Brazilian Portuguese Speech Recognition for Navigation on Mobile Device Applications

Edwin Miguel Triana Gómez
etriana@ime.usp.br

Thesis proposal

submitted to the
Institute of Mathematics and Statistics
of the
University of São Paulo
in partial fulfillment for the
qualification of the
Master in Computer Sciences

Advisor: Fabio Kon

São Paulo, June 2010

Abstract

Modern handheld devices are equipped with high capability components allowing the use of sophisticated applications such as Web browsers and spreadsheets. Because of low electricity dependency, mobile devices can take advantage of software tools in non-typical scenarios where desktop computers cannot operate. However, there are some usability issues with these devices because of their small screens and lack of keyboards. Many graphical components on small screens and complicated navigation schemas result in low user satisfaction. An approach to reduce this negative impact is the use of multimodal interfaces such as a spoken language interface. With this type of interface, a speech recognition module collects user utterances and converts them to word chains. These word chains can then be used as system commands. Recent work has shown that current mobile devices are able to execute speech recognition applications despite their lower processing abilities.

The Borboleta telehealth system is a open source mobile device application that supports data collection during residential visits carried out by a specialized healthcare team. During each visit, the healthcare professional checks the patient health status and collects information using the Borboleta interface. If the professional is focused just on the patient, he could forget to record important patient data, but if he is focused on the system operation, he could fail to get some relevant information from the patient. The current research proposal addresses the goal of reducing the user attention level required by Borboleta operation by providing speech recognition capabilities to augment navigation through the software functions, allowing the professional to pay more attention to the patient. The proposed project methodology is composed of a bibliography revision to establish the state-of-the-art of the field, a review of available speech recognition software, a data collection of Brazilian utterances to train and test the system, a design and development stage that will define the system architecture and integration with the Borboleta and a testing process to measure the system accuracy, its usability and acceptance level.

Contents

1 Introduction

New technologies on mobile devices have increased their application range. Initially, these were used as mobile phones, note takers, or address directories. New features include high performance processors enabling the use of better graphical user interfaces and execution of specialized tasks, multi-platform, high-level development environments such as Java ME, wireless LAN (Wi-Fi) for faster connectivity, global positioning systems, and multi-touch screens. At the same time, new software has been developed, from small applications like multimedia players to more complex ones like Web browsers or calendar managers. However, the small screens have been a drawback for the use of sophisticated applications. Too many functions on tiny screens or too many window changes to complete a task result in low user acceptance levels [26]. Multi-modal interfaces are one approach to this issue, where input methods such as touch screens, buttons, and voice allow clear interfaces and receive better usability results.

A specialized application is the Borboleta system, a Java ME project to collect medical information in the primary care program of the Brazilian health system SUS [11], being developed at the University of São Paulo FLOSS Competence Center[1]. Since the Borboleta system offers different application features, good device interaction like well organized forms, shortcuts, menus, feedback (e.g., dialogs or sounds) and speech recognition to execute user commands is important.

One advantage of using voice input is that it is a natural skill and does not require much attention or direct observation to use the device, allowing increased attention to other activities [35]. Other advantages are the reduction of hand use or keyboard interaction to input information. This makes access easier for handicapped persons. Nevertheless, using voice input for interaction has some challenges such as environment noise, speech accent, large vocabularies, low quality microphones, and device capability in voice processing. In the last twenty years, robust speech recognition systems have addressed these problems. Two open source examples are HTK [51] and Sphinx 4 [48], which have very acceptable performance levels with low word error rates [40]. However, these systems were developed for desktop computers with good processing resources. In mobile devices, the limited processing and storage resources make the implementation of voice recognition systems even harder.

This work examines human computer interaction for the Borboleta system and the implementation of an automatic speech recognition (ASR) module to the navigation schema.

1.1 Problem statement

Mobile devices offer the opportunity to fast access to information, at any moment, in practically any place. That availability motivates researchers and developers to implement new applications. The goal in these implementations is to provide useful tools to access information in work environments where normal computers cannot be easily used. Issues such as device size and wired-electricity dependency make desktop computers unfit for many tasks. Mobile devices can better handle non-regular workplaces, such as data collecting in outdoor environments.

In the mobile devices field, useful or important applications require user attention and focus to produce correct input or perceive device output. However, when these are done in combination with other important tasks, there can be detrimental effects to the task at hand as well as to the data input. An example is getting directions from a GPS system while driving in a large city, which requires entering a destination address and reading the result from a 5 inch screen. An approach that tries to reduce this type of problem is the use of multi-modal interfaces [35], involving different input and output methods to reduce the

[1] http://ccsl.ime.usp.br/borboleta

required attention level by making the device easier to use. An example of this approach is the use of speech recognition as an input method along with a touch screen and speech synthesis to indicate system response.

Borboleta is a system for collecting patient information using mobile devices, which requires a high level of user attention. During a medical visit, the doctors see the patients, check their health conditions and at the same time input medical information into the device. While using the Borboleta system, if the professional is focused on system operation, he or she may skip important patient information. On the other hand, if he or she focuses on the patient, he or she may manipulate the device incorrectly and record erroneous information. Moreover, medical visits are time limited due to patient commitments and other scheduled visits for the professional. Data collection procedures must be done quickly, while still collecting accurate and useful patient information.

1.2 Objectives

Currently, Borboleta has a sophisticated user interface because of its multiple functionalities. That interface is composed of approximately 23 different screens, which contain different graphical components such as tabs, textfields, lists, and checkboxes, specific system menus and a defined navigation schema. To access any functionality in the system, it is necessary to know the complete path to reach it. Such a path could have more than two steps or intermediate windows, increasing the operation time for an individual task. A possible solution is the use of shortcuts to go to specific places, but this requires adding new interaction elements such as menus or buttons. Including new buttons increases the interaction elements displayed on the small screen. For this reason, our objective is to verify if the introduction of voice as an alternative input method to navigate through the system will improve the Borboleta system's usability. If so, professionals could focus on the more important activity of patient care and less on the application operation.

Speech recognition of the English language has been widely studied in previous work, but Brazilian Portuguese has had few initiatives in this field. There have been recent efforts to provide tools for speech recognition for Brazilian Portuguese, such as the FalaBrasil initiative [33]. Most of these projects are based on a vocabulary of commonly used words. In the medical domain, there are no relevant projects that use Brazilian Portuguese for speech recognition. Our specific objective is to use a small Portuguese vocabulary to control the Borboleta system, and to open the path for speech recognition of health care notes, requiring a larger specialized vocabulary.

For navigation, there are three main approaches that could be carried out to perform the recognition task: template matching, phoneme recognition, and whole-word recognition. Each one has strengths and weakness depending on the problem domain. The objective is to evaluate and conclude how these perform within the Borboleta system. In order to implement an automatic speech recognition (ASR) system on mobile platforms we will evaluate whether the mobile devices currently used in Borboleta can support speech recognition and whether the implementation improves the user experience controlling the mobile device. Previous work has implemented English speech recognition for mobile devices [24, 53], but most of these have been distributed speech recognition, where the signal processing phase happens on the device and the actual speech recognition happens on a distant server. For system navigation, all calculations need to be executed on the mobile device; in particular, in the case of the Borboleta project, the mobile devices does not even have network connectivity during the home visits.

1.3 Overview

This document consists of a literature review covering speech recognition and mobile computing, followed by a methodology section, which describes the approach to addressing the question of the usefulness of a multimodal interface to the Borboleta mobile system. The document concludes with a description of preliminary results and future implications of this work.

2 Background

The task of speech recognition incorporates concepts from theoretical and applied sciences including signal processing, stochastic processes, search algorithms, and software engineering. This section provides a historical review of speech recognition, its state of the art, and the basis of the field. This is followed by a description of existing Brazilian Portuguese speech corpora. The mobile platform for development is characterized, and the section concludes with an overview of the the Borboleta system.

2.1 History of Speech Recognition

Transcribing utterances to text is not a recent challenge. Current automatic speech recognition (ASR) systems with large databases and low word error rates have their roots in the inspiration to simulate human behavior on machines.

Since the late 1800s, there were approaches to reproduce human speech in works such as von Kempelen's "Speaking Machine" [15], but it was not until 120 years later that Bell Laboratories registered the relationship between the speech spectrum and its sound characteristics [25]. This important relationship resulted in the implementation of different speech recognition algorithms based on speech power spectrum measurements. Some decades after that (1950-1970), science fiction works such as Isaac Asimov's novels, the television series *Star Trek* (1966), and the movie *2001: A Space Odyssey* (1968) increased the interest in human-machine interaction, specifically, the use of the voice as a tool to communicate with computers and control them. Relevant advances were reached in this period. In 1952, an ASR system for isolated digit recognition was built using formant frequencies, which are spectral peaks manifested in the speech power spectrum [25]. In 1962, the article "The phonetic typewriter" [39] was published containing an analysis of different utterance portions using a speech segmenter, an early approach for continuous speech recognition systems [25].

Tom Martin's work in dynamic programming for speech pattern aligning in non-uniform time scales [31] had an impact on the Sakoe and Chiba research on dynamic time warping (DTW) [25], an important concept used in conjunction with different search techniques such as the Viterbi algorithm, where extracted features are compared with previous models in order to find the corresponding word. With the aim of developing speaker-independent systems, AT&T Bell Laboratories developed algorithms to represent sound patterns for words with a vast number of samples. This work culminated in the development of statistical modeling techniques [25]. Such techniques were the basis for hidden Markov models, currently the most common approach in the field.

Until the 1980's most speech recognition methodologies used template-based pattern recognition. In this decade the statistical approach emerged with the use of hidden Markov models (HMMs), a framework developed at AT&T Bell Laboratories and based on research at the Institute for Defense Analysis (IDA) in the late 1960's and James Baker in 1975 [1]. HMMs became the most popular method because they are robust for continuous recognition and variation in time, and allow speaker-independence capabilities.

Also in the 1980s, an alternative technology was proposed. From the field of artificial intelligence came the use of artificial neural networks (ANNs). From the point of view of pattern recognition, a multilayer perceptron, a particular form of parallel distributed processing, could perform the mapping between utterances and words with an acceptable accuracy level, following a training process using the back propagation method of ANNs. This approach gave good results for isolated word recognition, but not for continuous words with large data sets because of the temporal variation on speech recognition [25].

2.2 State of the art

The last two decades have seen great advances in the improvements in HMMs and implementations of robust automatic speech recognition systems in academia, the military, and industry. The following is a brief description of some of the more relevant speech recognition systems, focusing on open source systems.

Sphinx: In 1988, the first version of Sphinx was released [28]. Sphinx is an open source HMM-based speaker-independent speech recognition system. Since this early version, Sphinx has evolved. Sphinx-II (1993) reduced the word error rate from 20% to 5% on a 5000 word vocabulary by using a data-driven unified optimization approach [21]. In 1997, the new Sphinx-3 could be configured at run-time along the spectrum of semi- to fully-continuous execution [36]. Sphinx-3 was developed in C++ and is still in active development. Pocketsphinx, a Sphinx-II version developed for hand-held devices, was released in 2006 [22]. This version was designed for continuous speech recognition on devices with low hardware capabilities. Despite its features, Pocketsphinx has only been tested on desktop computers and more recently on Linux-based mobile devices [46] and on the iPhone. The most recent version of the Sphinx series is Sphinx-4, developed in Java [48], which presents a modular framework for HMM-based speech recognition systems. Sphinx has been the base of many projects, including Lumenvox, Microsoft Whisper, Jvoicexml, Gnome-Voice-Control and Zanzibar [7].

HTK: The Hidden Markov Model Toolkit (*HTK*) was developed to support hidden Markov model processing, primarily for speech recognition tasks. The Speech Vision and Robotics Group of Cambridge University released the first version in 1989 which supported diagonal and full covariance Gaussian mixture distributions for observation probabilities. In 1995, HTK became part of the Entropic Cambridge Research Laboratory (ECRL). In this year, a redesigned platform was released as version 2.2, which supported discrete density HMMs, cross-word triphones, lattice and N-best recognition output, back-off bigram language models and decision-tree state clustering. Microsoft purchased the ECRL in 1999, and all HTK rights went to Microsoft. In September 2000, the source code was released with the aim to improve the system, and to let researchers and developers around the world participate in project development. The current version (3.4) has an improved set of features for speech recognition such as perceptual linear prediction (PLP), N-gram language modeling, speaker adaptive training, discriminative training and large vocabulary decoding [3].

Julius: An open source continuous speech recognition system for large vocabularies, Julius was initially developed at Kyoto University in 1991, and from 2005 was copyrighted by the Nagoya Institute of Technology. This system performs recognition with up to 60K words, and it has a recognition rate of 90% in real time. The system uses N-gram language models and HMM acoustic models, and it is widely compatible with HTK file representation formats (HMM definition and model files, feature parameters). Desktop Linux is the target platform, but there are versions for Windows and Solaris [32]. An embedded Julius-based version was developed to run on a T-engine board with a 240MHz RISC processor, a developmental hardware platform [27]. A desktop speech recognition system for Brazilian Portuguese based on Julius was developed at the Brazilian Federal University of Pará [42].

Commercial applications: A broad range of commercial speech recognition systems are available. Most of these are HMM-based and compatible with desktop computers. Some of the systems are: Byblos, developed by Raytheon BBN technologies [6], Decipher, developed at SRI [8], Dragon NaturallySpeaking from Nuance [34], and ViaVoice, an IBM speech recognition system [23].

A common measure for evaluating speech recognition accuracy is word error rate (WER) shown in Equation 1.

$$WER = 100\% * \frac{Subs + Dels + Ins}{\#\ of\ words\ in\ the\ correct\ sentence} \tag{1}$$

where $Subs$ is a count of substitutions for a correct word, $Dels$ are omitted words and Ins are inserted words [20].

Table 1 shows typical automatic speech recognition word error rates (WERs) as of 2007 for continuous recognition on different speech tasks [49]. Connected digits is a domain of digit strings recorded in a clean environment. The DARPA resource management corpus includes military commands, queries, and responses in a clean environment. The recognition task allows no more than 1000 words. The Wall Street Journal corpus includes speech read from newspaper stories in a clean environment, with a 5000 word vocabulary for the testing task. The SWITCHBOARD corpus is spontaneous conversations in a telephone environment, which makes it a more difficult task. The best WERs for typical tasks in this domain are still around 20% [16, 19] depending on the particular test set and modeling approaches. For small vocabularies (e.g., connected digits) and clean environments, accuracy rates close to 99% are expected. WER increases for larger vocabularies, unplanned or spontaneous speech, multiple speakers and noisy environments.

Table 1: *Current recognition word error rates on desktop ASR systems [52]*

Task	**Vocabulary Size**	**WER (%)**
Connected digits	11	0.6
Resource Management	1,000	2.7
Wall Street Journal	5,000	7.2

2.3 Speech Recognition

Automatic speech recognition is a process where utterances are translated to character strings with an equivalent meaning. That character string might be a set of digits, a word or a sentence, and can be used as an execution command, to get the text of the speech for subsequent processing, or in automatic dialog or speech understanding systems.

A typical speech recognition system is composed of two phases, a front-end and a back-end (Figure 1). The front-end module processes the voice signal to get observation vectors that are used in training and recognition [2]. The back-end module is comprised of a word search, combining information from acoustic and language models extracted from a training phase.

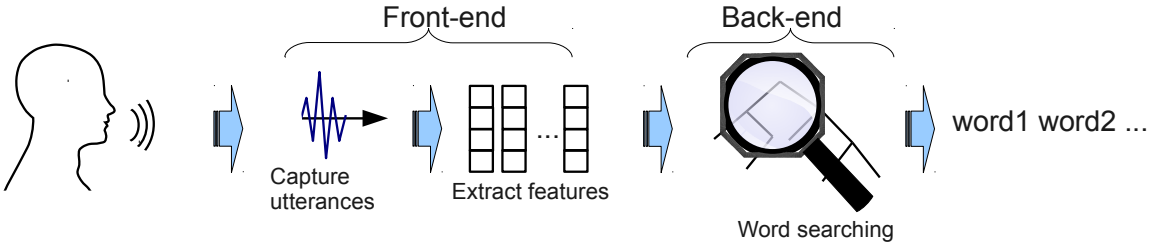


Figure 1: *Speech recognition process*

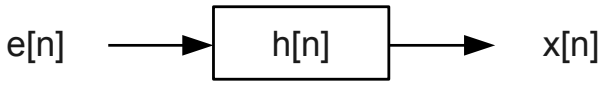


Figure 2: *Speech recognition process*

2.3.1 Front End: Signal processing

In the front-end, voice utterances are digitized to waveforms and subsequently transformed to a parametric representation for further processing. Different approaches are used to get the parametric representation. Filter bank analysis [37], linear predictive coding (LPC) [37], and Mel frequency cepstral coefficients (MFCCs) [20], have been efficient techniques in speech signal feature extraction. Characterization of the signal is based on the source-filter model, a speech production model where the speech $x[n]$ is the result of the interaction of the air from the lungs (source or excitation $e[n]$) with the vocal tract (filter $h[n]$) in time [20]. Figure 2 shows the source-filter model. The objective in this stage is to separate the source and the filter so that filter information can be used in recognition. Different techniques to estimate the filter are based mainly on speech production models, such as the filter-bank and LPC, and speech perception models, like MFCC. When the filter is estimated, an inverse filter is used on the speech to get the source [20].

This subsection will briefly describe three common approaches for representing the speech signal for subsequent processing: filter-bank, LPC, and MFCC. These techniques use as their starting points the spectral representation, which contains relevant frequency domain information. In speech recognition, the Fast Fourier Transform algorithm (FFT) is often used to get the spectral representation along with other transformations such as the discrete cosine transformation.

Filter Bank approaches divide the frequency of the signal into N bands to measure the intensity or energy in each one [37]. The energy of the segment is obtained by calculating the area under the curve or summing the squares of the values. Three types of filter banks are used: uniform, logarithmic, and non-uniform. A uniform filter bank divides frequency into uniform intervals, losing useful signal characteristics because lower frequencies carry more information. Logarithmic filter banks divide the signal logarithmically, simulating human auditory perception. If the first range has a width W, then subsequent ranges will be $a^n W$, where a is a logarithmic growth factor. In non-uniform filter banks, ranges are composed of two parts. In frequencies from 100Hz to 1KHz ranges are spaced equally; from 1KHz, ranges will be divided with the logarithmic scale. This partition is known as the Mel scale [20], and it is widely used in speech recognition because it emphasizes lower frequencies.

Linear Predictive Coding coefficients (LPC) estimates the value *x[n]* of a signal, by combining the past p speech samples [20]. So if we have

$$x[n] = \sum_{k=1}^{p} a_k x[n-k] + E[n] \tag{2}$$

where the coefficients a_k are constants over the speech frame, and $E[n]$ is the prediction error. Then, the estimated signal is given by Equation 3

$$\tilde{x}[n] = \sum_{k=1}^{p} a_k x[n-k] \tag{3}$$

and the error is given by Equation 4

$$E[n] = x[n] - \tilde{x}[n] = x[n] - \sum_{k=1}^{p} a_k x[n-k] \tag{4}$$

The aim here is to estimate the p coefficients that minimize the prediction error $e[n]$. A solution to estimate such coefficients is the Yule-Walker equation [20]:

$$\sum_{j=1}^{p} a_j \phi_m[i,j] = \phi_m[i,0] \qquad i = 1, 2, \ldots, p \tag{5}$$

where $\phi_m[i,j]$ is an LPC coefficient for the sample m.

Mel Frequency Cepstral Coefficients (MFCCs) are based on a signal representation technique that provides a better characterization than filter banks alone and is currently the state-of-the-art in automatic speech recognition [18]. The MFCC process is based on cepstral analysis, where multiplication of the source and filter components can be represented as the convolution in the time domain of the speech excitation signal and the filter. The source and filter can then be separated readily using basic filtering methods [18]. MFCCs are based on speech perception models, that is, a nonlinear frequency scale used to simulate the human auditory system. The MFCC process starts in filter-bank analysis, where a discrete cosine transformation is applied to the Mel filters. The resulting MFCC will maintain the characteristics of the non-uniform filter-bank and a smaller element vector will be needed to represent such information. Additionally, higher coefficients are removed, allowing faster computation in subsequent processing [18].

Once the signal representation has been selected, a distance metric must be defined to measure the similarity of the captured signal with previous data. Given parameter vectors of N elements, a real Cartesian space R^N will contain all real elements of the parameter vectors. Distance measures will be computed over this space. Three distance types are commonly used: City block, Euclidean, and Itakura metrics [38].

2.3.2 Back End: Speech Recognition Approaches

Once signal processing is performed, the next stage is to recognize the utterance, based on a previous training phase. **Training** is where characteristics of utterance samples are extracted to create models and later map new sounds to these models. Training depends on the recognition approach and can require generating templates (template matching) or acoustic models (e.g., hidden Markov models) coupled with language models (finite state grammars or N-gram models). In recognition, the system performs a mapping process to retrieve the closest word to the incoming utterance. Figure 3 illustrates the recognition in the back-end.

How a speech recognition system identifies utterances depends, initially, on the recognition type, whether isolated or continuous words are recognized, and secondly, how the system performs the recognition, that is, using template matching or statistical modeling techniques. **Isolated word** recognition refers to the identification of one word at time. Systems listening for commands or simple words are good candidates for isolated word recognition because each utterance will be related to a specific word or short phrase. **Continuous recognition** refers to the determination of word tokens in continuous audio streams. In this case, the system will process longer utterances, identifying the end of a word and the beginning of the next. Examples of this type of recognition are dictation systems and live translation applications. In continuous recognition, two approaches have been used to recognize speech. The **whole-word** method, where the system has models of whole words to match the speech, only

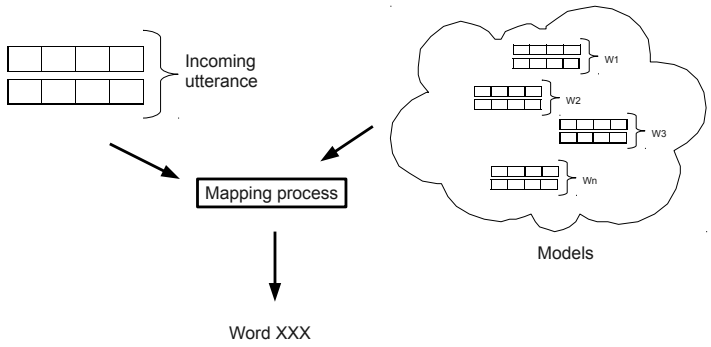


Figure 3: *ASR Back-end: recognition phase*

recognizes words that have been previously modeled. **Phone** recognition, where utterances are divided into sub-word segments, uses a dictionary, mapping these segments, called phones, to words. The system is able to identify previously seen words as well as words not seen but composed of phones which have been seen in training data.

Template matching is a pattern matching technique used mainly during the 1970s and 1980s. Foundations of this technique came from the fact that when a person says a word several times, each signal representation will follow a pattern which can be used to identify an utterance in a speech recognition system. Thus, if the system has the acoustic patterns (templates) of any supported word, the template matching method will find the closest template to the signal representation of the incoming utterance. Basically, the search consists of selecting the template with the lowest distance from the utterance. Euclidean distance is a common metric used in this technique. While this technique can work in simple applications, there are some limiting factors. First, while several recordings of a word from a single person will give us a pattern, it will not be valid for any other person. Thus, template matching is user-dependent, and each user must train the system by giving examples of the supported words. Second, this technique assumes that a word is completely defined, that is, the start and end of the word are readily determined. For isolated recognition this is not a problem, an utterance will match with a word. However, continuous word recognition must have an extra processing phase to divide the signal into individual words.

Time and pitch variations in utterance representations can be found in recordings from the same person. Word segments may vary in duration. For example, the word "computer" can be said as "compuuter" or "commputrr", both have a different syllable lengths but the same meaning. To deal with this time variation, a mathematical technique known as dynamic programming or dynamic time warping (DTW) is used. DTW minimizes the cumulative distance between two strings, but avoids the calculation of all possible combinations [18]. Figure 4 shows the optimal path between "computer" and "commputrr". Point i, j represents the lowest distance in this segment of the words. The minimal cumulative distance is given by Equation 6.

$$D(i,j) = \min[D(i-1,j), D(i,j-1), D(i-1,j-1)] + d(i,j) \tag{6}$$

For continuous recognition, the main problem is the end-detection of single words. In this case, template matching has been carried out by concatenating templates and performing the DTW algorithm on the entire sentence. To separate templates and identify words, a silence template is included between normal templates. Wildcard templates are used in sentences where the end of the word is not identifiable.

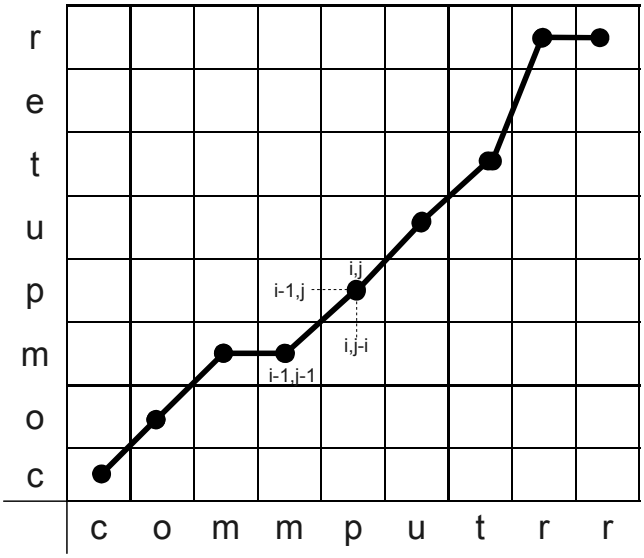


Figure 4: *Dynamic Time Warping*

Statistical approach: A second approach for the back-end phase is statistical modeling using a combination of acoustic and language models. While template matching is a useful methodology for isolated-word recognition, word boundary identification is difficult for connected-words and continuous speech. Speech variations such as pitch, length, internal noise (vocal tract variation), environmental noise and co-articulation effects at word boundaries make sentences harder to recognize. In the 1970s, hidden Markov models (HMMs) were proposed to confront these acoustic issues [1]. With HMMs, it is assumed that models can generate feature vectors of any supported word in the system [18]. The recognition process evaluates the *a posteriori* probability that a model could generate the same feature vectors as incoming utterances, but without actually generating such vectors [18]. The likelihoods of different model sequences can then be compared to choose the most likely hypothesis that could have generated the test sentence

Taking as an example an isolated word, which eliminates boundary problems, and given a set of feature vectors or observations O, defined as the set of observations along time t, $O = o_1, o_2 \ldots, o_t$; the recognition problem is to find the highest probability for a particular word given a set of observations, or $P(w_i|O)$, and is defined by Equation 7

$$\underset{i}{\text{argmax}}\{P(w_i|O)\} \tag{7}$$

where w_i is a word in the set of all possible in the system [51]. Using Bayes' rule, the probability distribution in Equation 7 becomes:

$$P(w_i|O) = \frac{P(O|w_i)P(w_i)}{P(O)} \tag{8}$$

Given a set of prior probabilities $P(w_i)$ represented by the language model, Equation 8 is reduced to the computation of $P(O|w_i)$, that is, $P(o_1, o_2 \ldots |w_i)$, an intractable problem because of the dimensionality of the vectors [51]. Here, the HMM is introduced to replace the $P(O|w_i)$ with a simpler solution. From the HMM diagram (Figure 5), the o_t vector is generated from the probability density $b_j(o_t)$, and the transition from a state i to a state j has the probability a_{ij}. The joint probability that O is generated by the model M is:

$$P(O, X|M) = a_{12}b_1(o_1)a_{22}b_2(o_2)a_{23}b_3(o_3)\ldots \tag{9}$$

where X is the unknown sequence through the model states. X is calculated by summing over all state sequences $X = x(1), x(2), \ldots, x(T)$. Thus,

$$P(O|M) = \sum_{x} a_{x(0)x(1)} \prod_{t=1}^{T} b_{x(t)}(o_t) a_{x(t)x(t+1)} \tag{10}$$

with $x(0)$ being the initial state and $x(T+1)$ the final state. The solution for Equation 10 is based on recursive procedures [51]. Finally, Equation 8 is solved by assuming that $P(O|w_i) = P(O|M_i)$. Parameters a_{ij} and $b_j(o_t)$ are estimated by performing training over all possible words in the training set. This process generates a model that represents the acoustics associated with real words. Baum-Welch re-estimation is used to calculate the HMM parameters [51]. The Viterbi algorithm performs the search for the $\text{argmax}_i P(O|Mi)$ to get the closest model sequence to the unknown sentence [51].

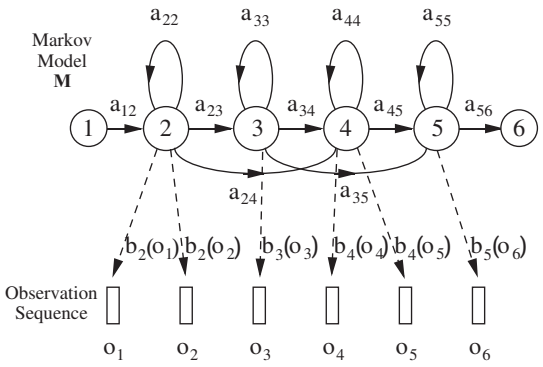


Figure 5: *Markov generation model [51]*

The global model M is characterized by the number of states in each component model, the size of the observation symbols, transition probability matrices $A = a_{ij}$, output probability matrices $B = b_i(k)$, that represent the probability of emitting symbol o_k when a state i is reached, and an initial state distribution π, $M = \{A, B, \pi\}$. In practice, $P(O|M_i)$ is an acoustic model, where B is a discrete output probability matrix for finite sets of (isolated) observations, or an output probability function for continuous recognition. Multivariate Gaussian mixture density functions are commonly used for the output probability function, because of their capability to approximate any probability density functions [20]. To improve the accuracy in the system, a large number of mixtures can be chosen, but this increases the computational complexity. On the other hand, discrete models have low computational complexity, but low performance levels. Semi-continuous models provide a good balance between system accuracy and computation level [20].

Semi-continuous and continuous speech recognition are dealt with by concatenating HMMs, where boundaries are represented by initial and final states. Similarly to template matching, boundary detection is a common issue. Typical HMM training uses a pre-training phase, called bootstrapping [51], to detect boundaries using Baum-Welch re-estimation in parallel. Connecting whole-word HMMs can be used to create sentences and connecting phone models can be use to create words and/or sentences.

When a search is being executed, the system must find the closest model sequence to the input utterance. To reduce the search space and improve the performance, *a priori* knowledge of the language structure can be used. The probability $P(W)$ represents the likelihood that the word string W corresponds to the utterance, and is defined by the language model. Here, a language model is a probabilistic model of the all possible sequences in the system. Stochastic language models, such as the n-gram, are commonly used in speech recognition where sequences of n words are used rather than all possible sequences. An n-gram model is

described as

$$P(W) = P(w_1, w_2 \ldots, w_n)$$
$$P(W) = P(w_1)P(w_2|w_1)P(w_3|w_1w_2) \ldots P(w_n|w_1, w_2, \ldots w_{n-1})$$
$$P(W) = \prod P(w_1|w_1, w_2, \ldots w_{i-1})$$

Limiting n to a value like 3 reduces the size of the model.

2.4 Brazilian Portuguese corpora

To train and test speech recognition systems, a data set of utterances and their transcriptions for a specific language must be available. There are only a few sources for Brazilian Portuguese speech corpora. Spoltech is a Brazilian Portuguese corpora with 8119 utterances from the Oregon Graduate Institute (OGI) and the Rio Grande do Sul Federal University (UFRGS) [13]. OGI-22 is another available corpus that consists of telephone speech from 22 languages [12]. This is a smaller corpus and only a few utterances have orthographic transcriptions. Previous work with these two corpora has been carried out at the Federal University of Pará. This work showed word error rates of 18.6% and 19.9% for Spoltech and OGI-22 respectively [33]. Globalphone is another multilingual corpus collected at Karlsruhe University [45]. For Brazilian Portuguese, the corpus was created by having 102 speakers read from the Folha de São Paulo newspaper. Globalphone is available under a commercial license which was too expensive for this project and would not allow open source licensing for the resulting code. VoxForge is a recent initiative to make available free resources for speech recognition [47]. Here, different corpora and acoustic models are available under GPL licensing. A Portuguese data set with a total of 51 minutes is available. All available corpora consist of vocabularies of commonly used words. Many utterances were elicited with requests for addresses, phone numbers, and personal information. While the breadth of speakers is useful, the specific vocabularies do not meet the needs of Borboleta, which uses speech in the healthcare domain.

2.5 Speech recognition on mobile devices

This section describes mobiles devices and their platforms, followed by speech recognition approaches for mobile devices and current available software. Finally, the Borboleta system architecture is briefly described.

2.5.1 Mobile devices and platforms

Mobile devices are portable electronic devices that can work in many environments. Typically, mobile devices have lower size and weight than normal computers. In today's market, there are a wide range of mobile devices, such as notebook computers, personal digital assistants (PDAs), mobile phones, calculators, game consoles, e-book readers, and portable media players. In this case, we will examine Borboleta compatible devices, that is, PDAs and mobile phones running Java mobile edition (Java ME). For our target devices, common characteristics include a processor with speeds between 300MHz and 1GHz, RAM from 32MB up to 512MB, storage (internal and/or external) from 512MB up to 16GB. For input/output methods, the devices are typically equipped with 4" touch screens, mini-qwerty keyboards, accelerometers, GPS, and audio and video capture/player libraries. The devices normally come with GSM and UTMS networks (mobile phones), wireless network connectivity (Wi-Fi) and bluetooth capabilities.

In the same manner as desktop operating systems, mobile platforms abstract hardware access and manage system resources, but mobile platforms take into account additional re-

quirements, such as battery life optimization in resource usage. Similar to the wide range of mobile devices available, many different platforms are available. Typically, mobile platforms are compatible with a specific set of devices. It is not possible to install another OS on a device if the new one is not compatible with the device. This issue causes device dependence on specific OSs, and execution of applications developed for a particular platform. Java ME was developed to address the portability problem. This platform is a virtual execution environment or virtual machine that guarantees a high compatibility level among different mobile platforms by adding virtualization middleware between the OS and the application. The interaction between application and virtual machine will thus be the same across all platforms but, between the virtual machine and the OS, there will be specific implementations. Symbian, RIM Blackberry, Windows Mobile, iPhone OS, Android and Palm OS are some of the more relevant operating systems for mobile devices. Figure 6 shows the market distribution for mobile operating systems for the first quarter of 2010.

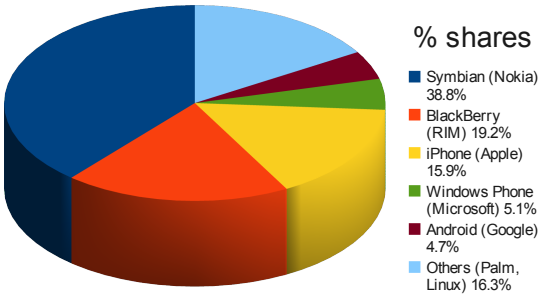


Figure 6: *Mobile OS market Q1 2010 [4]*

Symbian, which has a majority of the market share, is an OS for mobile devices, mainly for Nokia and Sony Ericsson devices. Symbian was developed in C++ and recently was released as open source. Symbian architecture, as shown in Figure 7, was designed as a multilayer system, where each layer offers specific functionalities and is able to communicate with the lower tier. Native applications run on the S60 applications layer using the language Symbian C++, an implementation of ANSI C++ optimized for mobile devices.

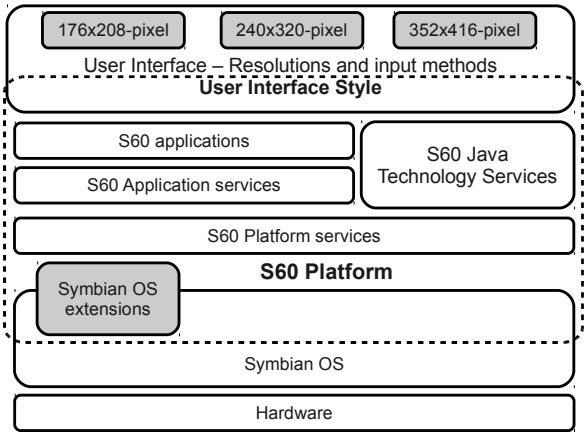


Figure 7: *Symbian architecture [49]*

The Borboleta project has been developed for Java ME compatible devices to maintain compatibility with most of the mobile platforms. Java ME provides two configurations for two ranges of devices. The first is connected limited device configuration (CLDC) for devices with low resources, such as early generation mobile phones and PDAs. The second is connected device configuration (CDC) which is used in devices such as new line mobile phones and

PDAs with at least 200MHz processors, multimedia enabled and with a wireless connection. Figure 8 shows the Java ME architecture for both configurations. Despite the high level of compatibility, not all Java ME implementations provide all the specifications of Java ME. Some implementations offer functionalities for basic operations, but do not support advanced tasks like multimedia processing. For example, multimedia functions like capturing video and audio are enabled in Nokia cellphones (Symbian OS), but unavailable for HTC mobile phones (Windows Mobile) [10].

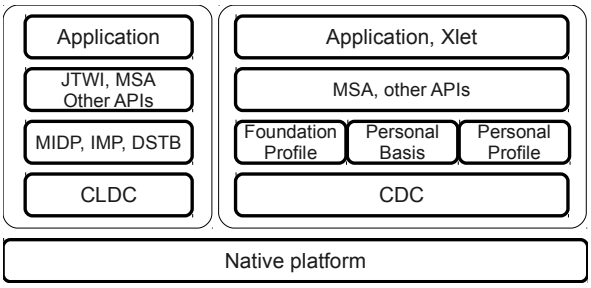


Figure 8: *Java ME architecture [43]*

Most automatic speech recognition (ASR) systems have been developed in C and C++ because of its high performance. Thus, in porting ASR systems to mobile devices, the software performance and language compatibility must be taken into account. Performance wise, C++ has an advantage over Java ME because it runs natively on the device, as opposed to Java ME, which is interpreted. Therefore, C++ directly accesses OS functionalities. For language compatibility, mobile OSs are mainly written in C++, so most of these devices could support speech recognition operations, with some modifications for specific platforms

Developing exclusively for Symbian brings portability problems with other platforms because of its specific syntax and data types. An approach to this issue is the portable operating system interface (POSIX), an initiative to unify the interface of the various Unix libraries. This same approach has been carried out for mobile platforms like Symbian, which currently can support applications developed for Linux ports [50]. Open C/C++ is a set of libraries developed by Nokia to make the porting process to Symbian easier. Table 2 shows the available libraries compatible with POSIX, although it is important to clarify that not all libraries are 100% compatible.

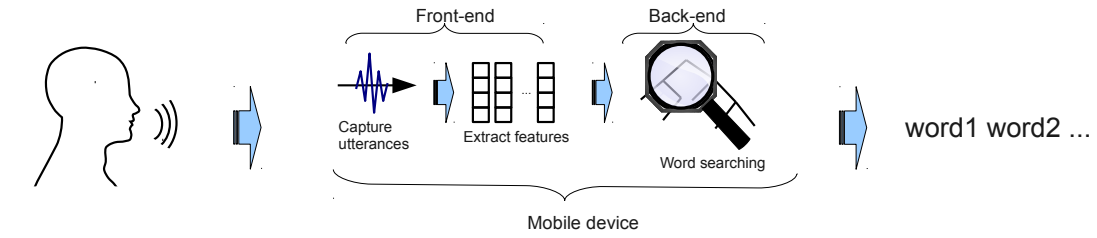
Table 2: *POSIX libraries for Symbian*

Implementation	Supported libraries
P.I.P.S	libc, libm, libpthread, libdl
Open C	P.I.P.S, libz, libcrypt, libcrypto, libssl, libglib
Open C/C++	Open C, IOStreams, STL, Boost

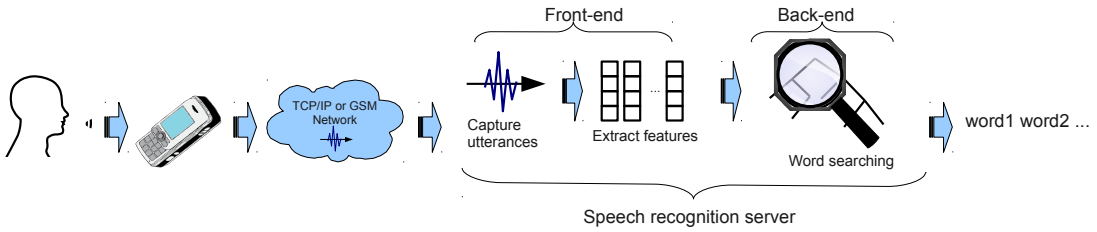
2.5.2 Speech Recognition on Mobile Devices

Because the speech recognition process consists of two phases, signal processing and recognition, it can be decoupled. Possible approaches for implementing automatic speech recognition (ASR) on mobile devices include embedded terminal systems (ETS), network speech recognition systems (NSR) and distributed speech recognition systems (DSR) [52]. An ETS (Figure 9.a) has full ASR embedded in the device. This system often has a small vocabulary, but does not require external communication. In the NSR (Figure 9.b), the front-end and the back-end are on the server and the mobile device simply transmits the speech signal to the server. Results could then be returned to the mobile device. The main

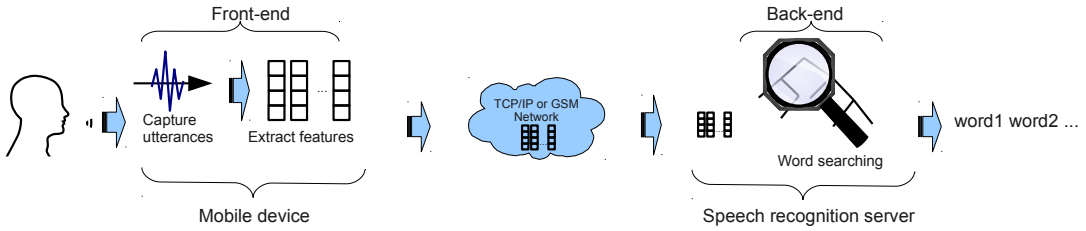
advantage of an NSR is that the server has more powerful resources, which allow the use of large vocabulary systems with low word error rates. On DSR systems (Figure 9.c), the front-end is embedded in the mobile device and the back-end is on a server. The device extracts the observation vectors and sends these to the server where they are recognized. An advantage of DSR over NSR is the low bandwidth used in the communication process because the vectors are smaller than the waveforms.



(a) Embedded terminal system



(b) Network speech recognition



(c) Distributed speech recognition

Figure 9: *Speech recognition approaches for mobile devices*

2.5.3 Available ASR Software

Despite the fact that speech recognition has had great advances in recent years, this technology has not been widely used on mobile devices. Speech recognition requires high processing power and resources that mobile devices did not have until recently. Approaches to achieve speech recognition on mobile devices began with template matching techniques, where the application had to be trained for specific users. Speaker dependence has undesirable effects on the use of this technology, as not all users are willing to train the system. If a new user wants to use a trained system, it will not be compatible with his voice. If a system is trained for many users, many templates must be saved and matching across the entire set uses too many computing resources. Currently, mobile devices are equipped with good processing units and many available resources, increasing the interest in developing more sophisticated applications. PocketSphinx and PocketSUMMIT are two speech recognition applications for mobile devices that are part of this recent initiative. A front-end Java ME implementation for a DSR was carried out by Zaykovskiy [53].

PocketSphinx is a continuous speech recognition system for hand-held devices developed by the Carnegie Mellon University Language Technology Institute [22]. The system is based

on Sphinx-II [21], but with the triphone-senone mapping representation from Sphinx-III [36]. To run on mobile devices, the software includes memory optimizations to reduce the use of RAM by directly accessing the ROM memory for acoustic and language models, data alignment access for modern CPUs, and the use of fixed-point arithmetic because floating-point operations on mobiles are typically emulated. Also, there were modifications in the speech recognition algorithms, primarily in the feature extraction procedures, Gaussian calculations, Gaussian mixture model computations and HMM evaluation. The system was tested on the Sharp Zaurus SL-5500 hand-held computer with a 206MHz processor, 64MB of SDRAM, 16MB of flash memory and a Linux-based operating system. After all modifications, the word error rate of the system on a 994 word vocabulary was 14% and operated in 0.87 times real time. A successful example of complete recognition processing on a mobile device is the Parakeet project [46]. This project is based on Pocketsphinx and was developed for a Linux-based (Maemo) mobile device.

PocketSUMMIT is a continuous ASR system for mobile devices developed by the MIT Computer Science and Artificial Intelligence Laboratory [17]. The system is based on SUMMIT, a desktop speech recognition system [44]. Optimizations for this system include the use of only landmark features, an 8:1 compression for Gaussian parameter quantization, and finite-state transducer (FST) size reduction by pruning and simplification of the decoder by a state/word graph in the first pass of search. The system was tested on a Windows CE/mobile device with a 408MHz AMR-based processor and 64MB of RAM. The word error rate ranged from 9.5% to 10.7% using a 2000-word weather information domain.

2.6 The Borboleta telehealth system

The Borboleta telehealth system is a mobile application to collect medical information during residential visits in the homecare program provided by the University of São Paulo Primary Healthcare Center [14]. The Borboleta system is composed of two modules, a mobile component to support the visits and *SAGUISaúde*, the central application that manages and provides the medical information in the health center. Figure 10 shows the overall architecture of the Borboleta system. To conduct the medical visits, the mobile device is synchronized with the server by downloading the medical information for the scheduled visits. The visit is conducted and the data is saved in the mobile devices. The collected information is then uploaded to the server and can be accessed using a Web browser.

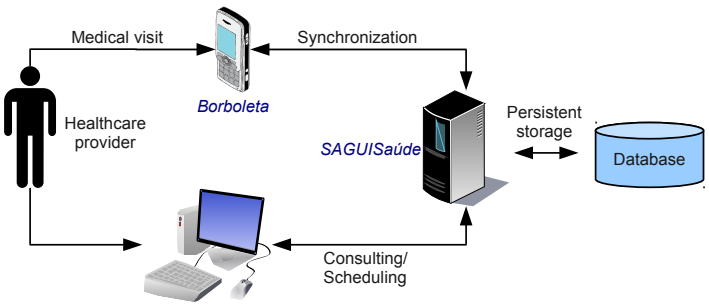


Figure 10: *Borboleta - Overall architecture*

Because of the complex and varied medical information that must be collected, the Borboleta mobile module offers different input screens as well as defined procedures to complete a task. The main features of the system are collection of patient and caregiver data, visit scheduling, and collection of information about the residential visit, patient status, problems and needs. The system provides a synchronization schema using XML over an SSL connection through a wireless network. Figure 11 shows some screenshots of the complex graphical

interface of the Borboleta software.

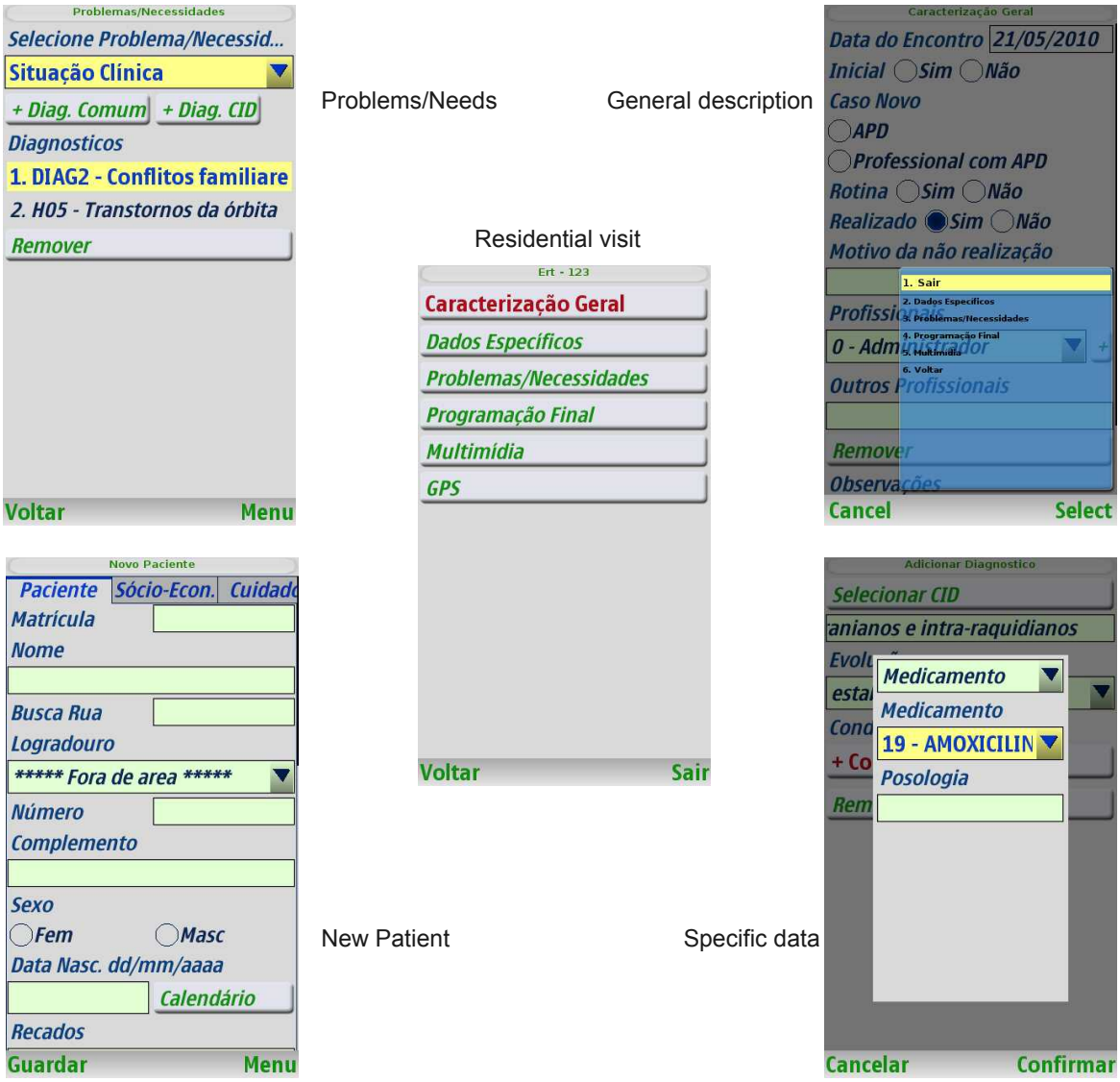


Figure 11: *Borboleta - Interfaces*

The system has been developed in Java for the Java ME specification in order to guarantee the highest compatibility with mobile devices. Currently the system runs on Windows Mobile, Palm OS, Android and Symbian, but new multimedia features are supported only by the Nokia devices (Symbian S60).

3 Methodology

To achieve the speech recognition navigation goal for the Borboleta system a set of activities that address each one of the specific goals has been defined. There will be a bibliography revision to establish the state-of-the-art of the field, a review of the current software, a data collection of Brazilian utterances to train and test the system, a design and development stage that will define the system architecture and integration with the Borboleta and a testing process to measure the system accuracy, its usability and acceptance level. The following subsections state the proposed activities and the workplan.

3.1 Literature review

An extensive review of the related bibliography has been carried out to get the state of the art in speech recognition, especially on mobile devices. This review includes approaches for speech recognition on mobile devices, multi-modal interfaces with voice as an input method, developed projects and previous results. This also includes a study of the speech recognition basics and development methodologies for mobile devices, including development platforms, available frameworks and signal processing techniques. Part of this review has been included in the current document as theoretical background (see Section 2).

3.2 Software review

Current software for speech recognition has presented good accuracy results in large systems. In the last decade, mobile platforms have improved their capabilities to perform sophisticated tasks such as multimedia processing. Because of these improvements, some speech recognition applications have been developed to target mobile platforms. The main goal in the software evaluation phase is to select a speech recognition system that can be integrated into the Borboleta system. The evaluation of the systems will be done as follows.

The review process will include speech recognition robustness and accuracy, software compatibility with mobile devices, software architecture, source code availability, license and development community activity. Accuracy in speech recognition is important because executed commands depend on it. Therefore, the robustness evaluation will be based on existing benchmarks related to the word error rate, response time and computer resource usage. Currently, large vocabulary automatic speech recognition systems present good word error rates because of computational capacity. Since the scope of this project is the use of speech recognition for commands on mobile devices, the vocabulary is smaller than for continuous recognition. While good performance is expected, the computational capacity and environmental factors are also expected to have an impact on the results.

Speech recognition software compatibility in mobile devices is related to the execution of the application on the device and the interaction or integration with the Borboleta system. There are assorted mobile platforms, which are developed in specific languages and with their own requirements. To find a tool that runs on any platform is difficult, so to reduce this problem, the software portability will be restricted to the mobile devices used in the project and its development language. Borboleta is developed in Java for mobile devices (Java ME) although most of the devices are also C++ enabled. Integration with Borboleta will be evaluated on two possible interaction schemes. First, speech recognition as an external application running on the hand-held that communicates with Borboleta through any communication schema, and second, an internal module managed by the main system. The Borboleta system is open source software under the BSD license and the intention is to keep the same license for additional modules. By creating an open source system, an active development community can maintain sustainable software.

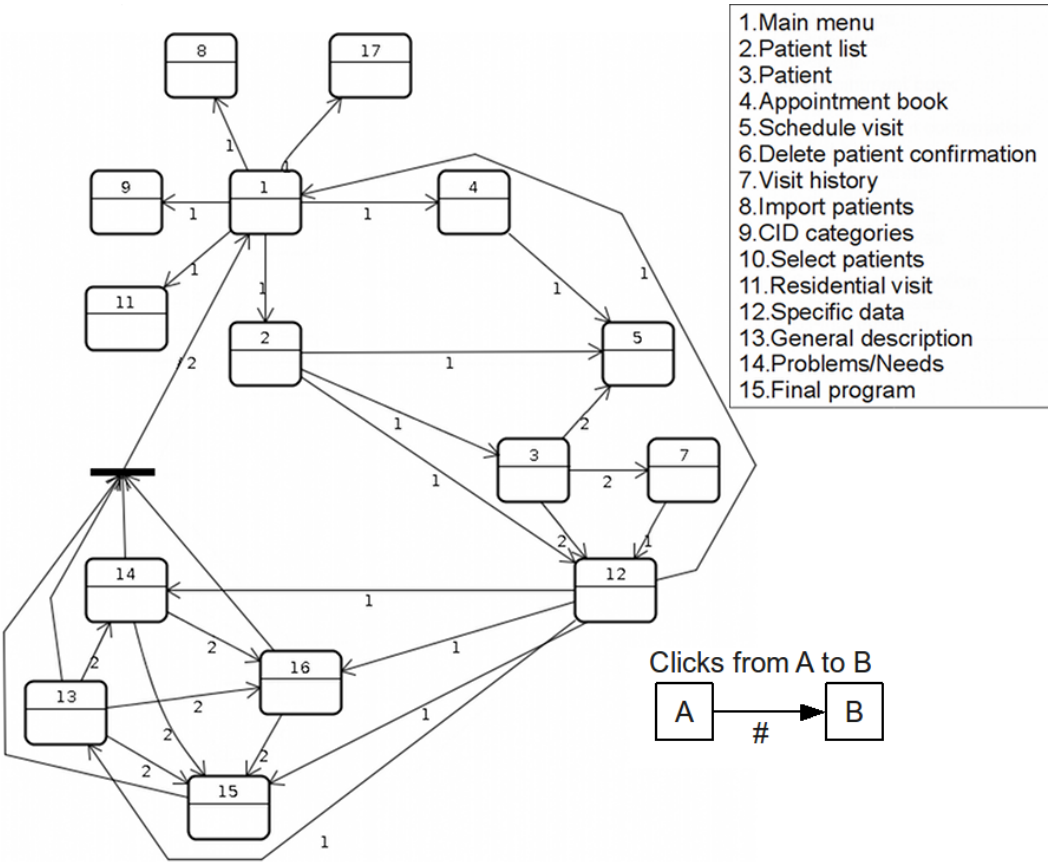


Figure 12: *Borboleta navigation diagram*

3.3 Data collection

To train the acoustic and language models and to test the system, a set of recordings and their word-level transcriptions was collected. The collection began by defining the vocabulary of voice commands that will be used during the application execution. After this definition of keywords, utterance collection and data preparation completed this phase of the project. The data collection step covered the keyword list definition, utterance collection and data preparation for the training and testing phase.

Keyword definition. Keywords are words that the system will recognize as commands. The keyword list is not composed of all the system functionalities but only the ones that need more than two user actions, such as button clicking or selecting a menu option, to be reached. Figure 12 shows the navigation diagram for the Borboleta system. This diagram defines the current paths for any function in the system. Each edge value is defined as the cost for passing through the vertex (a window), that is, how many user actions will be required to access a specific function. Like a usability metric, a function cost can be defined to determine the commands, or keywords, the system will accept. Based on the diagram, the paths that are longer than two will be part of the keyword list. For example, if the user is in the *Problems/Needs window* (box 15) and wants to go to the *CID categories* (box 9), the shortest path will be through *Main menu* (box 1), with a cost of three user actions. The fifteen commands on the keyword list are shown in Table 3.

Utterance collection. To collect data that matches the expected application environment for training the system and to get workable results in the testing phase and later in a produc-

Table 3: *Keyword list for navigation on Borboleta*

Portuguese	**English**
Menú inicial	Main menu
Dados específicos	Specific data
Caracterização geral	General description
Problemas e necessidades	Problems and needs
Programação final	Final program
Encontros realizados	Visit history
Encontro	Residential visit
Novo encontro	New residential visit
Paciente	Patient
Agenda	Appointment book
Agendar visita	Schedule visit
Catálogo CID	International Disease Catalog (IDC)
Sair	Exit
Voltar	Back
Fechar	Close

tion system, the goal in this step was to emulate most of the environmental characteristics where the system will be used. First, the recordings were collected on devices similar to the ones used by professionals during medical visits. This reduces the effects of different microphones and it will help us define the typical noise environment. Recordings were done in a reasonably quiet place, such as a room with normal environment noise and low external sounds. During the recording process, healthcare professionals held the hand-held device normally while directing their voice toward the device, and spoke naturally. Modifications in the way the user holds the device are not desirable because it affects the signal quality. At a consistent distance, the sound level can be used as an activation event for the speech recognition process. The system must recognize command utterances that are reproduced in a natural way and not just as isolated words. Figure 13 shows the correct way to hold the device when recording data or using the system.

The Borboleta healthcare team is composed of eight persons, 2 men and 6 women. If the keyword list has fifteen commands, and each team member records five sets, then 600 samples were collected to be used in training and testing phases. Samples were collected in WAV format using PCM encoding with an 11025Hz sampling rate and 16-bit precision [9]. According to Huang [20], 11kHz sampling reduces errors by about 10% over a baseline of 8kHz, which is typical for telephone speech. 16kHz sampling rates reduce the word recognition error by 10% more, but this sample rate is not available in all mobile devices so will not be used.

Data preparation. With the data collected, the next step is to prepare it for the speech recognition system. Faulty data, such as recordings with loud background noise, were removed. Then, the data was organized so that it can be accessed by the system. Most speech recognition systems require a dictionary that contains all the words the system will support, a filler file of non-speech sounds, a file path list that the system will use to find data, a file containing the available unit sounds whether words or phones, and the transcriptions. Construction of the dictionary and the phone file uses the Speech Assessment Methods Phonetic Alphabet (SAMPA) [30]. SAMPA for Portuguese was also used in the FalaBrasil project [33]. The data will then be separated into a training set to create the acoustic models and a test set for evaluating system accuracy.

Figure 13: *Data collection method*

3.4 Initial Training and Testing phase

70% of the collected data will be used to train the system, generating the acoustic models for the three different modeling approaches: isolated commands, whole-word models, and phone-based recognition. The language model will be in the form of a finite state network based on the navigation diagram.

Each recognition approach requires a different configuration file set, although they will use the same language models. Initial testing will be carried out on the speech recognition software running on a desktop computer. Running the experiments on a desktop computer will not affect the accuracy results because the training and testing data is collected with mobile devices and the software will be limited to run on mobile devices. Recognition will be performed on the remaining 30% of the collected data, using desktop simulations for preliminary results. The preliminary testing will be used to select the modeling approach used in the final system.

3.5 Software design and development

Borboleta software is developed in Java for mobile devices (Java ME). This platform guarantees software compatibility across a broad range of handhelds. However, there are different Java Virtual Machine (JVM) implementations, each one compatible with a specific mobile platform. Currently, specific implementations do not have the same functionalities. For media processing, there are only a few implementations that offer media capture functionality; thus, not all devices are capable of performing speech recognition. C++ is available in many devices and has the advantage that most mobile platforms have been developed in this language, so access to resources is faster. Speech recognition algorithms could run faster than on a virtual machine. However C++ implementations are commonly platform-dependent, reducing the application compatibility range.

The software design and development phase will be approached from two sides, porting the speech recognition software to mobile devices, and integrating the software with Borboleta. Porting the software will be carried out by evaluating the language compatibility, software robustness and accuracy. Language compatibility is related to implementation options for the different speech recognition phases, like media capturing, signal processing and word searching. Evaluating the software robustness and accuracy allows assessment of the application development level and comparisons with previous results on mobile devices. Software integration with Borboleta refers to the communication schema between the two applications. The communication will depend directly on the selected platform. Java implementation will

allow for easier integration, because Borboleta is developed in this language. For a C++ version, an interoperation schema must be defined. Interoperation approaches between Java and C++ have been successful on desktop computers, but for mobile devices, the platform limitations reduce this intercommunication. Approaches like inter-process communication via sockets for mobile devices will be evaluated in this stage.

3.6 Evaluation

Software validation involves two relevant aspects: system performance and accuracy, and usability evaluation. System performance and accuracy is related to the system's ability to identify utterances correctly with an acceptable time range. Usability evaluates how software functionalities are accessed by the users, the complexity of the system and user acceptance.

Typically, an automatic speech recognition system (ASR) is evaluated using the word error rate (WER). WER measures the system accuracy by contrasting recognized utterances with real transcriptions [5]. WER will be used to compare isolated word, phone-based and whole-word modeling approaches. Recognition performance will also include recognition speed and resource usage, an important aspect in mobile environments with few processing and storage resources, and battery-life dependency. The results from the performance tests will help us to decide if it is feasible to have a speech recognition system incorporated with Borboleta on the mobile device The test set, based on the keyword data, will be available to perform experiments to compare the word error rates for the different configurations. A final accuracy evaluation will be executed in a real scenario with real users, using the range of mobile devices used in the Borboleta project.

For usability evaluation, a typical usability testing approach will be used [41]. The approach aims to insure consistency in navigation schemes, system responses, and system vocabulary [41]. A preliminary navigation evaluation will be performed before integration of ASR with the system. The same navigation evaluation using speech recognition will also be executed. Both results will be compared to get usability results and to evaluate the usefulness and usability of including ASR in Borboleta. In the usability evaluation for speech recognition a set of tasks are executed by the user. After each task, the users complete an after scenario questionnaire (ASQ). At the end of the test, each users completes a post-study system usability questionary (PSSUQ) [41]. Questions may include statements like "The system was easy to use" and "I successfully completed my task" evaluated on a 5-point Likert scale [29].

During the design and development stages, tests of accuracy and usability will be executed to get feedback from the users to improve the system before the final version.

3.7 Workplan

Project activities are being carried out based on the timetable scheduling shown in Table 4. The literature review has been included in Section 2 of this document. Software evaluation has resulted in a preliminary selection of Sphinx over HTK and Julius. Currently, a deep study of Sphinx code is being carried out to define the software porting and integration tasks necessary to incorporate it with Borboleta. In the data collection phase, 600 command samples have been collected from 8 caregivers of the Borboleta team. Currently, the initial training and testing phase is being executed. A study of Symbian development for C++ is being carried out to improve the skills in this platform. A first version running on mobile device should be complete by the end of September. Accuracy and usability tests will be carried out regularly after an initial version is released. Final testing will be completed in December. During this time, software modifications will likely be done to optimize the accuracy and integration with Borboleta. Thesis preparation will be done from October to end of January, as well as the preparation of a paper to publish the results of the completed

work. Interspeech 2011 has been identified as a potential conference for a paper submission. The thesis defense is scheduled for the end of February and correction in March.

Table 4: *Timetable*

Activity	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Fev	Mar
Literature Review													
Software evaluation													
Keyword definition													
Utterance collection													
Data preparation													
Training and Testing													
Software Design													
Software Development													
Accuracy tests													
Usability tests													
Thesis writing													
Thesis preparation & defense													
Corrections													
Paper writing													

Completed task
Current task
Incomplete task

4 Preliminary work and results

During project development, we have obtained some important results for the project itself as well as related projects. The state of the art in speech recognition for mobile devices was defined, as well as the current state of speech recognition for Brazilian Portuguese. During keyword definition, the Borboleta navigation diagram was created. This diagram allowed the project group to identify usability problems of earlier Borboleta versions, such as ambiguous menu options, duplicated function paths and unnecessary steps. Also, a cost diagram (Figure 12) was extracted from the navigation diagram with the aim of evaluating user actions that could be performed by using speech commands. Software for collecting utterance samples for speech recognition was developed. This software creates data sets from sentence files or free time recordings in WAV format. The software is Java ME compatible, giving the advantage of recording with mobile devices and performing data collection in unrestricted environments using the mobile devices. 600 command samples were collected for training and testing acoustic models. After preparation the data set was reduced to 525 samples because of high background noise or incomplete sentences. Additionally, the infrastructure is being prepared for future work in FLOSS in large vocabulary speech recognition for Brazilian Portuguese.

5 Significance/Implications

This research will contribute to the Borboleta project in its current work on mobile devices by extending the project to include multimodal interfaces. The goal is to improve the user experience of the mobile application. Usability improvements of the Borboleta system would result in better patient care during medical visits because of the reduced need for software attention. The current project will lay the foundations for subsequent research in speech recognition for Brazilian Portuguese. Additional contributions will include automatic speech recognition performance on mobile devices and usability results for multi modal interfaces on mobile devices. This project will also bring advances in speech recognition closer to society in a developing country through new interaction paradigms.

References

[1] J. K. Baker. Stochastic modeling for automatic speech understanding. *D. R. Reddy, ed., Speech Recognition: Academic Press, New York*, pages 521–542, 1975.

[2] Rebecca Anne Bates. *Speaker dynamics as a source of pronunciation variability for continuous speech recognition models.* PhD thesis, University of Washington, 2003.

[3] Cambridge University Engineering Department. HTK - Hidden Markov Model Toolkit - http://htk.eng.cam.ac.uk, April 2010.

[4] Canalys. Worlwide smart phone market Q1 2010 - http://www.canalys.com/pr/2010/r2010051.html, May 2010.

[5] Chia-Ping Chen. *Noise Robustness in Automatic Speech Recognition.* PhD thesis, University of Washington, 2004.

[6] Y. L. Chow, M. O. Dunham, O. A. Kimball, M. A. Krasner, G. F. Kubala, J. Makhoul, P. J. Price, S. Roucos, and R. M. Schwartz. BYBLOS: the BBN continuous speech recognition system. pages 596–599, 1990.

[7] CMU. CMU Sphinx Wiki - http://cmusphinx.sourceforge.net/wiki, May 7 2010.

[8] M. Cohen, H. Murveit, J. Bernstein, P. Price, and M. Weintraub. The Decipher Speech Recognition System. In *International Conference on Acoustics, Speech, and Signal Processing*, pages 77 –80 vol.1, apr 1990.

[9] IBM Corporation and Microsoft Corporation. Multimedia programming interface and data specifications 1.0, August 1991.

[10] Rafael Correia, Arlindo Flávio da Conceição, Fabio Kon, Rubens Kon, and José Ricardo Brand ao. Sistema Móvel Multimídia de Código Aberto para Atenção Primária de Saúde com Visitas Domiciliares. *10th Workshop on Free Software*, 2009.

[11] Rafael Correia, Fabio Kon, and Rubens Kon. Borboleta: A mobile telehealth system for primary homecare. *23rd ACM Symposium on Applied Computing*, 2008.

[12] CSLU and OGI. 22 language v1.3 - http://cslu.cse.ogi.edu/corpora/22lang/, April 2010.

[13] CSLU and UFRGS. The Spoltech Brazilian Portuguese v1.0 - http://cslu.cse.ogi.edu/corpora/spoltech, April 2010.

[14] Helves Domingues, Rafael José Peres Correia, Fabio Kon, Eduardo Ferreira, and Rubens Kon. Análise e Modelagem Conceitual de um Sistema de Prontuário Eletrônico para Centros de Saúde. *WIM - Workshop de Informática Médica*, 2008.

[15] Homer Dudley and T. H. Tarnoczy. The Speaking Machine of Wolfgang von Kempelen. *The Journal of the Acoustical Society of America*, 22(2):151–166, 1950.

[16] G. Evermann, H.Y. Chan, M.J.F. Gales, B. Jia, D. Mrva, P.C. Woodland, and K. Yu. Training LVCSR systems on thousands of hours of data. *ICASSP*, I:209–212, 2005.

[17] I.L. Hetherington. PocketSUMMIT: small footprint continuous speech recognition. *INTERSPEECH 2007*, 2007.

[18] John Holmes and Wendy Holmes. *Speech Synthesis and Recognition.* Taylor & Francis, Inc., Bristol, PA, USA, 2002.

[19] J. Huang, V. Goel, R. A. Gopinath, B. Kingsbury, P. Olsen, and K. Visweswariah. Large Vocabulary Conversational Speech Recognition with the Extended Maximum Likelihood Linear Transformation (EMLLT) Model. *ICSLP*, 2002.

[20] Xuedong Huang, Alex Acero, and Hsiao-Wuen Hon. *Spoken Language Processing: A Guide to Theory, Algorithm, and System Development.* Prentice Hall PTR, Upper Saddle River, NJ, USA, 2001.

[21] Xuedong Huang, Fileno Alleva, Mei-Yuh Hwang, and Ronald Rosenfeld. An overview of the SPHINX-II speech recognition system. In *HLT '93: Proceedings of the workshop on Human Language Technology*, pages 81–86, Morristown, NJ, USA, 1993.

[22] D. Huggins-Daines. PocketSphinx: a free real-time continuous speech recognition system for hand-held devices. *Proc. ICASSP 2006*, pages 185–188, 2006.

[23] IBM. Via Voice - http://www-01.ibm.com/software/pervasive/embedded_viavoice, April 2010.

[24] Dale Isaacs and Daniel J. Mashao. A Tutorial on Distributed Speech Recognition for Wireless Mobile Devices. *Southern African Telecommunications and Applications Conference (SATNAC 2007)*, pages 9–13, 2007.

[25] B.H. Juang and Lawrence R. Rabiner. Automatic Speech Recognition - A Brief History of the Technology Development. *Elsevier Encyclopedia of Language and Linguistics, 2nd Edition*, 2005.

[26] Tomonari Kamba, Shawn A. Elson, Terry Harpold, Tim Stamper, and Piyawadee Sukaviriya. Using small screen space more efficiently. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 383–390, New York, NY, USA, 1996. ACM.

[27] H. Kokubo, N. Hataoka, A. Lee, T. Kawahara, and K. Shikano. Embedded Julius: Continuous Speech Recognition Software for Microprocessor. In *Multimedia Signal Processing, 2006 IEEE 8th Workshop on*, pages 378 –381, October 2006.

[28] Kai-Fu Lee, Hsiao-Wuen Hon, Sanjoy Mahajan Hwang, and Raj Reddy. The Sphinx Speech Recognition system. In *HLT '89: Proceedings of the workshop on Speech and Natural Language*, pages 125–130, Morristown, NJ, USA, 1989.

[29] Rensis Likert. A technique for the measurement of attitudes. *Archives of Psychology*, 140:1–55, 1932.

[30] University College London. Speech Assessment Methods Phonetic Alphabet, http://www.phon.ucl.ac.uk/home/sampa, April 2010.

[31] T. B. Martin, A. L. Nelson, and H.J. Zadell. Speech recognition by feature abstraction techniques. Technical report, Air Force Avionics Lab, 1964.

[32] Nagoya Institute of Technology. Open-Source Large Vocabulary CSR Engine Julius - http://julius.sourceforge.jp, April 2010.

[33] Nelson Neto, Patrick Silva, Aldebaro Klautau, and Andre Adami. Spoltech and OGI-22 Baseline Systems for Speech Recognition in Brazilian Portuguese. *International Conference on Computational Processing of Portuguese Language*, 5190/2008:256–259, 2008.

[34] Nuance. Dragon NaturallySpeaking Solutions - http://www.nuance.com/naturallyspeaking, April 2010.

[35] Sharon Oviatt and Rebecca Lunsford. Multimodal Interfaces for Cell Phones and Mobile Technology. *International Journal of Speech Techonology*, 8(2):127–132, June 2005.

[36] P. Placeway, S. Chen, M. Eskenazi, U. Jain, V. Parikh, B. Raj, M. Ravishankar, R. Rosenfeld, K. Seymore, M. Siegler, R. Stern, and E. Thayer. The 1996 Hub-4 Sphinx-3 System. In *Proc. of DARPA Speech Recognition Workshop*, 1996.

[37] Lawrence Rabiner and Biing-Hwang Juang. *Fundamentals of speech recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[38] Steve Renals. Speech Recognition. COM3140 (Speech Technology) Course Notes, Department of Computer Science, University of Sheffield, 1998.

[39] T. Sakai and S. Doshita. Phonetic typewriter. *The Journal of the Acoustical Society of America*, 33(11):1664–1664, 1961.

[40] Alexander Schmitt, Dmitry Zaykovskiy, and Wolfgang Minker. Speech recognition for mobile devices. *Int J Speech Technol*, 11:63–72, 2008.

[41] Helen Sharp, Yvonne Rogers, and Jenny Preece. *Interaction Design: Beyond Human-Computer Interaction*. Wiley, 2nd edition, March 2007.

[42] Patrick Silva, Pedro Batista, Nelson Neto, and Aldebaro Klautau. *An Open-Source Speech Recognizer for Brazilian Portuguese with a Windows Programming Interface*, chapter Computational Processing of the Portuguese Language, pages 128–131. Springer Berlin / Heidelberg, 2010.

[43] Oracle Sun. Java micro edition - http://java.sun.com/javame, May 2010.

[44] Spoken Language Systems. SUMMIT - http://groups.csail.mit.edu/sls/technologies/asr.shtml , May 2010.

[45] Karlsruhe University. GlobalPhone Portuguese (Brazilian) - http://www.elda.org/catalogue/en/speech/s0201.html, May 2010.

[46] Keith Vertanen and Per Ola Kristensson. Parakeet: a continuous speech recognition system for mobile touch-screen devices. In *IUI '09: Proceedings of the 13th international conference on Intelligent user interfaces*, pages 237–246, New York, NY, USA, 2009. ACM.

[47] VoxForge. VoxForge - http://www.voxforge.org, April 2010.

[48] W. Walker, P. Lamere, P. Kwok, and B. Raj. Sphinx-4: A flexible open source framework for speech recognition. Technical report, Sun Microsystems Laboratories, 2004.

[49] Nokia Forum Wiki. S60 3rd edition: Application development - http://wiki.forum.nokia.com/index.php/s60_3rd_edition:_application_development, May 2010.

[50] Mark Wilcox. *Porting to the Symbian Platform*. Wiley, 2009.

[51] Steve Young, Gunnar Evermann, Mark Gales, Thomas Hain, Dan Kershaw, Xunying Liu, Gareth Moore, Julian Odell, Dave Ollason, Dan Povey, Valtcho Valtchev, and Phil Woodland. *The HTK book for HTK V2.0*. Cambridge University Press, 1995.

[52] Dmitry Zaykovskiy. Survey of the speech recognition techniques for mobile devices. *SPECOM'200*, pages 88–93, 2006.

[53] Dmitry Zaykovskiy and Alexander Schmitt. Java (J2ME) Front-End for Distributed Speech Recognition. *21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07)*, 2007.