, there is a count of the words in the audio file and a per-word error rate.
- 10) Shows the statistics for all the audio files that have been processed so far, including per-word and per-file error rates.
- 11) This is the final line from the Batchrec prints that sums up all of the statistics from the different audio files.

Thus the statistics for the Batchrec run with the phonetic transcriptions produced by Autopron display 14 substitutions of words, i.e 14 separate names (a first name or a last name) were misrecognized and substituted for another name. The 14 substitutions amount to 2.82 % of the 248 audio files.

5.4 Results from manually produced transcriptions

The possibility of manually producing transcriptions for all the entries of the dictionary in question is, besides from being time consuming, also prone to involve errors due to the human element. Even so, the manually produced transcriptions tend to be more accurate than the ones produced by Autopron. The following are the results from the Batchrec run on manually produced transcriptions for the evaluation names.

```

1.          File 248: prompts\names\N_nikola_vojnovic2.wav
2.          Grammar: .TOP
3.          Transcription: nikola vojnovic
4.          Result #0:      nikola vojnovic (conf: 83, NL conf: 82)
5.          NL Res.#0:      <pid "314">

```

```

6.      Total Audio: 3.63 sec
7.      Utt. Times: 0.65 secs 0.179xRT (0.65 usr 0.00 sys 0.179xcpuRT) 99%cpu
8.      Ave. Times: 0.61 secs 0.229xRT (0.59 usr 0.00 sys 0.2219xcpuRT) 97%cpu
9.      Rec Errors: 0 ins, 0 del, 0 sub = 0.00% of 2 words.
10.     Rec Total:  1 ins, 0 del, 16 sub = 3.43% of 496 words (3.23% of 248
        files).
11.     Rec Final Total: 1 ins, 0 del, 16 sub = 3.43% of 496 words (3.23% of
        248 files).

```

The results from this Batchrec run show slightly poorer recognition than from the Autopron utility. These 16 substitutions make up a 3, 23% error rate.

5.5 Results from transcriptions produced by LottaPron

Below is, once again, the final audio file checked in the Batchrec run. This time when the transcriptions made by the LottaPron program were added to the dictionary.

```

1.      File 248: prompts\names\N_nikola_vojnovic2.wav
2.      Grammar: .TOP
3.      Transcription: nikola vojnovic
4.      Result #0:   nikola vojnovic (conf: 85, NL conf: 85)
5.      NL Res.#0:   <pid "314">
6.      Total Audio: 3.63 sec
7.      Utt. Times: 0.70 secs 0.193xRT (0.70 usr 0.00 sys 0.193xcpuRT) 100%cpu
8.      Ave. Times: 0.62 secs 0.234xRT (0.62 usr 0.00 sys 0.2336xcpuRT) 99%cpu
9.      Rec Errors: 0 ins, 0 del, 0 sub = 0.00% of 2 words.
10.     Rec Total:   1 ins, 0 del, 2 sub = 0.60% of 496 words (0.40% of 248
        files).
11.     Rec Final Total: 1 ins, 0 del, 2 sub = 0.60% of 496 words (0.40% of 248
        files).

```

The results from the Batchrec run with the phonetic transcriptions produced by LottaPron show only 2 substitutions, i.e. 2 names were misrecognized by the system. The error rate for these two substitutions is 0,40% for all 248 files. The audio file that was not recognized correctly is the Serbo-Croatian name Djordje Brstina when pronounced by one of the male subjects. The confidence score for this file is merely 24 and the system has surprisingly interpreted Djordje Brstina as jan erik krook!

5.6 Comparison of the results

5.6.1 Substitutions

Comparing the three final lines from the Batchrec runs above gives a quick first look of the performance of the different transcriptions. Also, the different error rates refer to the differences in performance.

Autopron:

Rec Final Total: 0 ins, 0 del, 14 sub = 2.82% of 496 words (2.82% of 248 files).

Manual:

Rec Final Total: 1 ins, 0 del, 16 sub = 3.43% of 496 words (3.23% of 248 files).

LottaPron:

Rec Final Total: 1 ins, 0 del, 2 sub = 0.60% of 496 words (0.40% of 248 files).

Looking at the substitutions, i.e. the names that were not recognized and replaced by another name by Batchrec, one can see that Autopron and the manually produced transcriptions generated a greater number of substitutions than LottaPron. The only name that was misrecognized by LottaPron is, as mentioned above, Djordje Brstina. This name was recognized with the transcriptions from Autopron but not with the manual transcriptions.

Interestingly enough, almost all of the misrecognized names in the runs with Autopron and manually produced transcriptions consisted of a first name that already existed in the standard dictionary. In fact, the only misrecognized name that was not in the standard lexicon is Djordje Brstina. This is correspondingly the fact for all the three Batchrec runs. This could either indicate that the last names were too badly transcribed for the system to recognize or it could mean that the transcriptions for the already existing names in the standard dictionary were deficient.

5.6.2 Confidence scores

Unfortunately, the confidence scores are not automatically calculated for the entire test set. The mean average of the confidence scores have therefore been calculated by hand for each of the three Batchrec runs. These average scores are based upon all the evaluation names, i.e.

also the names that existed in the standard dictionary are included. Remember that the confidence scale spans from 0 – 100, where 100 is absolute recognition and 45 is the default threshold for rejection.

The mean average for the Autopron run is 68,8 and for the manual transcriptions the result is 69,0. These results from the Autopron run and the manual transcriptions are high and represent good speech recognition. However, the calculations based upon the LottaPron run are even higher at 71,9 which indicates a stronger speech recognition performance based on the transcriptions from the LottaPron program.

5.7 Discussion of the results

The evaluation results may not be completely accurate since the used name set is too small. Also, the fact that 18 of the 40 names in the test set already existed in the standard dictionary makes it harder to correctly calculate the accuracy scores. The results of the evaluation runs, that have been presented above, can therefore not be seen as statistically significant but should instead be regarded more as a pointer to the real performance of a future better developed LottaPron program.

The results from the substitution rates show that the transcriptions from LottaPron generate far fewer substitutions than the manually produced transcriptions and the ones from Autopron. This indicates that the transcriptions from LottaPron cover more of the pronunciation variations than the other transcriptions. This may be since LottaPron generates as many variations as possible for each name when Autopron settles for generating up to ten of the most plausible ones. For example, where Autopron generates three variations for Lepenica:

```
lepenica          l E p E n I k a
lepenica          l e p E n I k a
lepenica          l E p E n I tS a
```

LottaPron generates far more:

```
lepenica          l E p e n i tS a
```

lepenica	l E p e n I tS a
lepenica	l E p E n i tS a
lepenica	l E p E n I tS a
lepenica	l e p e n i tS a
lepenica	l e p e n I tS a
lepenica	l e p E n i tS a
lepenica	l e p E n I tS a
lepenica	l E p e n i k a
lepenica	l E p e n I k a
lepenica	l E p E n i k a
lepenica	l E p E n I k a
lepenica	l e p e n i k a
lepenica	l e p e n I k a
lepenica	l e p E n i k a
lepenica	l e p E n I k a

The most plausible variations are probably for the most part enough for the system to correctly recognize a name. This is why Autopron can produce such low error rates as 2,82% of 248 files. But, as mentioned before, proper names are not always pronounced in the most regular way which means that Autopron will fail in those situations where the name is not pronounced as would be expected at first glance.

This significant difference in generated variations, which is a pervading difference of all the variations produced by the LottaPron program and the Autopron utility, clearly affects the confidence scores as well. The more variations the speech recognition system has to choose between the easier it is to always find at least one pronunciation that the system trusts is the correct one and thus can be given a high confidence score. Therefore, the transcriptions from the LottaPron program generate a higher confidence score mean average than the other two test sets. The fact that the manually produced transcriptions generate a higher confidence score average than the ones produced by Autopron indicates that the human mind is more able to predict the most common variations of a name than Autopron.

However, merely producing as many transcription variations as possible, as LottaPron does, may not be very successful in the long run. If the LottaPron program was to be further expanded, containing more rules and covering several languages, the amount of possible transcriptions for one word would be very large. This would probably mean that several of the transcriptions end up looking similar to each other even though they do not transcribe the same word. This gives the recognizer more opportunities to make the wrong recognition. To avoid this scenario an order of precedence for the rules has to be developed which determines which of the transcriptions that should be picked first by the recognizer.

6 Concluding remarks and future developments

The purpose of this thesis has been to develop an automatic generator of phonetic transcriptions of foreign names, called LottaPron, which is to be used in a speech recognition process. The pronunciation rules in the LottaPron program have been developed with the help of pronunciation variations for Finnish and Serbo-Croatian names produced by 10 subjects.

The evaluation of LottaPron indicates a small improvement in the speech recognition performance. This is probably because LottaPron produces all possible variations instead of as with Autopron, only the most probable variations. The confidence scores of the Batchrec evaluation show higher results than both transcriptions produced by Autopron and the manual transcriptions for the transcriptions made by the LottaPron program. Also, the error rate for the amount of misrecognized names is lower for the LottaPron program than for the two other test sets.

A further development of the LottaPron program would be to examine a larger amount of names with the help of several more subjects. Naturally, the name and the subject sets that were used for gathering the initial data to build the LottaPron program upon, are too small for reliable statistical results. Therefore, a larger scaled and a more thorough investigation of the pronunciation variations for the two languages should be in place if a more accurate program is to be constructed.

Also, a wider development of LottaPron would be to examine the pronunciation variation for several more languages and add rules for these pronunciations. These rules would obviously also have to be based upon subject pronunciations.

In connection to the extension of pronunciation rules and language groups a further progress would be to develop a ranking order for the transcriptions. This would prevent the recognizer from confusing similar transcriptions with each other when the amount of transcriptions grows very large.

In conclusion, with a few more rules and language groups, for example Middle Eastern and African languages, the LottaPron program would be able to process a large part of the ‘new’ names in Sweden.

Bibliography

Application Developer's Guide Version 8.0 (1996-2001). Nuance Communications Inc.

Chomsky, Noam & Halle, Morris (1968). *The Sound Pattern of English*, New York: Harper & Row, Publishers.

Dahlstedt, Karl-Hampus (1971). *Svårigheter i svenskans uttal. En handledning vid undervisningen av finska, tyska och jugoslaviska invandrare*, Lund: Gleerups.

Eklund, Robert & Lindström, Anders (2001). Xenophones: An investigation of phone set expansion in Swedish and implications for speech recognition and speech synthesis, *Speech Communication* **35**: 81-102.

Elert, Claes-Christian (1970). *Ljud och ord i Svenskan*, Stockholm: Almqvist & Wiksell.

Ethnologue language family index. [Electronic].

Available: <<http://www.ethnologue.com>> (December 2004).

Grammar Developer's Guide Version 8.0 (1996-2001). Nuance Communications Inc.

Hammond, Michael (2003). *Programming for Linguists: Perl for Language Researchers*, Malden: Blackwell Publishing.

Introduction to the Nuance System Process Version 8.0 (1996-2001). Nuance Communications Inc.

Jurafsky, Daniel & Martin, James H (2000) *Speech and Language Processing. An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, New Jersey: Prentice Hall.

Katamba, Francis (1989). *An Introduction to Phonology*, London: Longman.

Kenstowicz, Michael (1994). *Phonology in Generative Grammar*, Cambridge: Blackwell Publishing.

Linell, Per, Svensson, Bengt & Öhman, Sven (1971). *Ljudstruktur. Inledning till fonologin och särdragsteorin*, Lund: Gleerups.

McAllister, Robert (1998). *Talkommunikation*, Lund: Studentlitteratur.

Nationalencyklopedin (1989-1996). Höganäs: Bra Böcker AB.

Serbernas Riksförbund i Sverige. [Electronic].

Available: <<http://www.srpstvo.com/swe/vesti/index.php>> (December 2004).

Serbo-Croatian Pronunciation Guide. [Electronic].

Available: <<http://www.dimensional.com/~phantomr/folkdanc/alphabet/serbo-croatian.htm>> (December 2004).

Appendix A

Swedish

Phoneme set

The following table shows the pronunciation mappings for the Swedish language.

Phonetic Category	Phoneme	Example	Phoneme	Example
Vowels	I	Sil	Y	Syl
	I	Sill	Y	Syll
	U	Full	E	Hel
	U:	Ful	E	häll, herr
	A	Matt	%	Nöt
	A	Mat	&	mött, förra
	u:	Bot	O	Mål
	U	Bott	O	moll, håll
	E:	Häl		
Stops	P	Pol	D	Dop
	B	Bok	K	Kon
	T	Tok	G	God
Nasals	M	Mod	g~	Lång
	N	Nod		
Fricatives	F	Fot	S	Sot
	V	Våt	X	Sjok
	C	kjol, tjugo	H	Hot
Approximants	R	Rov	J	Jord
	L	Lov		

Phonetic Category	Phoneme	Example	Phoneme	Example
Phone Sequences	r d	Bord	r s	Fors
	r l	Karl	a O	Count
	r n	Barn	a I	Flight
	r t	Bort	E U	Euro

Phonetic Category	Phoneme	Example	Phoneme	Example
Foreign Phones	TS	Chew	T	Keith
	DZ	Giorgio	S	Nashville
	W	Worldwide		

Appendix B

The first name collection

Namninsamling

Instruktioner

Ring upp 08-545 451 12. Du kommer att höra en kort instruktion. Därefter är det fritt fram att börja läsa in! Läs namnen ett i taget dvs. [Förnamn Efternamn] tillsammans. Gör en kort paus efter varje rad. Läs i normal fart; inte övertydligt eller överlångsamt. Om du har flera uttalsförslag på ett namn så läs dem också! Läs då de olika versionerna efter varandra dvs. en förnamnsversion följt av eventuella förnamnsversioner och därefter efternamnet följt av eventuella efternamnsversioner. Alltså kan ett namn med flera versioner se ut så här: [Viveka, Vivéka, Duit, Doit, Duvit]. När du har läst hela listan är du klar! Tack så mycket för din medverkan!

Namnlistan

Sara Farahmand

Teresia Dropulic

Anette Gaude

Erkki Nurmela

Jenny Vokbus

Nina Grigorieva

Marie Agobian

Jukka Väyrynen

Christina Humble

Davorko Brkic

Bruno Romskaug

Keijo Sipinen

Ywonne Gauffin

Borivoje Radojcic

Zenita Castemo
Maarit Hiltunen
Allan Pinaitis
Barbara Wozniak
Eric Verban
Matti Suomela
Roy Poltan
Anela Miletic
Lars Fornarve
Päivi Sihvo
Gitte Timan
Almir Malkoc
Eva Knutas
Risto Liikanen
Cecilia Sanchez
Aida Omerbasic
Gustav Wachtmeister
Seppo Hampunen
Bengt Schulze
Tarja Kokko
Tayana Bengzon
Vesna Stojic
Orwokki Östlund
Petri Tupasela
Adriana Rosenberg
Bojan Ilic

Namninsamling**Instruktioner**

Ring upp 08-545 451 12. Du kommer att höra en kort instruktion. Därefter är det fritt fram att börja läsa in! Läs namnen ett i taget dvs. [Förnamn Efternamn] tillsammans. Gör en kort paus efter varje rad. Läs i normal fart; inte övertydligt eller överlångsamt. Om du har flera uttalsförslag på ett namn så läs dem också! Läs då de olika versionerna efter varandra dvs. en förnamnsversion följt av eventuella förnamnsversioner och därefter efternamnet följt av eventuella efternamnsversioner. Alltså kan ett namn med flera versioner se ut så här: [Viveka, Vivéka, Duit, Doit, Duvit]. När du har läst hela listan är du klar! Tack så mycket för din medverkan!

Namnlistan

Waldemar Guricke

Hamdija Lepenica

Thomas Bajohr

Anna Laaksonen

Stefan Mirow

Emina Semovic

Mirosława Forsberg

Erkki Kataja

Mathias Demetriades

Elena Ionescu

Marcus Kopa

Hanna Talasmäki

Maria Dewoon

Ismet Kurspatric

Gilny Waldebäck

Jan Paakkari

Johannes Mooe
Djordje Brstina
Erik Abele
Leila Rönkkö
Diana Hemanus
Gordana Andrejic
Christel Guillet
Margit Perttu
Alexandra Kassimova
Dragan Romcevic
Bertil Hiertonn
Marit Krauja
Anna Sieurin
Nikola Vojnovic
Carina Schanson
Maria Lahtonen
Melania Alcalde
Dejan Tisma
Heinz Hagenbeck
Pekka Haapanen
Ewa Orsini
Milinko Pralica
Barry Murman
Veera Kapari