NORTH-WEST UNIVERSITY
YUNIBESITI YA BOKONE-BOPHIRIMA
NOORDWES-UNIVERSITEIT

Effective automatic speech recognition data collection for under-resourced languages

NICOLAAS JOHANNES DE VRIES

Effective automatic speech recognition data collection for under-resourced languages

by

N.J. de Vries

Dissertation submitted in partial fulfilment of the requirements for the degree

Master of Engineering (Electrical and Electronic)

at the

North-West University, South Africa

Supervisor: Prof. M.H. Davel

November 2011

ABSTRACT

As building transcribed speech corpora for under-resourced languages plays a pivotal role in developing automatic speech recognition (ASR) technologies for such languages, a key step in developing these technologies is the effective collection of ASR data, consisting of transcribed audio and associated meta data.

The problem is that no suitable tool currently exists for effectively collecting ASR data for such languages. The specific context and requirements for effectively collecting ASR data for under-resourced languages, render all currently known solutions unsuitable for such a task. Such requirements include portability, Internet independence and an open-source code-base.

This work documents the development of such a tool, called *Woefzela*, from the determination of the requirements necessary for effective data collection in this context, to the verification and validation of its functionality. The study demonstrates the effectiveness of using smartphones without any Internet connectivity for ASR data collection for under-resourced languages. It introduces a semi-real-time quality control philosophy which increases the amount of usable ASR data collected from speakers.

Woefzela was developed for the Android Operating System, and is freely available for use on Android smartphones, with its source code also being made available. A total of more than 790 hours of ASR data for the eleven official languages of South Africa have been successfully collected with Woefzela.

As part of this study a benchmark for the performance of a new National Centre for Human Language Technology (NCHLT) English corpus was established.

Keywords: under-resourced languages, new languages, speech resources, ASR corpora, automatic speech recognition, developing world, speech data collection, spoken language resources, Android, NCHLT.

ACKNOWLEDGEMENTS

I would like to thank:

- Marelie Davel, for the many hours of feedback and guidance she has provided.
- Etienne Barnard, for his timely inputs.
- Jaco Badenhorst, for providing the quality control criteria and example algorithms in Python.
- Stjepan Rajko, for providing the initial version of the WAVE-recording class.
- My lovely wife, Tamryn, for her continued love and support.
- And further, all the CSIR-Meraka HLT members and staff for their inputs.

"For from him and through him and to him are all things. To him be glory forever. Amen."
(Rom. 11:36, ESV)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE

INTRODUCTION

The most natural mode of human communication is speech – as evidenced by illiterate people conversing fluently in their mother-tongues. With respect to developing human-machine interfaces, science and technology has come a long way in the past few decades with varying degrees of success [1–3]. In the developed world, speech technology has a well established track record of usefulness with applications such as call routing, directory services, dictation and travel information. These applications are saving large companies and small businesses alike significant amounts of money and even generates annual revenues in the billions of dollars for others [1].

With the rapid increase of the number of mobile phones worldwide, the input modality of speech has become increasingly important. Speaking is much faster and more natural than keyboard entry, especially for languages such as Cantonese and Japanese with large character sets [4–8]. For human-robotic interfaces, speech input will become a necessity as the tasks that robots can perform become increasingly complex.

In the developing world, this picture is similar yet different. With more recent applications such as health information services [9, 10], education [5], information access [11], agriculture [9] and government services [12], speech technologies are slowly demonstrating some of the impact that it could have in these environments, in for example breaking down barriers of inequality of information accessibility [13] and generating revenue for future economic sustainability [5]. But in order for automatic speech recognition (ASR) technology to impact the developing world more significantly, a number of "hurdles" must first be overcome [14, 15]. One of these hurdles is the collection, or expansion, of ASR corpora.

While under-resourced languages may be found in any geographical area, they are typically found in developing world contexts. Apart from collecting data for under-resourced languages, techniques such as language adaptive acoustic modelling as discussed in Schultz *et al*. [16] may also be used

when no, or limited data of related languages exists, or when language independent models are available. Techniques such as *cross-language transfer* (no training data used for target language), *language adaption* (limited target language training data used for adapting acoustic models), *bootstrapping* (initialising acoustic models from a different language) [17], *data pooling* (directly combining data from different languages) [18], and *harvesting* audio and transcripts from the Internet [19] or broadcast news [20], may also be considered. However, it is often found that a point is reached where the only alternative is collecting more, or at least some well-matched language-specific data [4]. This is apart from the motivation that larger amounts of ASR data necessarily leads to better recognition performance [8, 21].

This study focusses on effectively collecting ASR data for under-resourced languages, in order to both enable and stimulate the development and expansion of ASR technologies for these languages. It was performed in a South African context with the immediate impetus for this work provided by a National Centre for Human Language Technology project, seeking to collect broadband speech corpora for all the eleven official languages of the country.

1.1 BACKGROUND

1.1.1 ASR RESOURCE SCARCITY

Only about 20 to 30 of the world's 6,900 languages have significant quantities of digitised data, and much of this data is only in textual form [22, 23]. With under-resourced languages residing primarily in developing world contexts, various additional data collection challenges exist which may exacerbate the already difficult task of collecting ASR data. Some specific challenges will be discussed in Section 1.1.3.

In South Africa, a recent Human Language Technology (HLT) audit by Sharma *et al*. [24] in 2009, indicated that for developing speaker-independent ASR systems, orthographically transcribed speech corpora should be one of the highest priority items on the HLT agenda for most of South Africa's eleven official languages, in order for speech technologies to advance and for a thriving HLT industry to emerge.

As part of a large government-funded three year project, called *Lwazi*, conducted from 2006 to 2009, an ASR corpus containing all eleven official languages of South Africa was developed to demonstrate the use of speech technology for information service delivery [25]. Table 1.1 shows this publicly available corpus containing all South Africa's official languages, with the respective columns showing (i) the official languages of South Africa, (ii) their ISO 639-3:2007 language codes, (iii) estimated number of home language speakers in South Africa, (iv) language family (SB indicates Southern Bantu), and (v) the size of the Lwazi ASR corpus in minutes. The total size of the N-TIMIT corpus is provided in the same table for comparison.

Applications requiring access of information over telephone channels differ significantly in terms of the data collected from applications in which wide-band data need to be transcribed. For example,

when attempting to effectively transcribe wide-band broadcast news, broadband ASR data would be required since it is commonly known that band-limited data would negatively affect the recognition accuracy when attempting to recognise wide-band speech.

The Lwazi corpus was recorded by users calling from 'normal' telephone channels, and answering specific questions posed by the system. The data collected in this way is thus well-matched for ASR applications intending to use telephone channels to access information, but less suitable for transcribing wide-band data due to the band-width constraints imposed by telephone channels. When ASR data needs to be collected for a wider range of applications, both channel mismatch as well as bandwidth mismatch need to be considered. Ideally, data collection should match both the bandwidth as well as the channels that will be used in the target application for the ASR technology, but when limited resources are available for data collection, a compromise has to be made.

In the case of channel mismatch, a number of techniques, with varying degrees of success exist to adapt data between the data used to develop ASR systems and the data needing to be transcribed [26]. For bandwidth mismatch, data collected at higher sample frequencies could be sub-sampled to match the required bandwidth of the target application.

In order to expand the availability of ASR corpora for a wide range of potential applications in South Africa, the Department of Arts and Culture of the Republic of South Africa in 2009 commissioned the collection of *broadband* speech corpora for all eleven official South African languages over the following three years, since prior to this time such broadband corpora did not exist (See Section 1.1.4 for more detail). This would increase the availability of larger corpora for each language, as well as address a need for having broadband corpora.

Table 1.1: *The Lwazi corpus from Barnard* et al. *[25]. Used with permission.*

Language	**Code**	**No. speakers (million)**	**Language family**	**Total minutes**
isiZulu	Zul	10.7	SB:Nguni	525
isiXhosa	Xho	7.9	SB:Nguni	470
Afrikaans	Afr	6.0	Germanic	213
Sepedi	Nso	4.2	SB:Sotho-Tswana	394
Setswana	Tsn	3.7	SB:Sotho-Tswana	379
Sesotho	Sot	3.6	SB:Sotho-Tswana	387
SA English	Eng	3.6	Germanic	304
Xitsonga	Tso	2.0	SB:Tswa-Ronga	378
siSwati	Ssw	1.2	SB:Nguni	603
Tshivenda	Ven	1.0	SB:Venda	354
isiNdebele	Nbl	0.7	SB:Nguni	564
Eng (N-TIMIT)				315

1.1.2 TYPES OF ASR RESOURCES

ASR resources may consist of a number of components, depending on the purpose for which the resources are intended. In general, ASR resources consist of a pronunciation lexicon, a phoneme set and a set of audio data with associated orthographic transcriptions [27]. Each of these are briefly described:

Pronunciation lexicons, also called pronunciation dictionaries provide a mapping from words to sound-units called phonemes. By providing such a mapping during the training of acoustic models, sound-units expected to be found in the audio data could be modelled. When decoding audio data using acoustic models, probable phone sequences could be mapped back to words using the same pronunciation dictionary. These dictionaries could be developed by non-experts by using a bootstrapping approach [28]. A **phoneme set** is a set of orthographic symbols used to represent semantically distinct sounds in a specific language. This orthographic representation may take any form, for example, a single or set of ASCII-characters, as long as a one-to-one mapping exists between these representations and the abstract sound-units (*phonemes*). **Audio data** is the actual digitised speech waveforms of the recorded audio signals, while **orthographic transcriptions** are the associated textual representations or transcriptions of the audio data. A further data set typically part of ASR resources, is the **meta data** associated with each speaker, such as age and gender, to facilitate the development of, for example, gender-dependent acoustic models.

Depending on the intended use of an ASR corpus, various other information sources may also be included. For example, if research or recognition is intended to be based also on spatial and temporal activity in the cerebral cortex, then information such as fMRI and EEG data would also be included in the ASR resources compiled [29].

In this study, the primary concern is only with collecting audio data with the associated orthographic transcriptions as well as meta data for each speaker, and not with any of the other information sources typically (or less typically) found in an ASR corpus.

1.1.3 EFFECTIVELY COLLECTING ASR DATA FOR UNDER-RESOURCED LANGUAGES

The challenges of collecting ASR data for under-resourced languages are numerous. With most under-resourced languages residing in developing world areas, specific requirements for effectively collecting ASR data for such languages exists.

1.1.3.1 PORTABILITY

A primary requirement for effective collection of ASR data in developing world areas, is the key aspect of portability. Mother-tongue speakers of under-resourced languages are often either residing in more rural areas, or small communities of speakers distributed over large geographic areas. In conducting data collection campaigns for these languages, transporting people to stationary recording

environments, or semi-portable equipment to remote locations, is often unfeasible [30].

A further advantage of a more distributed approach to ASR data collection offered through such portability, is the parallel nature in which these campaigns could be conducted. By employing highly portable equipment for such campaigns, more than one speaker's audio data may be recorded at a time, and in more than one geographic location. The only upper limit of parallelisation in this regard would be specific equipment budget constraints, and any associated manpower constraints. In contrast, renting professional or semi-professional studio time, or constructing such studios, only allows for very limited parallelisation. Compounded through the dynamic recruitment nature typical of under-resourced language data collection campaigns, these fixed-location approaches become impractical.

1.1.3.2 INTERNET INDEPENDENCE

If Internet connectivity was assumed, a number of opportunities would be available for ASR data collection, such as, downloading textual corpora for recording, uploading recorded data to a central server, and even performing some form of semi-real-time quality control of the audio data on back-end servers. But, such an assumption is simply not valid for the vast majority of developing world regions in which most under-resourced languages reside [5, 31].

While some developing regions have cheap, reliable Internet connectivity, this is not generally the case for most developing regions. Such connectivity may be non-existent, highly congested, or provided on an ad-hoc basis. For example, the DakNet project [31] uses a wireless router mounted on top of a bus to 'transport' email between villages and an Internet connection in a nearby city and thus to the rest of the world. This bus is effectively acting as a "digital postman" collecting and delivering 'mail'.

Wireless connectivity to the Internet, while generally available in large parts of South Africa, can unfortunately not yet be assumed in certain of the more rural areas. Even if this connectivity existed in these areas, the cost of accessing such services are at times prohibitive. In the bigger cities some open wireless access points exists, but with the limited bandwidth available through these access points, field workers spent many hours in uploading data collected for similar data collection campaigns. When private access points were used, fieldworkers had to fork out large amounts of money for uploading the data recorded.

In conclusion, in the instances that Internet connectivity is available in these developing regions, cost, throughput, latency and stability may be hugely prohibitive factors for large scale data collection campaigns, especially on limited budgets.

1.1.3.3 OPEN-SOURCE SOFTWARE

One of the best arguments for open-source software is the flexibility and opportunity for customisation that it provides for diverse contexts [5]. For ASR data collection for under-resourced languages, this is a particularly important aspect as the unique needs of different languages, *locales* and recording campaigns are simply too diverse to be envisaged *a priori* for all contexts.

Also, in projects with highly constrained budgets, free or low cost software may not only provide the necessary impetus for ASR data collection projects, but may be the only means of completing such projects within budget requirements.

Given the above requirements, an obvious solution that may come to mind is that of using portable digital recorders (similar to the now discontinued Sony MD Walkmans [32]), capturing audio data in a lossless format. These devices are indeed highly portable and have no reliance on the Internet, but introduces a number of complicating factors in both presenting and controlling randomised prompt material, as well as additional post-processing required in associating recorded audio files with transcriptions and meta data for each speaker and session [15].

In conclusion, it is clear that the primary requirements for collecting ASR data effectively for under-resourced languages, are the key aspects of (i) portability of such a tool, (ii) a total independence from any Internet connectivity, and (iii) the flexibility and customisability that open-source software provides.

1.1.4 THE NCHLT DATA COLLECTION PROJECT

One of the projects under the auspices of the National Centre for Human Language Technology (NCHLT), funded by the South African Department of Arts and Culture, set out to collect 50-60 hours of broadband speech data for each of the eleven official languages in South Africa, spanning six of the nine provinces. This forms part of an initiative to encourage the development of speech technologies in all of the official languages.

In order to develop a balanced corpus, two hundred speakers (100 male and 100 female) of each of the eleven official languages are to be recorded, with around 500 utterances per speaker. The resulting ASR corpora will consist of more than 1.1 million utterances from more than 2,000 individuals; with most languages still considered under-resourced. As part of this project, tools also had to be developed to facilitate the collection and processing of these corpora.

1.2 PROBLEM STATEMENT AND OBJECTIVES

As building transcribed speech corpora for under-resourced languages plays a pivotal role in developing ASR technologies for such languages, a key step in developing these technologies is the effective collection of ASR data, consisting of transcribed audio and associated meta data.

The primary problem is that no suitable tool currently exists for effectively collecting ASR data for such languages. The specific context and requirements in effectively collecting data for under-resourced languages, as described above, renders all currently known solutions unsuitable for such a task.

As a review of relevant literature in Chapter 2 will indicate, the challenge remains to develop a tool that will enable the effective collection of ASR data for under-resourced languages, keeping the context and unique requirements of portability, Internet independence and the open-source nature, in

mind.

This work documents the development of such a tool, called *Woefzela*, from initially determining further requirements of effective data collection in this context, to the verification and validation of its functionality and initial intent. The *objectives* of this study can thus be stated as:

- Developing an open-source mobile data collection tool for effective data collection and annotation for under-resourced languages.
- Verifying that the specifications of this tool have been achieved.
- Validating the resulting data produced by this tool.

The following section will provide the scope within which this project was conducted.

1.3 SCOPE

This project aims to develop an effective mobile data collection tool that will be useful in developing world contexts for collecting new or additional resources for ASR system development; specifically for under-resourced languages. Thus, this work recognises that:

- The optimal use of any existing ASR data is especially important when dealing with under-resourced languages, but this work focusses on collecting more (in a strictly relative sense) resources for these languages.
- A general software prototyping methodology was followed during the development of this tool towards achieving the deliverables of a specific project (See Section 1.1.4). The prototype is both verified against the design specification as well as validated in terms of delivering the expected output.
- The Android Operating System was chosen as the target platform for developing this tool, among other reasons, because of its open-source nature, its rapidly growing popularity and its freely available development tools. This is to encourage future development and extension of this tool, and to facilitate cheap or free distribution to support under-resourced languages globally. Other mobile operating systems were not comprehensively considered in this decision for these reasons.
- The design and stratification of any ASR text corpus is critical for the overall performance of an ASR system; and is no simple task. This study was not involved in the text corpus design for the various languages and as such simply used the textual corpora "as-is" for data collection and system validation.
- This study limits itself to ASR corpora created with a specific speaking style, namely prompted speech, and thus does not include references to other speaking styles such as spontaneous speech, re-told stories, 'map tasks', and others [33].

1.4 ABBREVIATIONS

Table 1.2 provides a list of frequently used abbreviations throughout this document.

Table 1.2: *Abbreviations frequently used in this document.*

Abbreviation	Expansion
ASR	Automatic Speech Recognition
GUI	Graphical User Interface
HMM	Hidden Markov Model
LPCM	Linear Pulse-Code Modulation
QC	Quality Control or Quality Check
SD card	A "Secure Digital" memory card
WAVE or WAV	An uncompressed binary file format for storing audio information; originally defined by Microsoft and IBM
XML	eXtensible Markup Language

1.5 SIGNIFICANCE OF PROBLEM

By developing a relevant mobile data collection tool, meeting the requirements of portability, Internet independence, and open-source code, as discussed in Section 1.1.3, the development of ASR systems for under-resourced languages will not only become feasible, but this may also provide a much needed impetus for developing ASR technologies for these languages, where few currently exist.

Although very little direct economic value may be attached to speech technologies for some of these under-resourced languages, improved information access especially in highly inaccessible areas may impact significantly on the quality of life for many individuals where basic communication infrastructure is available.

1.6 CHAPTER OVERVIEWS

In Chapter 2 an overview regarding relevant established and emerging data collection methodologies in literature will be provided, with a specific focus on strategies and tools relevant for ASR data collection for under-resourced languages, placing the problem in the context of previous and current efforts by others.

Chapter 3 will describe the design and development of Woefzela. Chapter 4 will evaluate the conformance of Woefzela to the design requirements, while in Chapter 5 a number of ASR systems will be developed and evaluated to confirm that the data produced by Woefzela conforms to its original intent – to develop ASR systems.

Chapter 6 will conclude with a summary of the findings, conclusions and contributions that this work has made, also providing suggestions for future research.

CHAPTER TWO

ASR DATA COLLECTION STRATEGIES

2.1 INTRODUCTION

This review will highlight relevant work in recent scientific literature pertaining to data collection strategies for Automatic Speech Recognition (ASR). In order to provide additional background for this discussion, Section 2.1.1 will provide a brief overview of the aspects of purpose and quality of ASR corpora, while Section 2.1.2 will emphasise some of the specific characteristics of ASR data. Then, a review of ASR data collection strategies in literature will follow in Section 2.2, concluded by an overview of the ASR corpus development process in the last section.

2.1.1 PURPOSE AND QUALITY OF ASR CORPORA

ASR corpora are designed and developed with a specific purpose or use in mind and this purpose determines the types and quality of information required [15, 16, 34]. Two broad categories of such uses of corpora may be that of language research and technology development. Acoustic environment, bandwidth, recording channel and many other decisions should explicitly form part of the design phase for these corpora. Also, when the quality of data forming part of a corpus is evaluated, the purpose of such a corpus should be kept in mind explicitly or implicitly.

If the primary purpose of creating a corpus is indeed for general ASR research, careful thought should be given to the minimum criteria for each of these decisions [16].

2.1.2 CHARACTERISTICS OF ASR DATA

The content and character of an ASR corpus may differ significantly from that required for developing other speech technologies such as text-to-speech (TTS) synthesizers. For example, in TTS corpora additional part-of-speech tags are usually required that will aid the pre-processing of sentences for

adjusting timing, pause and emphasis when the sentence is synthesized. TTS corpora also primarily consist of a small number of speakers with larger amounts of data per speaker compared to ASR corpora intended for speaker-independent speech recognition which typically consist of a much larger number of speakers, but with less data for each speaker.

In ASR corpora the larger number of speakers is required to effectively model the variability between speakers needed for sufficiently broad speaker-independent statistical models. In TTS corpora a larger amount of data per speaker is needed to have sufficient samples of each speech sound in all the different contexts (position in word and sentence, prosodic variant etc.) for a unit-selection approach, or to have a sufficient number of samples to train the required hidden markov models (HMMs) for each of the different contexts when HMM-based synthesis is used.

Also, typical TTS data would require 'near studio' quality acoustic environments when recorded, while for ASR data it is often more important that the recording and application acoustic environments are matched as closely as possible. For example, when higher recognition accuracies for in-car environments are required, in-car recordings would typically be necessary when building ASR systems for such applications [35].

In certain cases, some or all of the ASR data may be usable in developing different speech technologies, but such data may also require extensive and costly re-annotation for a corpus to be usable [34].

2.2 ASR DATA COLLECTION STRATEGIES

Over the past few decades various strategies and processes for collecting ASR data have become well established, while other exciting new trends indicate a number of emerging strategies. Both these approaches will be discussed below, with an emphasis on strategies that are most useful for ASR data collection for under-resourced languages.

2.2.1 ESTABLISHED STRATEGIES

2.2.1.1 PROFESSIONAL STUDIO ENVIRONMENTS

Most large-scale ASR corpus creation projects, such as the TIMIT corpus [36], procure the use of a professional sound studio for recording speech samples. This has the advantage that environmental noise and disturbances can be controlled leading to reduced unwanted acoustic events and thus higher quality speech data. Such studios are also often equipped with computers that present the prompt text together with some basic instructions to the reader, and thus reduces paper noise and other message-passing techniques; but all of this comes at a price.

Generally, such studios are extremely costly to construct or very costly to rent which only well-funded projects may be able to afford. Compounding with this fact is the cost and logistics of transporting all speakers to this fixed location in order to collect voice data.

2.2.1.2 TEMPORARY STUDIO ENVIRONMENTS

Due to the mobility or location diversity required by some projects, temporary sound studios (or sound booths) are also sometimes used to record speech data [33]. These arrangements may minimise some of the drawbacks of professional studios, such as cost and location dependence, but introduce other potential problems. Temporary studios tend not to have the same quality of insulation to external noises as permanent studios, and is thus more susceptible to interruptions and other acoustical events during recording. A specific problem may be the noise introduced by computer ventilation systems that would require special attention. But, in spite of these drawbacks, this is still a viable alternative for certain speech data collection projects. The GlobalPhone corpus [37] could fall in this category.

The major disadvantage of this approach for ASR data collection for under-resourced languages, is the trade-off between the cost of such a temporary studio, the number of deployments that could be made in parallel, and the transportation of relatively bulky and fragile equipment where road-access remains a major risk to such equipment [30, 33].

2.2.1.3 TELEPHONE-BASED COLLECTION

Telephone-based recordings are often closely matched with the intended use of speech data, especially when Spoken Dialogue Systems are to be deployed on telephone networks. In certain cases, these calls originate from fixed-line telephones [9], in others from mobile phones, and yet in others even a combination of both [11]. With telephone networks, especially mobile phone networks growing rapidly in developing world regions [14], telephone-based ASR data collection is often a viable, if not preferred alternative for studio based data collection. An example of a corpus collected in this way is the Lwazi corpus [25].

Practically, several specific issues need to be carefully considered when intending to collect ASR data over telephone networks, as highlighted by De Wet *et al*. [15], such as bandwidth limitations, lack of control over the speaker's environment, handset noise, user screening (for example, first language speaking ability) and user identity verification to avoid duplication of users. Nevertheless, when these factors are carefully considered and addressed, telephone-based recording strategies are a definite option for collecting matched data for telephone-based applications.

When requiring data with a wider bandwidth, telephone-based data collection would not suffice as typical telephone networks provide limited bandwidth to allow for channel multiplexing. With the potential expansion of wideband telephone networks (50 Hz to 7 kHz) [38], such data collection may become a viable option in the future, but currently such networks are not typically available in developing world contexts.

2.2.1.4 LIVE-SERVICE COLLECTION

Collecting additional ASR data on a live recognition service such as Google's Voice Search service [1], is a well established and cost-effective means of extending the amount of available speech

data. Although the quality of such data needs to be verified prior to employing it in adapting acoustic models, to avoid deterioration of the recogniser performance, this is a powerful strategy, and a technique often utilised by commercial ASR systems.

However, when insufficient ASR data is available to train the initial acoustic models with, or when no live service to deploy such initial models exists, this alternative is not available. For example, only a few ASR systems exists for African languages, and of these, none currently provides any live services [27].

2.2.2 EMERGING STRATEGIES

2.2.2.1 WEB-BASED COLLECTION

In recent years, various web-based strategies have emerged, built around Internet infrastructure. With the rapid growth of crowdsourcing approaches to various Human Intelligence Tasks (HITs) in the domains of Natural Language Processing and other Human Language Technology tasks such as machine translation, several corpus creation strategies have also emerged. Thus, a brief digression to the use of the most well-known of such services, Amazon's Mechanical Turk, is in order.

Amazon Mechanical Turk overview

Amazon Mechanical Turk (AMT) is one of the major players in the domain of Internet crowdsourcing for HITs [39–41]. The work-flow system of this service by Amazon provides a means for *employers* to distribute requests for HITs to be performed. The *employees*, sometimes referred to as "Turkers", can find these requests on-line and elect to perform such a task, for a specific remuneration, in a given time-frame. Once this task has been completed to the satisfaction of the employer, the employee is remunerated for the task by the employer, with Amazon requesting a percentage of these earnings from the employer [39].

Such a low-cost, high-volume transaction-based service lends itself greatly to sub-tasks required for developing ASR corpora such as transcription of audio files, verification of transcriptions and even the recording of speech data.

Transcription and quality verification on the web

As a key step in the construction of an ASR corpus, transcription of large amounts of short audio files, or verification of existing transcriptions, are required. By utilising services such as AMT, developers of ASR corpora may procure these transcriptions (or verifications) at a fraction of the cost of traditional methods, and within a reasonably short turn-around time compared to designated transcribers involved in a project.

The quality of such transcriptions have motivated numerous studies in the recent past [42,43], with an excellent overview of the different approaches and issues involved, provided by Parent *et al.* [39].

In a study done by MIT [44], dynamically constraining the input of the transcriber seems to provide promising results for obtaining good quality transcriptions. Overall, the consensus seems to be that as long as the variability in the quality of work done through crowdsourcing services, such as AMT, form part of the design phase of such projects, results comparable to that of human transcribers can be achieved.

Recording audio on the web

In a recent study by McGraw *et al*. [40] the authors used AMT for both collecting speech data from users as well as transcribing this data with HITs. Inherent in this approach, however, is the variability of recording equipment attached to on-line computers, the acoustic environments in which the participants choose to record the speech, and other potential drawbacks. Nevertheless, in the developed world (and some parts of the developing world) where Internet access is readily available, this approach has definite advantages.

2.2.2.2 DATA HARVESTING

A further strategy emerging in recent years is that of data harvesting from existing sources. Harvesting audio data with associated "approximate transcriptions" from on-line sources provides opportunities especially valuable for under-resourced languages, to extend any existing corpora in a cost effective way [19]. Other sources such as broadcast news or lecture notes are also mined for such audio data, each presenting unique challenges [20, 44]

A common challenge typically found by harvesting such sources, is the transcription accuracy. Often news scripts or lecture notes existed prior to audio recording, leading to varying degrees of accuracy when recording such read or retold texts. Recent techniques have shown very promising results in processing these inaccurate transcriptions, but further work still remain [19].

In conclusion of this sub-section, although these well established methods of ASR data collection is still much used, they lack in different aspects of cost-effectiveness, bandwidth limitations, or simply the infrastructure required for effective use in collecting ASR data specifically for under-resourced languages. The web-based data collection approaches on the other hand, have several advantages such as location independence and cost-effectiveness, but fails on the key attribute of Internet independence, a primary requirement for effective ASR data collection for under-resourced languages as seen in Section 1.1.3. Data harvesting techniques shows specific promise for under-resourced languages, but requires the data source to firstly exist, and the procurement of such data to be cost effective.

2.2.2.3 SMARTPHONE-BASED COLLECTION

In two recent publications, by Hughes *et al*. from Google Research [45] and Lane *et al*. from Carnegie Mellon University [46], *smartphones* are used to collect speech data for ASR system development or

rapid porting of ASR systems to "new" languages. Each of these publications will be discussed in further detail below, as this provides the specific context for the problem investigated in this study; with specific reference to employing these technologies for under-resourced languages.

Google Research

In Hughes *et al*., the authors had a specific need to collect ASR data for Google's "Voice Search" engine [7], and thus developed a proprietary, Android-based smartphone tool, called *DataHound*, to collect this data.

The functionality provided by DataHound is very effective for general ASR data collection. On launching the software application, some basic meta data, such as age, gender and accent of the speaker is requested, along with the recording environment, selecting from among categories such as indoors, outdoors, or other environments. The user is also required to provide a non-structured "user name", which is subsequently used to associate meta data with recorded audio for the specific user.

The general architecture of DataHound is that of a typical client-server model. The client is the application that runs on an Android smartphone that presents prompts to the user and records the spoken prompts directly through the smartphone's microphone and not via the band-limited mobile phone channel. The server-side stores the textual prompts ready for download to a device and is responsible for controlling the uploading of recorded audio data upon the client's request. Although Internet connectivity is not a necessity *during* recording, at some stage a wireless Internet connection is required, firstly to download a new set of textual prompts, and secondly to upload recorded audio files and meta data.

With DataHound, textual corpora consisting of phrases needing to be read, can only be downloaded through a wireless connection to the Internet. This does have an advantage of easier manipulation of these textual corpora from the server-side, for example, when prompts need to be removed from the corpus due to words being offensive, but has the disadvantage that such Internet connectivity is needed in order to obtain any prompts and dispatch any recorded data.

In the discussion, Hughes *et al*. also points out that speakers do not always read the requested prompts accurately, causing each transcription to be an approximation of what was said. Upon initial investigation, Hughes *et al*. concludes that only about 10% of utterances are not correctly represented by their associated transcriptions, 2% of which could be detected automatically; while human transcribers on average have an error rate that is worse than 10%. This lead to their assumption that no special efforts need to be made to avoid these reading inaccuracies during the recording process. However, in their final analysis, they discovered that a number of recording sessions suffered from poor signal-to-noise ratios or contained systematic errors made by the users, which suggested further work necessary in this regard.

In a developing world context where literacy rates are generally much lower, this observation by Hughes *et al*. might become even more pertinent.

Carnegie Mellon University

Lane *et al*. focusses on a preliminary study to establish the feasibility of using smartphones to record prompts presented to an unsupervised reader in a remote location; and subsequently uploading the resulting audio with associated meta data to a central server.

Upon presenting the speaker with a minimal set of meta data to complete, the application advances to a recording stage where each utterance is presented in turn for recording or subsequent re-recording, based on the reader's judgment.

The general architecture used by Lane *et al*. is that of a custom application running on a standard *iPhone* handset that provides the direct interface with the speaker. The speaker is presented with a phrase or sentence to read from a text corpus residing locally on the device (presumably downloaded over an Internet connection prior to starting), records the utterance, and moves on to the next prompt. The user however has to hold down a push-to-talk button while speaking the prompted string, which has lead to confusion in some cases.

Once all the required prompts have been recorded directly onto the device's internal memory, i.e. without any requirement for Internet connectivity during recording, the user may opt to start the uploading process, causing the data to be uploaded to a dedicated server. Should the user wish to halt or not initiate the uploading process, he/she may choose to do so and re-initiate the upload process at a later stage. It is argued that this 'upload on request' functionality may be especially useful when Internet costs are high in the current location that the user find themselves in, since uploading may be suspended until a more affordable connection is available. Nonetheless, at some stage Internet connectivity is required.

Lane *et al*. concludes with an evaluation of the problems encountered with their smartphone data collection approach, as a lack in the quality of the resulting audio recorded, and explains that due to the remote nature of the recordings, no control could be exercised over the environment that the user chose to record in, resulting in "unsuitable" data at times. Problems with clipping of the audio signal, insufficient energy in the signal, and low signal-to-noise ratios caused by excessive background noise, resulted in such losses.

In summarising and further commenting on some of the relevant aspects of the data collection tools from Hughes *et al*. and Lane *et al*., various issues become apparent: The basic functionality required from highly portable ASR data collection tools for under-resourced languages, is available in both of these tools, since both make use of smartphone handsets. But, both tools have a certain degree of dependence on Internet connectivity, at least during certain stages of the process, an attribute that is highly undesirable when collecting ASR data in a number of developing world contexts. As also practically experienced during a data collection campaign using DataHound in South Africa, limited Internet bandwidth in some areas and the cost of uploading recordings posed serious financial and logistic problems for in-house Contractors.

A potential solution in this regard would be to provide wireless connectivity from the smartphones

to a *local* laptop computer in the vicinity of where data is being recorded, facilitating both the prompt download as well as the data upload activities. This would in some sense 'simulate' the functionality of Internet connectivity, yet be totally independent of an actual Internet connection. As a potential extension of this project, this falls outside the scope of an initial prototype, but could be considered for future work.

Further, with the proprietary nature of DataHound, and potentially a similar constraint placed on the tool used in Lane *et al*., the lack of the much needed customisability provided by open-source software, poses a serious problem when when collecting ASR data for under-resourced languages, where such flexibility is paramount; as discussed under Section 1.1.3, page 4.

2.3 ASR CORPUS DEVELOPMENT PROCESS

Several different speaking styles exist, since speakers vary the way they speak depending on the context that the word or sentence is spoken in. Two of these broad categories of speaking styles are read and elicited speech, each taking various forms of expression. This study, and thus the corpus development process described below, focusses primarily on read speech, since the reading of specific prompts is easily facilitated through feature-rich handsets such as smartphones.

Building a digital ASR corpus for any language is a non-trivial task with various complicating factors such as corpus stratification and design, finances, logistics, licensing issues, personnel issues, time lines, quality control, recruiting of first language speakers, recording environment planning, database management, computer hardware and transport logistics, to name but a few high-level challenges [15, 16].

Following the approaches outlined by various authors for developing different types of speech-related corpora [15, 47–50], the primary stages are discussed next.

2.3.1 CORPUS DESIGN

The first stage of ASR corpus creation is that of corpus design. During this crucial stage, various design decisions are required, such as the total number of speakers, the gender distribution among these speakers, their age distribution, the average length of utterances required, and many more. In general, the purpose of this design phase is to ensure that as much variability as possible is captured in the speech data, while matching the actual intended acoustic environment for the application of the ASR corpus, as closely as possible [25]. Such design is non-trivial and requires extensive knowledge of statistical acoustic models as well as the origin of variability in speech signals.

2.3.2 PROMPT TEXT SELECTION

The next stage of creating an ASR corpus is that of selecting relevant text that needs to be read by the different speakers. The selected text may have to conform to various criteria (such as a specific trigram coverage of the orthography), but would depend significantly on the purpose the corpus is

intended for. When a prompted speaking style is required (as is the case in this study), this textual input material is sometimes called “prompting material” [15].

2.3.3 AUDIO RECORDING

These textual words, phrases or even sentences (depending on the corpus design) need to be recorded in some digital format by a number of speakers; the amount which once again depends on the corpus design. Various recording hardware and software tools ranging from advanced studio equipment to portable recorders may be used.

2.3.4 TRANSCRIPTION AND ANNOTATION

After the recordings have been made, the actual words recorded need to be transcribed or verified either automatically or manually, depending on the level of technology available [51, 52]. These transcriptions typically take on one of two forms, namely, orthographic transcriptions or phonemic transcriptions.

Orthographic transcriptions, is a mapping of the speech signal to the orthography, or writing system of a language, with the basic writing unit called a *grapheme* [16]. Phonemic transcriptions is a mapping of the same speech signal to a set of symbols, each representing the semantically distinct sounds of a language, with these basic sound-units called *phonemes*. Depending on the purpose and requirements of a corpus, either or both forms of transcriptions may be needed.

Usually during this stage, certain pre-agreed annotations sometimes called ‘markups’, also need to be added to the transcriptions to indicate events occurring in the audio, such as noise, mispronunciations or even prosodic annotations for certain types of corpora [53, 54].

These annotations or tags, and the level to which they are applied, again depends on the purpose for which the corpus is intended. In general, these transcriptions also need to be aligned with the audio contents, by segmenting the audio file in correspondence with each word in a transcription, a process for which HTK [55] is often used.

2.3.5 QUALITY VERIFICATION

Upon completing transcriptions either manually or automatically, the transcription accuracy is usually verified in whole or in part, prior to approving the data set. Recent studies by Roy *et al*. [56] have investigated the use of automatic tools for estimating transcription “difficulty” to aid in selecting text needing thorough verification. Roy *et al*. also investigated the use of automated tools to perform quality verification to reduce human effort. Although the latest trends indicate the strong use of crowdsourcing techniques, especially to perform quality verification of transcriptions [39], this will be discussed in a later section to avoid duplication of such a discussion.

The detail of the actual Quality Control (QC) required, again depends on the purpose for which the corpus is intended. The two main groupings of QC required are for transcriptions and for audio

Table 2.1: *Comparison of candidate data collection tools for under-resourced languages.*

Primary requirement	**Hughes *et al.***	**Lane *et al.***
Portability	✓	✓
Internet independence	×	×
Open-source	×	×

data. Should all the individual files (audio and/or transcriptions) pass all the necessary QC stages, a complete corpus *validation* and *evaluation* (as defined in [57]), may further be required prior to releasing the data as an official ASR corpus - "a set of data collected and prepared for a specific use" [16].

2.4 CONCLUSION

Several well established ASR data collection strategies exists and are still commonly used for collection campaigns. However, as pointed out in Section 1.1.3 (page 4), effectively collecting ASR data for under-resourced languages poses unique challenges.

With the most recent trend in ASR data collection capitalising on the increasing availability of smartphones with its associated decreasing costs, these devices provide some of the much needed flexibility of user interface design, ease of localization, and high portability required when collecting ASR data for under-resourced languages.

As the review of above literature has shown, only two known candidate smartphone solutions currently exist for collecting ASR data that could potentially be appropriate for under-resourced languages, namely, DataHound [45] and the iPhone application developed by Lane *et al*. [46]. But in comparing these tools with the criteria set in Section 1.1.3 for tools that would be effective in collecting data for under-resourced languages, neither of these solutions are indeed suitable due to their lack of meeting two of these primary requirements, namely, total Internet independence and having an open-source code base.

Table 2.1 shows in summary, that although both these tools have the basic functionality required for ASR data collection, neither of these candidate tools successfully address the primary requirements of open-source customisability and Internet independence. The challenge thus remains to develop a tool that will enable the effective collection of ASR data for under-resourced languages by keeping the primary requirements of portability, Internet independence and the open-source nature of such a tool in mind.

CHAPTER THREE

WOEFZELA - A NEW TOOL

3.1 INTRODUCTION

The effective collection of ASR data for under-resourced languages is no trivial task. As described in Chapter 1, the primary requirements of portability, Internet independence and the open-source nature of any proposed solution is vital. The review of the literature in Chapter 2 confirmed that the challenge remains to develop such a tool.

This chapter documents the design and development of this tool, called *Woefzela*. Much conceptual information is drawn from *Validation, Verification, and Testing for the Individual Programmer* by Branstad *et al*. [58] and *SWEBOK* [59] in subsequent discussions.

The first part of this chapter will describe how the overall product requirements came into being, contrasting specific requirements provided by an external project initiator, with requirements derived through a requirements analysis process.

The second part of this chapter, from Section 3.3 onwards, will discuss the software design process – from conceptual design, to architecture, through an overview of the software construction stage to final testing.

3.2 PRODUCT REQUIREMENTS

Apart from the primary, non-negotiable product requirements of portability, Internet independence and the open-source customisability described in previous chapters, another set of specific requirements determined the final product specifications. These are called the secondary requirements. The secondary requirements are further subdivided into provided and derived requirements, as the words indicate, having been provided with the former and having derived the latter.

3.2.1 PRIMARY REQUIREMENTS

In serving as a summary of the discussions of the primary requirements in previous chapters, Table 3.1 provides a reference to the primary requirements for developing a new tool that would be effective for collecting ASR data for under-resourced languages.

Table 3.1: *Primary requirements for Woefzela compared with candidates from literature.*

Primary requirement	**Hughes *et al*. [45]**	**Lane *et al*. [46]**	**Woefzela design**
Portability	✓	✓	✓
Internet independence	×	×	✓
Open-source	×	×	✓

3.2.2 PROVIDED SECONDARY REQUIREMENTS

Through a third party initiating this project, a number of basic requirements were provided. Firstly, an efficient ASR data collection tool had to be developed to collect broadband corpora for all the eleven official languages of South Africa, most of which were still considered to be under-resourced. Thus, the NCHLT project described in Section 1.1.4 (page 6) provided the larger context for the use of this tool with regard to the intended purpose of the corpora, the large geographic and language diversity to be covered, the volume and stratification of the data to be recorded, and the budget constraints.

Secondly, all the output format requirements such as using the XML-format for structured output files, recording audio data in the WAVE-file format with LPCM encoding, using a sample frequency of 16 kHz to ensure an 8 kHz bandwidth, and ensuring that all input and output of the tool complies to the UTF-8 Unicode standard, were provided.

Thirdly was the requirement given that the frequency of utterances recorded by all speakers of a specific language should converge over time to a uniform distribution, ensuring that the phoneme coverage predicted during the textual corpus design, is achieved.

Fourthly, a usage protocol similar to that which will be described in the following subsection, Section 3.2.2.1, had to be adhered to. Lastly, the basic functionality of this tool needed to be similar to that of DataHound [45], whilst improving on apparent drawbacks. This functionality will be described in Section 3.2.2.2.

Lastly, a more indirect yet important requirement was the need for this software tool to run on different smartphone hardware. This need arose in particular from the different handsets available at the time, as well as the more generally important aspect of capturing some of the needed variability in the data collected by using different handset models.

3.2.2.1 USAGE PROTOCOL

The provided protocol for collecting ASR data was to be based on the assumption that collection will be overseen by *Field workers*, who are responsible for canvassing, enrolling, training and guiding

Respondents who provide the actual speech data. These Field workers are therefore responsible for the actual data collection process.

Contractors, on the other hand, are generally responsible for a complete data collection campaign, typically recording a number of languages in parallel, and are required to recruit any needed Field workers.

From the perspective of a Field worker, the process of acquiring data from a single Respondent, needed to conform to the following protocol:

1. Screening: The language ability and fluency of the Respondent is assessed by a qualified mother-tongue speaker, prior to being enrolled for any further activities.

2. Registration: A basic record of the Respondent's personal information is created, including a record of data collection consent, and any agreed rewards for services rendered.

3. Training: The Respondent is trained on the use of the tool by a Field worker, and records an initial number of prompts in order to familiarise himself/herself with the general functioning of the application. This is called a *training session*.

4. Recording: Upon successfully completing the training session, the Respondent is presented with a target number of prompts, while recording the audio data. This is called a *recording session*.

5. Reward: Upon completion of the recording session, the session is automatically terminated by the application and the Respondent is rewarded by the Field worker, as per prior agreement.

A further set of provided secondary requirements were the functional requirements.

3.2.2.2 FUNCTIONAL REQUIREMENTS

Only two primary functional components were specified, as listed below:

1. Capture Respondent meta data: This information is essential in keeping track of any data associated with the Respondent. Information such as age, gender, primary language and recording environment is required.

2. Control audio recording and storage: The Respondent must be presented with a number of textual prompts to record, and both the prompts and the recorded audio files must be stored in some form. The ability to Start, Stop, Record, Playback, Re-record and Skip prompts, must also be implemented.

3.2.3 DERIVED SECONDARY REQUIREMENTS

Apart from receiving the provided secondary requirements, a number of other secondary requirements were derived from (i) the provided secondary requirements, (ii) interview discussions with in-house Contractors experienced in using DataHound, and (iii) discussions with colleagues regarding a previous Lwazi project; referred to in Section 1.1.1, page 2.

Through this basic requirements analysis process, a set of further secondary requirements were derived from these inputs, which are not as crucial as the primary requirements, yet impacts specifically on the effectiveness of this tool in collecting ASR data for under-resourced languages.

3.2.3.1 MAXIMISING RECORDING OPPORTUNITY

Remote locations compound a number of challenges faced when collecting ASR data for under-resourced languages. Transporting the equipment to remote areas or people from remote areas are costly and potentially risky. When first language speakers need to be reached in remote areas or when such speakers are sparsely distributed over wide geographic areas, *recording opportunity* is of prime importance.

Once a Respondent is engaged in a recording process, maximum usable data must be obtained during a session as the cost of procuring the services of the same person, or even another first language speaker, may not be feasible.

By assuming that the average respondent will only make a limited number of recording errors, all the quality assurance of the data is left for the post-processing stage. This may lead to subsequently discarding large amounts of data, incurring unnecessary losses.

Thus, by closing-the-loop on the quality of recordings as quickly as possible (i.e. on the mobile device), a waste of various resources are potentially avoided. The specific solution called *QC-on-the-go*, will be introduced in Section 3.3 as part of the conceptual design of Woefzela.

3.2.3.2 PROVIDING SUPPORT FOR FIELD WORKERS

The process of assisting a number of Respondents to donate speech data, may be very exhausting, potentially impacting on the quality of such supervision. In order to support Field workers as much as possible, some specific additional secondary requirements were derived.

Firstly, by ensuring that the usage protocol is – as far as possible – implicitly adhered to through program design, Field workers may address more exceptional issues. By requiring Woefzela to firstly enforce an enrollment procedure, followed by a training session which was previously manually enforced, prior to allowing a formal recording session, this outcome could be achieved.

Secondly, by requiring specific, structured information from each Respondent for successful enrollment, the challenge of sourcing needed information at a later stage, is alleviated. In creating standard profiles for Respondents, the user interface could be used to enforce entry of the required fields.

Furthermore, in South Africa a legal requirement exists that even part-time workers must be above the age of 16 in order to be remunerated for their services. This includes data donation services. By simply requesting the Respondent's South African identity number, the Field worker can derive the Respondent's age (while potentially verifying the validity of a provided identity number), circumventing embarrassing situations that may arise by asking a person's age to ascertain legal compliance.

Lastly, an often overlooked aspect of speech data collection is the ethical aspect of consent. By ensuring that all Respondents have seen and agreed to the Terms and Conditions prior to taking part in a recording session, this important issue is addressed.

3.2.3.3 PROVIDING SUPPORT FOR CONTRACTORS

As the main agents overseeing complete recording campaigns, Contractors are legally responsible for recording any financial remuneration awarded to Respondents for services rendered. By providing a specific field in the graphical user interface during Respondent enrollment for the remuneration agreed upon, Contractors could easily keep track of any such expenditure in electronic form.

Further, since the overall responsibility of the quality of data lies with the Contractor, it is important for Contractors to be able to associate specific recording sessions with each Field worker responsible. By enforcing an enrollment process for each Field worker along with the enrollment of each Respondent, individual performance management is facilitated; aiding the Contractor.

3.2.3.4 SIMPLIFYING POST-PROCESSING OF DATA

When data has been collected for a language, this data needs to be developed into an ASR corpus typically through renaming of files according to a certain convention and grouping of files into a predefined folder structure; apart from any quality control required. In order to simplify the automation of the post-processing of these files which are typically large volumes of data, a number of additional requirements have been derived:

- File and folder naming conventions must be consistent, and was chosen as such as to ensure that all file names are distinct across all recording devices, and across all languages recorded with Woefzela.
- Personal information of Field workers and Respondents, must be easy separable from collected data to avoid additional post-processing of data to remove references that could potentially be linked to an individual's identity.

In conclusion, all these above primary, provided secondary and derived secondary requirements were synthesised into a software design for Woefzela. Some of the overall conceptual design components such as QC-on-the-go, are further explained in this chapter, while Chapter 4 will elaborate on other solutions provided in meeting these requirements.

3.3 SOFTWARE DESIGN

3.3.1 CONCEPTUAL DESIGN

The conceptual design of Woefzela can be divided into a general functional design and the more specific QC-on-the-go functionality, while keeping all the primary and secondary product requirements in view. Each will be discussed.

3.3.1.1 GENERAL FUNCTIONAL CONCEPT

The first stage of the program functionality encompasses the enrollment process during which Field worker, Respondent and Session profiles are completed. In this stage, information is gathered and stored in the required format.

In the second stage, a training session commences. The Respondent is shown by the Field worker how to use the tool and is given a number of prompts to record. Full processing is done on these prompts, including QC-on-the-go, to ensure that the Respondent experiences the full functionality of the tool and becomes familiar with the quality indicators at the bottom of the user interface.

The data generated during the training session is identical in format and location, to the data that would be generated during a later recording session, apart from the fact that the data is clearly marked as training session data. This enables these training sessions to be easily included, or excluded, during post-processing, depending on individual campaign and post-processing requirements.

During the third stage, called a recording session, the actual recording of the target number of prompts is done. This is controlled in the same way as a training session, apart from the target number of prompts being different.

The fourth stage is that of finalisation of the recording session. During this stage, the Respondent is informed of the completion of the session and any outstanding audio files that have already been queued for quality control, is finalised.

In terms of conceptual design, the complete application is thus divided into four sub-tasks, or activities. The first, second and third activities present graphical user interfaces for data entry. The fourth activity is the main recording graphical user interface. In this activity the user is presented with a prompt to record and a number of buttons to control the recording process with.

The software application stays on this main recording screen during the complete training and recording sessions, unless the user chooses to terminate the program. Once the recording session has been completed, the user is presented with an overlayed message informing him/her of the successful completion of the session. Upon acknowledgement of this message, the program terminates.

3.3.1.2 QC-ON-THE-GO CONCEPT

In order to maximise recording opportunity as explained in Section 3.2.3.1, the QC-on-the-go functionality of Woefzela was devised. Since the primary goal of recording speech data is to ensure that

a certain target number of good recordings are collected for each speaker (with the quality of speech data being dependent on the particular application of the data), by guiding Respondents towards a specific number of good recordings per session, rather than a specific number of audio files per session, more usable speech data should be collected.

Traditional methods of checking the quality of data during post-processing to ascertain the quality introduces unnecessary losses when collecting speech data for under-resourced languages.

Since Internet independence must be maintained, this semi-real-time quality control mechanism, QC-on-the-go, cannot be performed on back-end servers in time to change the target number of prompts based on the quality of recorded prompts, and thus this functionality must be implemented on the device itself.

Thus, once a Respondent has finished recording a prompt, the audio file is submitted to a QC-on-the-go service, running in the background on the device for such further quality checks, with the results of these checks being written to an XML-file. If any of these quality criteria is not met for an audio file, an additional prompt is loaded for the Respondent to record, thus pursuing a target number of good recordings, and not simply any audio data.

A basic set of quality control criteria that was deemed appropriate and computationally tractable for the hardware available at the time, were initially chosen. These criteria are by no means exhaustive or well researched, but simply provided an initial starting point for the proposed novel QC-on-the-go process. These quality criteria are described next.

QC flags

The following quality checks are to be performed on the audio file and the subsequent results stored in an associated XML-file:

Volume level

On mobile phones it is often easy for the user to unintentionally cover the microphone causing the volume of the recording to be too low, or speaking too closely into the microphone – causing the volume to be too high. This flag in the XML-file will indicate such QC failure.

Start/stop errors

Should a user start speaking prematurely, truncation of the speech signal will happen at the start of the recording. On the other hand, if the user presses the button to stop recording too early while still speaking, truncation at the end of the recording results. In order to avoid both these errors, as well as provide feedback to the user, an empirical root-mean-square threshold was set for the first and last N milliseconds of the audio. Should the threshold be reached, it was deemed that an unacceptable amount of energy was present in the signal at the start/end, and that potential information could be lost.

To further inform the user when the hardware is ready, in an attempt to avoid such truncation errors, the prompt text changes colour to indicate the various states. In this aspect, DataHound has a superior implementation by maintaining a rotating buffer for recordings and subsequently writing the buffer to a file while including 0.5 seconds of recorded audio before initiation and after termination of the recording.

In very noisy environments such as traffic, close proximity to air-conditioning noise or background sources such as bird songs (common in rural African settings), testing of the quality check feature caused the recordings to fail the quality criteria, seemingly indicating a start/stop truncation error, when the actual problem involved too much energy being present in the surrounding ambient noise. This is in fact a desirable conclusion, as such high ambient noise levels indicate that the environment is unsuitable for speech data collection.

3.3.2 ARCHITECTURE

In order to describe the general architecture used, a brief overview of the Android application framework is provided to serve as a basic guide to the implementation details and terminology mentioned later in this chapter.

3.3.2.1 ANDROID APPLICATION FRAMEWORK

Framework components

The major components of the Android Operating System shown in Figure 3.1, is a Linux kernel with a set of system libraries and an Android runtime environment. The virtual machine, called the Dalvik Virtual Machine, is similar to that of a standard Java Virtual Machine, but is especially optimised for devices with limited resources such as mobile phones.

Four application components form the major building blocks for a software developer, namely, Activities, Services, Content Providers and Broadcast Receivers. Depending on the target application in mind, Activities and Services form the primary building blocks.

Component interactions

An *Activity* is generally associated with User Interface (UI) screens and thus has a visible association with the user. *Services* on the other hand, run in the background and do not have any visible interface with the user.

An Android application, such as Woefzela, is therefore constructed in terms of different user interfaces, each performing a specific task or function that the user needs to interact with. In the background, different services, potentially influenced by the actions taking place on the User Interface, that is, in the Activity, may perform tasks that require no direct user intervention, except maybe at key stages for which an Activity would be required.

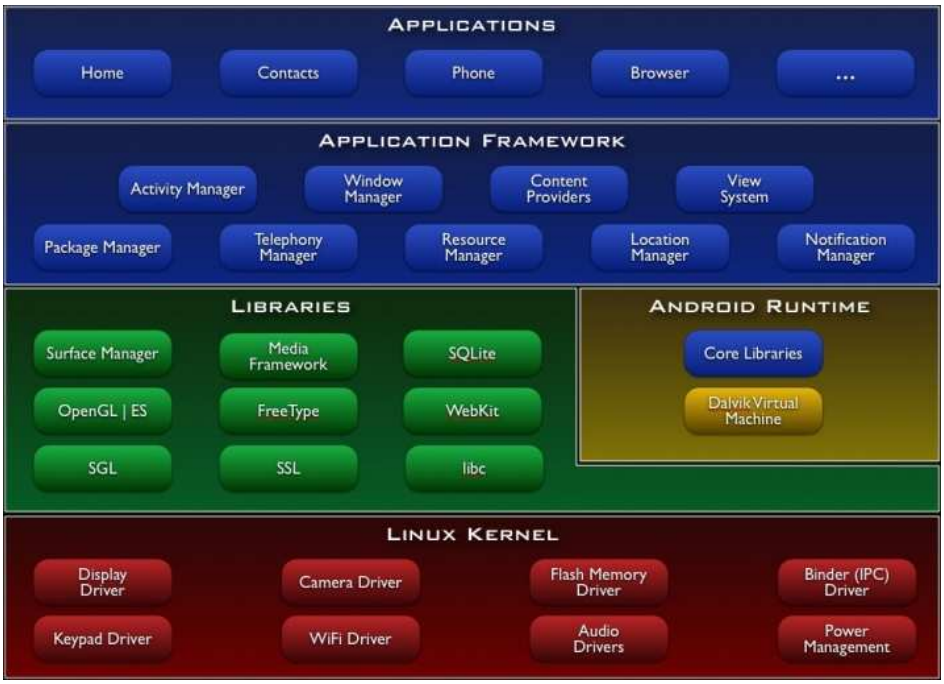

Figure 3.1: *Major components of the Android operating system [60].*

3.3.2.2 WOEFZELA FRAMEWORK COMPONENTS

The main Android framework components used in Woefzela are listed in Table 3.2. These framework components are loosely connected by means of an "AndroidManifest" XML-file in which these different Activities and their allowable interactions are defined.

Table 3.2: *Android framework components employed in Woefzela.*

Component	**Component class**
Field worker profile entry user interface	Android Activity
Respondent profile entry user interface	Android Activity
Session profile entry user interface	Android Activity
Main recording user interface	Android Activity
QC-on-the-go functionality	Android Service

3.3.2.3 PRINCIPAL OBJECTS

Two important data-structure-groupings exist in the design of Woefzela, namely, those associated with prompts, and those associated with the QC-on-the-go functionality, here referred to as the *QC Service*.

`PromptList` class

The `PromptList` class reads a set of textual prompts from an input corpus and, upon demand, provides the next prompt to be displayed. When this class is instantiated, the input corpus as well as the target number of prompts is passed along as parameters to the constructor. In order to fetch the next available prompt from this object, a call to the `extractNextString` method is made – without the need to pass any parameters.

`QCQueue` class

The second key class employed in Woefzela is called a `QCQueue`. A `QCQueue` is a linear first-in-first-out queue of `qcObjects`. These `qcObjects`, instantiations of the `QCObject` class, holds the necessary references to the audio files that require quality control, along with additional information regarding the prompt associated with each file.

Once a `qcObject` has been instantiated, it may be sent to an instantiation of a `QCQueue`, by invoking a method call `addItemToQueue`, passing a `qcObject` as a parameter.

In order to deal with the asynchronous nature of the QC Service, two instantiations of the `QCQueue` class could exist at any point in time. As soon as a `qcQueue` object has been dispatched for quality control, no communication with that object is reliable. Thus, while a `qcQueue` is being processed, all `qcObjects` are being added to a *second* `qcQueue` object. Once the asynchronous QC Service has completed all the items in the current `qcQueue` object, it reports back its availability for the acceptance of a new queue.

In summary, two `qcQueue`s would typically exist at any time. The first, actively being processed asynchronously by the QC-on-the-go algorithm, and the second, accumulating `qcObjects` that require quality control in the future.

Finally, as soon as the QC Service has completed an item in a `qcQueue` object, the results are packaged in a further object called a `qcResults` object. This object contains all the outcomes of the QC process, as well as all the original references sent via the associated `qcObject`. These results are then committed to an XML-file associated with the initial audio file.

This multiple queueing system is necessary, since once dispatch of a queue has been made to the QC Service, the QC Service runs asynchronously and must complete all the items in the queue before any further communication with the QC Service is reliable. Other designs are possible, but this design lead to a simpler implementation strategy.

3.4 SOFTWARE CONSTRUCTION

The coding of Woefzela was reasonably straightforward. Implementation was facilitated by dividing the functional aspects of the design into various parts, and coding these parts, integrating the functional units throughout. The general programming methodology used was that of Agile software

development in which a set of functionality is grouped and kept stationary until implementation of such a set has been completed. Implementation of each of these sets also included testing of the intended functionality.

3.4.1 PRINCIPAL CLASSES

The principal classes employed in Woefzela are listed in Table 3.3 with a short description of the main functionality of each class.

Table 3.3: *Principal classes of the Woefzela implementation.*

Class name	**Description**
FieldworkerProfile	A class providing the main Field worker user interface functionality.
LoadFieldworkerProfile	A class implementing the loading of an existing Field worker profile.
SaveFieldworkerProfile	A class implementing the ability to save a Field worker profile for future loading.
RespondentProfile	The main Respondent user interface functionality.
LoadRespondentProfile	A class implementing the loading of an existing Respondent profile.
SaveRespondentProfile	A class implementing the ability to save a Respondent profile for future loading.
SessionInfo	The main Session information entry user interface class.
SaveSessionInfo	A class implementing the ability to save a Session profile.
MainRecordingActivity	The main recording activity interface and control class.
RecordingWAV	A class implementing the functionality to poll the microphone hardware for values and store this in a WAV-file format.
MyService	The QC-on-the-go functionality primarily resides in this class which is run as an Android Service in the background.
Logging	A helper class that facilitated program development and debugging by easily turning different levels of debug messages on or off.

3.5 FINAL SOFTWARE TESTING

Software testing was done at various stages of program development and integration. As generally recommended, software testing should not be postponed until the product is completed, but should form a part of each development stage [58]. As required, standard testing practices such as structural testing, path testing and boundary value testing were performed.

Apart from testing during the different development stages, the primary mode of validating the software was by means of pilot deployments in which the tool was used in real recording conditions.

3.6 CONCLUSION

In this chapter an overview of the software requirements, design, construction and final testing of Woefzela was provided. In the next chapter, a verification process will be followed to ensure that

the software was built correctly, and in Chapter 5, the results produced by Woefzela will follow a validation process to confirm agreement with its original design intent – to effectively collect ASR data for under-resourced languages.

CHAPTER FOUR

WOEFZELA VERIFICATION

4.1 INTRODUCTION

In accordance with the *IEEE Standard for Software Verification and Validation*, IEEE Std 1012-2004 [61], various levels of "intensity" may be applied when performing software verification and validation, depending on the criticality of the application.

For *Woefzela*, sufficient validation of this tool to perform effective ASR data collection for under-resourced languages, is by means of building an actual automatic speech recogniser with data recorded with Woefzela, and subsequently evaluating the performance of this data. By comparing the recognition accuracy of such a system with that found in literature the quality of the collected data is in fact ascertained – as long as the architecture of the ASR system built, matches that in the compared literature. This validation is carried out in Chapter 5.

In terms of software verification, the current chapter intends to shows that:

1. Woefzela conforms to the basic product requirements as set out in Section 3.2 (page 19), including the primary and secondary requirements.

2. The real usability of any proposed tool is a key criterion in the effective collection of ASR data for under-resourced languages. Section 4.12 will specifically focus on two aspects of Woefzela's usability; providing further evidence for verification of its functionality and effectiveness as an under-resourced languages data collection tool.

4.2 PRIMARY REQUIREMENTS VERIFICATION

The primary product requirements of portability, Internet independence and open-source software, formed part of the explicit design criteria of Woefzela. As such portability is implied through the use

of smartphone handsets, and the open-source nature of the code through its license agreement and code availability.

Woefzela's independence from Internet connectivity was eliminated by employing the SD card on the smartphone handsets as the primary data exchange mechanism. Since Field workers are always in close proximity to Respondents, the utilisation of a laptop to upload data onto a hard drive and download prompts to the device solved this problem.

4.3 SECONDARY REQUIREMENTS VERIFICATION

In spite of the two origins of the secondary requirements, namely provided and derived requirements, as discussed in the previous chapter, these can easily be merged into a single set of secondary requirements.

Table 4.1 provides a summary of these requirements, with a cross-reference of the main section in which each requirement will be discussed, but, since a single requirement may impact on a number of design decisions, these discussions may also appear throughout other sections. In particular, Section 4.4 on the functionality implemented in Woefzela, relates a number of product requirements to the various functional aspects implemented in the final product.

Table 4.1: *Secondary requirements for Woefzela and verification reference.*

Secondary requirement	**Verification discussion**
Functionality	Section 4.4
Output format	Section 4.5
Utterance frequency	Section 4.6
Usage protocol alignment	Section 4.7
Maximising recording opportunity	Section 4.8
Providing support for Field workers	Section 4.9
Providing support for Contractors	Section 4.10
Simplifying post-processing of data	Section 4.11

4.4 FUNCTIONALITY

A description of the functionality implemented in Woefzela is provided below, primarily addressing the means through which the product requirements have been implemented in order to verify such product requirement compliance.

Capturing Respondent and Field worker profiles

This information is essential in keeping track of any data associated with both Field workers and Respondents. The profiles generated from this information also enables a new session to load information from a Field worker or Respondent when subsequent sessions involve the same person. Typ-

ically, only the Field worker information is often re-used, but for early session termination, or other reasons, Respondent profiles can also be reloaded. Also of importance is the fact that this profile information can be used, although only with proper consent, in the process of recruiting further first language speakers through these already established contacts.

Typical user interfaces for capturing the Respondent and Field worker's profiles are shown in Figures 4.1 and 4.2. In each of these interfaces, information such as personal names and family names as well as basic contact details are requested. Some of these fields are optional while others are mandatory to comply with various protocol aspects.

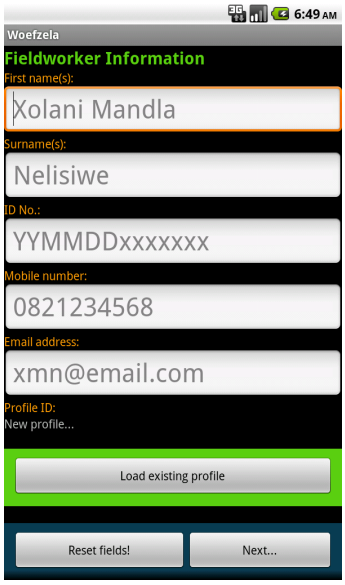

Figure 4.1: *A typical user interface for entry of Field worker profile information.*

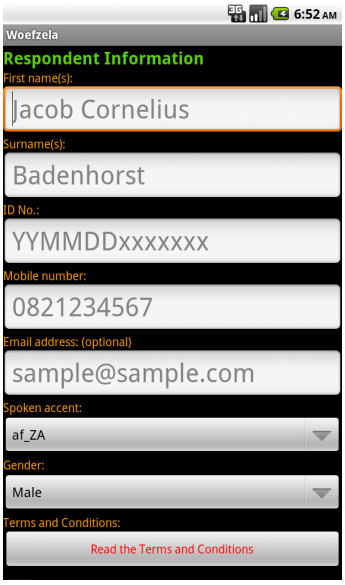

Figure 4.2: *A typical user interface for entry of Respondent profile information.*

Presenting Terms and Conditions

An often overlooked aspect of data collection is the informed consent from the Respondents for the intended use and distribution of the data. By embedding this functionality into the tool, this critical step is enforced, avoiding further legal and ethical complications. Figure 4.3 shows and example of such an agreement.

Generating unique profile keys

In order to associate Field worker profiles and Respondent profiles with Session meta data as well as all recordings, a unique *key* is generated for each Field worker and Respondent. The primary criteria used for such a reference key is its uniqueness across all persons for any compiled speech corpus recorded with Woefzela, and secondly that the association between the key and the individual must only be traceable in one direction. That is, by knowing the person, the key must be calculable, but by

knowing the key, it must not be computationally tractable to determine who the person is. This is to protect the privacy of individuals.

Capturing Session meta data

Figure 4.4 provides an example of the typical meta data associated with each Session. As can be seen from this image, only the unique keys associated with Field worker and Respondent profiles are used from this point forward in the program, ensuring that the generated data set does not contain any personally sensitive information.

As recordings are done in different geographic locations and noise environments, this meta data is often helpful when performing data selection or analysis on a corpus of speech data. By also including meta data such as the agreed remuneration for donating the data, difficult negotiations may be avoided. This meta data is subsequently associated with each Session recorded, providing the necessary meta information for each audio file.

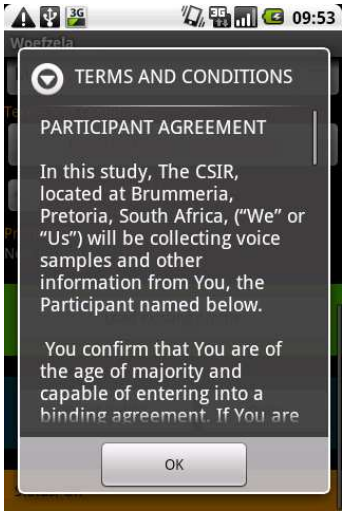

Figure 4.3: *An example of the Terms and Conditions presented to Respondents.*

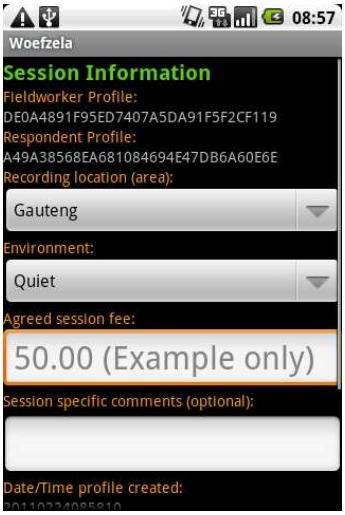

Figure 4.4: *Session information user interface example.*

Controlling training and recording sessions

In alignment with the Usage Protocol described in Section 3.2.2.1, all Respondents must first complete a training session prior to commencing to a recording session. Control over this process, including different recording targets, is exercised by the software.

Selecting and loading prompt batches

As the initial step of any recording or training session, the software selects the initial target number of prompts from a textual corpus provided on the SD card. The primary requirement in this regard is

that the actual recorded prompts, must converge over time, and across all devices, towards a uniform distribution of the frequency of each recorded prompt. This is to ensure that the tri-phone coverage provided by the final corpus closely approximates the intended coverage at the time of the design of the textual input corpus. Section 4.6 will provide a more in-depth discussion of utterance frequency.

Presenting a prompt

The software selects the first and subsequent prompts from the loaded batch of prompts, and presents it to the Respondent for recording. The key design requirement in this case is the compliance to the UTF-8 Unicode standard, since some of the languages to be recorded employs non-ASCII characters and certain diacritic symbols, distinguishing the *pronunciation* of words or phrases.

Controlling audio recording button functionality

The primary functional component of the tool is the recording, re-recording, playback and finalization of the audio for each prompt as exemplified in Figure 4.5.

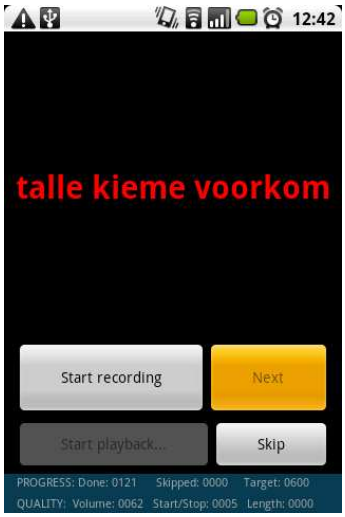

Figure 4.5: *An example of the main recording user interface.*

Figure 4.6: *User interface showing reasons for skipping a prompt.*

The Respondent is able to *Start* a recording at will and commence to speak the required prompt. Stopping the recording either after completion of recording the audio, or due to the desire to *Re-record* or *Skip* a specific prompt, forms part of this functionality. Should a Respondent wish to *Playback* a recording, he/she is able to do so.

If the Respondent chooses to *Skip* a prompt (before or after providing audio data associated with it), they are presented with a list of reasons for skipping a prompt as shown in Figure 4.6, should they desire to make known which reason led to such a decision.

The *Next* button, causes the software to move to a subsequent prompt if the target number of good prompts have not yet been reached. However, the Respondent is only able to move to the next prompt

by either recording audio data to be associated with the current prompt, or by opting to select the *Skip* button.

Store audio and meta data on SD card using associative file names

Since the primary means of storing and exchanging data with Woefzela had to be Internet independent, all file retrieval and storage is done via the SD card. The recorded audio files are stored on the SD card in a logical file system per session. Any accompanying files related to a specific audio file, such as the XML-file containing the prompt string, is stored in the same location with an associated file name. See Figure 4.7 for a typical example of this structure and the resulting XML-files generated.

Perform QC-on-the-go on each audio file

Integral to supporting the notion of maximising recording opportunity, is Woefzela's semi-real-time QC-on-the-go functionality. When quality control has been performed on an audio file, a resulting meta data file is created, capturing the outcome of this quality analysis.

This file is in an XML-format to facilitate easy parsing and aggregation of data during post-processing. The typical files created can be seen by file names ending in "stats.xml" in Figure 4.7. More detail regarding the actual content of these files will be provided in Section 4.5.1.

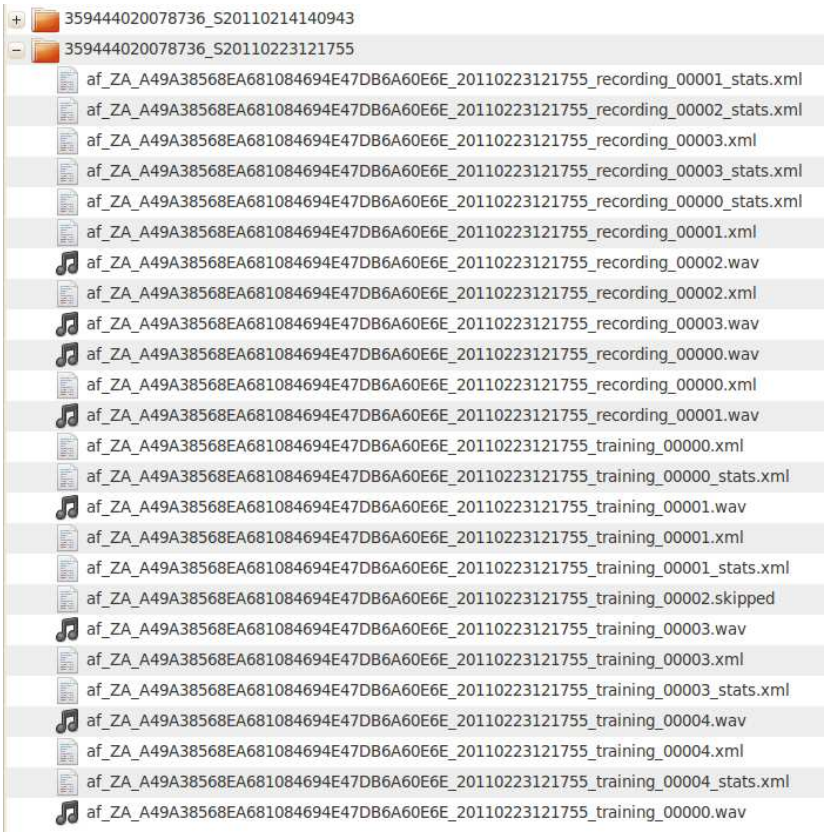

Figure 4.7: *Typical files created by Woefzela for each recording session.*

It is also vital to note that Woefzela informs the Respondent of the current number of recordings that have passed or failed the QC criteria by indicating these totals at the bottom of the main recording interface. This is a passive form of feedback to the Respondent, to avoid any disruptive messages while at the same time acting as a teaching aid.

Control session termination and closure

Upon reaching the target number of good recordings, Woefzela informs the Respondent of this state and subsequently finalises the session and all its associated outputs.

4.4.1 META DATA VERIFICATION

Although conceptually simple, for completeness, the meta data stored with each type of profile is also shown and discussed. With regard to Field worker and Respondent profiles, these are stored in plain text files as they need to be read by the mobile device whenever an existing profile is recalled, thus reducing the overhead of parsing more structured files. By simply placing the information on specific lines in a plain text file, this file can easily be read or written without dealing with unnecessary complex parsing of XML-tags.

4.4.1.1 FIELD WORKER PROFILES

Figure 4.8 shows the data typically contained in Field worker profiles. As mentioned previously, apart from the Session profiles, the Field worker and Respondent profiles must be managed securely to protect the privacy of the individual, but since Woefzela is intended as an in-house tool, this can easily be achieved.

The last line in this example indicates the unique key generated for this profile, which is then subsequently used for all data created in association with this Field worker.

```
MIME-Version: 1.0
Content-Type: text/plain
Nic
De Vries
7101155167083
0823501082
ndevries@csir.co.za
04FFA1D32B0B026AF952DA7B78C89DEF
```

Figure 4.8: *Typical Field worker profile information.*

4.4.1.2 RESPONDENT PROFILES

In Figure 4.9 an example of the fields typically appearing in Respondent profiles are shown.

Finally, the empty line in the example shown in Figure 4.9, indicates the optional ‘Email address’ of the Respondent. Since the graphical user interface does not enforce any entry in this field, the

profile maintains an empty line upon creation, and uses this empty line upon profile retrieval.

```
MIME-Version: 1.0
Content-Type: text/plain
Tamryn
De Vries
7911111111111
5551234567

en_ZA
Female
Terms have been accepted.
4DC6A0EF84CC2990BB062E33938D4C27
```

Figure 4.9: *Typical Respondent profile information.*

4.4.1.3 SESSION PROFILES

An example of the typical information found in a Session profile is shown in Figure 4.10. Here the unique keys associated with each Field worker and Respondent can again be seen, allowing a clear division of personal information and data collected with Woefzela; with basic data management diligence.

In these profiles, the IMEI of the mobile device has been omitted in the profile, since it is used in the actual file name of the profile in order to make these profile names unique across all devices. The “Agreed session fee” field was left empty by the Field worker as implied by the empty line in the profile example.

In the last line of the example, the typical use of the optional ‘Session specific comments’ field can be seen.

```
MIME-Version: 1.0
Content-Type: text/plain
04FFA1D32B0B026AF952DA7B78C89DEF
4DC6A0EF84CC2990BB062E33938D4C27
training
20110105202228
Gauteng
Quiet

Test v5
```

Figure 4.10: *Typical Session profile information.*

As can be seen from the above typical examples, these profiles capture the necessary information required for successful ASR data collection and corpus compilation.

4.5 OUTPUT FORMATS

Since the output of Woefzela has been successfully used in building ASR systems, the format of these files are implicitly stable and predictable, thus it will suffice to provide an example of the generated

folder structure as well as each type of file in this section. The folder structure generated by Woefzela can be seen in Figure 4.11, with the SD card at the root of this structure since all file interchange is done via the SD card.

The `CorpusInput` folder holds the various textual corpora from which prompts are extracted for recording, while the `OutputData` folder forms the root of all generated audio, transcription and XML-files, grouped per session identifier. Profiles associated with each Field worker, Respondent and Session is stored in the `Profiles` sub-tree, and low-level tracking information is kept in the `Tracking` folder.

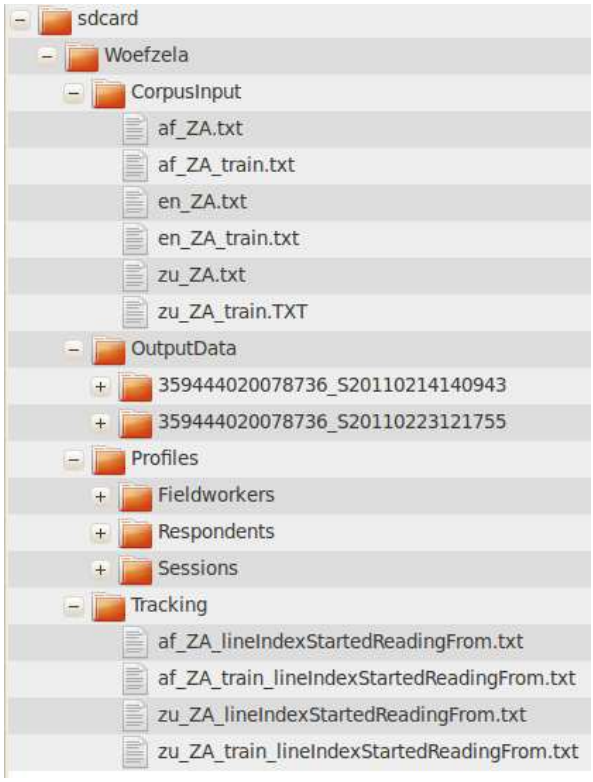

Figure 4.11: *The folder structure generated by Woefzela.*

4.5.1 TEXTUAL FILE FORMATS

The format of the generated prompt files, as well as the QC-on-the-go files are shown in Figures 4.12 and 4.13 respectively. Since the XML standard allows the freedom to define user tags as required, a set of standard tags were devised and consistently used throughout the design. This also allows for easy addition of tags in the future as required, without interfering with existing definitions of tags. From the header section of both of these files it can be seen that all such files are UTF-8 compliant.

The output generated by the QC-on-the-go functionality (as shown in Figure 4.13), takes on a similar form as that of the generated prompt files, yet contains specific information regarding the associated audio file on which the quality control was performed. These flags, apart from the “utterance length”-flag, were discussed in Section 3.3.1.2, page 25. The utterance length is calculated, as

can be seen from this figure, but not yet implemented as a quality decision criterion.

```
<?xml version="1.0" encoding="UTF-8"?>
<Woefzela>
    <prompt>australian broadcasting corporation</prompt>
</Woefzela>
```

Figure 4.12: *Example of a prompt XML-file associated with each audio file.*

```
<?xml version="1.0" encoding="UTF-8"?>
<Woefzela>
    <statistics>
        <prompt>australian broadcasting corporation</prompt>
        <audioIsClipped>false</audioIsClipped>
        <audioVolumeTooLow>false</audioVolumeTooLow>
        <audioTruncatedAtStart>false</audioTruncatedAtStart>
        <audioTruncatedAtEnd>false</audioTruncatedAtEnd>
        <audioUtteranceTooShort>false</audioUtteranceTooShort>
        <audioUtteranceTooLong>false</audioUtteranceTooLong>
        <audioUtteranceLength>2.579999942332506</audioUtteranceLength>
    </statistics>
</Woefzela>
```

Figure 4.13: *Typical XML output file generated by the QC-on-the-go functionality.*

In closing, the key aspect to draw attention to is the ease of extensibility of all of these files and tags for new data collection campaigns, as required.

4.5.2 AUDIO FILE FORMATS

Special mention must be made of the file format in which the audio data is stored. For simplicity of manipulation the WAVE-format was chosen as, although not the most compact format of storing audio data, these files are easily manipulated and inspected with basic software tools available. Through post-processing of the files for ASR system development, these files have been verified as having sample rates of 16 kHz, and as encoding the signals as 16-bit, signed, PCM values.

4.6 UTTERANCE FREQUENCY

In order to approach a certain tri-phone coverage when prompts are spoken, as is necessary for the current use case, it is important to ensure that a uniform distribution of the frequency of all the prompts in a textual input corpus is achieved as the textual corpus is designed with this in mind.

In order to achieve such a coverage, the initial design assumed that if a uniform distribution is aimed at for each device, then the same distribution will hold across all devices as, currently, all operational devices are employed in a single recording campaign. This is not a poor assumption, but the implementation of achieving this goal needed refinement from the initial version of the software.

In order to ensure a uniform distribution on a device, a simple solution was to keep track of the position in the corpus where the previous user had stopped reading (per language), and let the following reader start at the next corpus position. This would indeed ensure a uniform distribution on each device as this process is deterministic. In keeping track of this end-position, a ‘tracking file’ was

created on the device, as keeping track of variables during power loss, would introduce additional risk. The tracking file would be updated each time that a prompt has been completed, so that, in the worst-case, the exact position would only be wrong by one prompt when the battery fails or the system malfunctions in some way.

As Field workers would archive data from SD cards regularly, and subsequently clean the SD card from any unnecessary data, a special tool was created (in Python) to aid the Field workers in doing so. Although this tool was more intelligent with regard to the specific location and nature of the data, it was much slower in removing data from the SD cards than standard computer operating system functions.

Thus, instead of using the tool specifically developed for archiving and cleaning data from the SD cards, the Field workers were advised to use standard operating system tools to perform the similar tasks. This lead to the removal of the position tracking file, causing the corpus start position to be reset each time a SD card was archived. As a consequence, this lead to a non-uniform coverage of all the prompts in the corpus, impacting on the tri-phone distribution intended during the design of the textual corpus.

As a solution, a random starting point was generated for each session, eliminating the need for any start position tracking. This would allow the distribution of the prompt frequency to *tend towards* a uniform distribution over time, instead of providing a means to exactly control the distribution per device. In retrospect, the distribution across multiple devices would in any case only tend towards a uniform distribution, and not be an exact solution.

This solution was simpler, safer, more elegant, and easily verified through logging this generated starting position and subsequently analysing the distribution of these indices.

4.7 PROTOCOL ALIGNMENT

In referring to the earlier functional verification section, Section 4.4 on page 32, it would be simple to ascertain not only compliance with the data collection usage protocol, but also to see that a number of the protocol steps are in fact enforced through the individual graphical user interfaces, as well as through the fixed sequence in which these interfaces are allowed.

4.8 MAXIMISING RECORDING OPPORTUNITY

A complete analysis of the effectiveness of the QC-on-the-go functionality will be performed in Section 4.12.2 on page 43, indicating the maximisation of available recording opportunities, and therefore will not be duplicated here.

4.9 PROVIDING SUPPORT FOR FIELD WORKERS

Since compliance to the usage protocol is enforced (as discussed in Section 4.7), this also provides the necessary assistance for Field workers in collecting relevant and compulsory meta data required for each Respondent and Session.

In ensuring that a Respondent is presented with the Terms and Conditions of participation in a data collection session, and subsequently recording their acceptance/rejection of such terms, ethical aspects of consent are complied with.

4.10 PROVIDING SUPPORT FOR CONTRACTORS

Through the sequence of enrollment interfaces dictated by Woefzela, Field workers are required to associate their profiles with each recording session undertaken. In using this information, Contractors can control remuneration of Field workers through analysis of the amount and quality of data collected from each Respondent associated with a specific Field worker.

4.11 SIMPLIFYING POST-PROCESSING OF DATA

During post-processing of collected ASR data, the consistency of file and folder names, are essential. Through a naming convention involving each mobile device's unique International Mobile Equipment Identity (IMEI) number, time-stamps of the start of a session, and sequential utterance numbers, each recorded file name is unique across all devices. Evidence in this regard can also be seen from Figure 4.7 (page 36).

The information collected as part of the Respondent or Field worker's profiles are easily separable from the ASR data that the person recorded by generating a unique key from the information provided using a Message-Digest algorithm (MD5). This facilitates an easy separation process to ensure no personal information is contained in compiled ASR corpora.

4.12 USABILITY VERIFICATION

The primary evidence for the usability of Woefzela is the sheer amount of data successfully collected with this tool. However, strong secondary evidence also exists through an evaluation of the maximisation of recording opportunity, by analysing the effectiveness of the semi-real-time QC philosophy in reaching the desired number of good recordings compared to purely setting a fixed target number of audio files per session.

An overview of the ASR data collected with Woefzela is provided in the next section, while Section 4.12.2 will provide a more in-depth analysis of the impact of the QC-on-the-go functionality.

4.12.1 SUCCESSFULLY COLLECTED CORPORA

The ASR corpora, consisting of audio data with associated transcriptions and meta data, successfully collected with Woefzela, is summarised in Table 4.2 [62]. These files all passed the QC-on-the-go criteria of Woefzela. This highlights the usability, and thus effectiveness of this tool for collecting ASR data for under-resourced languages.

Table 4.2: *Summary of ASR corpora successfully collected with Woefzela. (Language order same as that used in Table 1.1).*

Language	**Speech data (hours)**
isiZulu	59.02
isiXhosa	76.48
Afrikaans	68.97
Sepedi	69.16
Setswana	70.16
Sesotho	72.42
SA English	72.86
Xitsonga	86.20
siSwati	81.90
Tshivenda	80.80
isiNdebele	60.40
Total	798.37

4.12.2 ANALYSIS OF SEMI-REAL-TIME QC PHILOSOPHY

A National Centre for Human Language Technology (NCHLT) project introduced in Section 1.1.4 (page 6), focussed on collecting broadband ASR data for all eleven official languages of South Africa. Having recorded more than a hundred hours of ASR data within a few months at the time, the start of this project served as *a rigorous test bed for Woefzela*, and for performing this analysis of the effectiveness of its QC-on-the-go functionality.

In a case study [63], relevant statistics of four of the languages available at the time, were analysed. These languages were Afrikaans (Afr), South African English (Eng), Sepedi/Northern Sotho (Nso) and Zulu (Zul). The aim of the analysis was to establish whether the introduction of the QC-on-the-go functionality aided achieving the desired number of good recordings per speaker. If so, this would maximise the recording opportunity, since first language speakers are not easily recruited and repeat recordings from the same speakers are often impractical in developing world environments.

Although common sense suggests that the pursuit of a desired number of good recordings (as opposed to having a fixed number of audio files per speaker) will lead to a better distribution of good recordings, the question that arises is whether an even simpler solution would not be sufficient. Will increasing the fixed target with a certain number of prompts not compensate for errors to be made by the Respondents? Also, if a moving target of good recordings is pursued, how close can one expect

to get to the set target? Both these questions will be explored below.

4.12.2.1 DATA SET SELECTION

In order to investigate these questions, a subset of the available data for each language was selected, consisting of all the sessions for which additional prompts have been loaded due to audio files failing any of the quality criteria listed in Section 3.3.1.2 (page 24). Eliminated from this list were sessions that *terminated* before reaching the initial target of 500 prompts. For example, for Afrikaans, of the 130 *potential* sessions, 11 terminated before reaching 50% of the prompts and 19 terminated with between 50% and 75% of the prompts completed. The remaining 100 sessions (amounting to just under 27 hours of speech data), formed the data set for the analysis performed on the Afrikaans corpus below. Similar information for the four languages analysed is summarized in Table 4.3.

Table 4.3: *Breakdown of the number of sessions per category to arrive at the analysis data set.*

Language	Potential	Terminated	Analysis set	Analysis hours
Afr	130	30	100	26.9
Eng	122	33	89	20.4
Nso	178	73	105	28.8
Zul	176	55	121	42.4

4.12.2.2 PERCENTAGE FAILED RECORDINGS

Figure 4.14 shows the distribution of the percentage errors that Afrikaans speakers made per session i.e. failed files versus total number of files, for the first 500 prompts; where QC has not yet had the effect of loading additional prompts. Table 4.4 summarises some statistics of this distribution, confirmed as typical for the four languages under investigation.

Table 4.4: *Summary of percentage total errors made per recording session for four languages.*

Language	Mean (percentage)	Std dev (percentage)
Afr	13.9	18.1
Eng	10.6	11.6
Nso	11.7	16.4
Zul	11.0	12.7

A number of interesting observations can be made from Figure 4.14 and Table 4.4. It appears that some users do not correct their behaviour based on the passive statistics presented to them on the device in semi-real-time. This suggests that a more disruptive method could be used to guide the behaviour of Respondents.

The substantial inter-speaker variability observed in these experiments (between 11.6% and 18.1%) suggests that the concerns related to a fixed number of recordings (without QC) are justified.

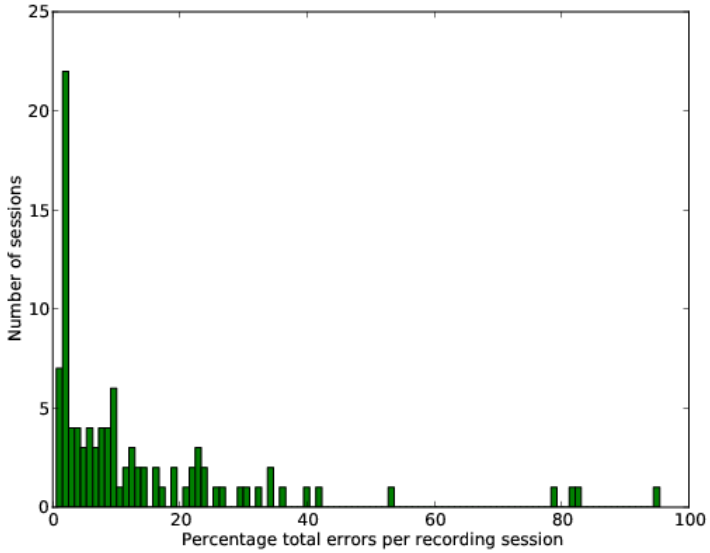

Figure 4.14: *Histogram of percentage total errors made per recording session for Afrikaans.*

Thus, targeting the number of good recordings, is indeed beneficial, as this high variance suggests that by simply adding a fixed number of additional prompts, the target would not be reached in a number of instances; without undue losses.

4.12.2.3 COMPARISON BETWEEN FIXED AND MOVING TARGET

The effectiveness of targeting the number of good recordings during the recording process, can be evaluated by computing the number of good quality recordings at the time that the session completes. Figure 4.15 shows the distribution of the number of recordings made per session that have passed the QC criteria for Afrikaans. Since the distributions of the other three languages are again similar to that of Afrikaans, the overall results are summarised in Table 4.5.

Table 4.5: *Summary of number of acceptable recordings made per session for four languages.*

Language	**Mean (No. of files)**	**Std dev (No. of files)**
Afr	456	79
Eng	475	27
Nso	461	72
Zul	471	52

A few brief comments on Table 4.5 are in order. It is important to note that although the actual target number of good recordings, of 500, is still not achieved, the bulk of the data now lies well within 10% of the actual target as can be seen from Figure 4.15. One reason – which holds for all the languages – for not achieving the actual target, is the fact that the semi-real-time QC algorithm runs only as processor time becomes available on the device, keeping the recording process at maximum

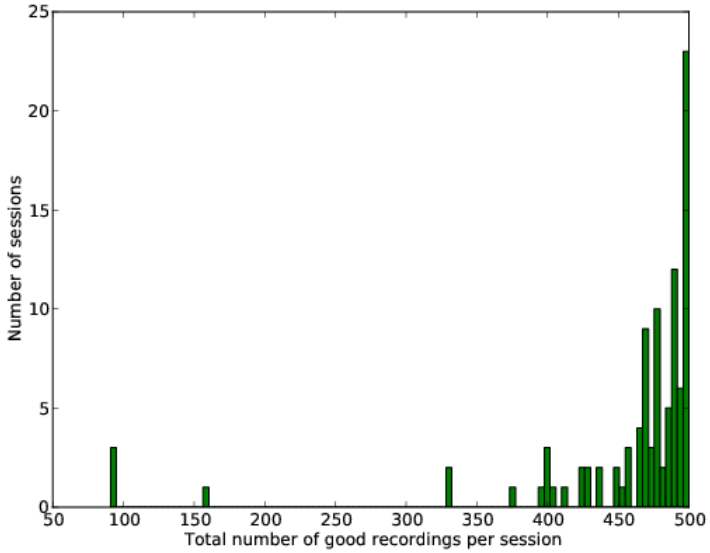

Figure 4.15: *Histogram showing the number of good recordings made per session using a moving target approach for Afrikaans.*

priority. This causes the data dispatched for QC to lag behind the data recorded. By employing faster CPUs or optimising the QC algorithms this lag could be reduced, but unless real-time QC can be achieved, such a lag will remain, leading to some data not being verified in terms of quality at the time of session completion.

Equally important as the target number of samples, is the observation that the variance for most of these languages is substantially less than in the uncontrolled case. This provides the opportunity to add a fixed number of recordings to the initial target without incurring extensive losses; a strategy which would not have been sensible in the higher variance case.

4.12.2.4 CONCLUSION OF SEMI-REAL-TIME QC PHILOSOPHY EFFECTIVENESS

Challenges faced in collecting data for under-resourced languages called for an embedded quality control method in order to optimise the recording opportunity per respondent session. This QC-on-the-go process monitors the quality of recordings and adds additional prompts to the session if the quality of recorded prompts falls outside a pre-defined set of criteria.

Based on this analysis, the semi-real-time quality control process deployed on the mobile device, increases the number of good recordings per session. This maximises the recording opportunity which is a key aspect of the usability of this tool for under-resourced language data collection. Figure 4.16 summarises this conclusion.

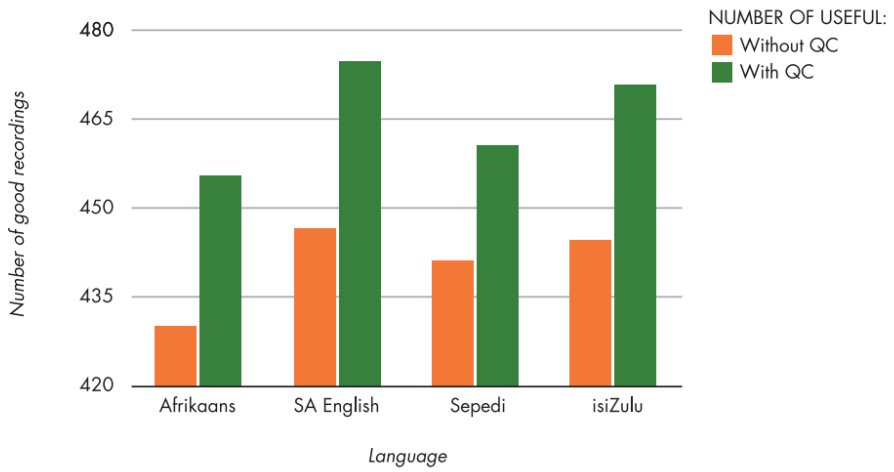

Figure 4.16: *Summary of the effectiveness of the QC-on-the-go philosophy.*

4.13 CONCLUSION

Through the evidence provided in this chapter, Woefzela is verified against its functional requirements, including its output format and meta data requirements, while the usability of Woefzela in performing effective ASR data collection for under-resourced languages was also confirmed.

For further information on Woefzela's functional abilities, a *User Guide* of the latest version of Woefzela is provided on-line [64].

In the next chapter, a number of ASR systems are developed based on the data collected with Woefzela, to show that the output generated by this tool has been validated for its intended purpose.

CHAPTER FIVE

WOEFZELA VALIDATION

5.1 METHODOLOGY

By building an actual automatic speech recogniser with an architecture comparable to that found in scientific literature, the quality of the input data, and not the recogniser performance as such, is evaluated, thus validating the data collected with Woefzela.

In comparing phone recognition results between systems rather than word recognition results, aspects of language modelling do not come into effect, thus making such a comparison more suitable for this study.

This chapter will present the experiments and results of a number of ASR systems developed with data collected by Woefzela. In Section 5.2 the architecture and results in relevant literature will be described, as this will be used to compare the experimental results with. In Section 5.3 an overview of the experiments will be given, followed by a number of sections discussing the various experiments. The closing section of this chapter will conclude on the findings.

5.2 COMPARATIVE RESULTS IN LITERATURE

The Lwazi corpus, previously summarised in Table 1.1 in Section 1.1.1, was collected over a telephone channel, capturing both read as well as elicited speech from approximately 200 mother-tongue speakers in each of the eleven official languages of South Africa.

A phone-based recogniser built by Van Heerden *et al*. [65] was based on standard Hidden Markov Model (HMM) acoustical models, by using the Hidden Markov Model Toolkit (HTK) [55] from the Cambridge University Engineering Department. The system employed mel-frequency cepstral coefficients (MFCCs) based on 13 coefficients and the respective first and second order derivatives

of these coefficients, to capture additional temporal information in the signal. In extracting these features, cepstral mean and variance normalization (CMN and CVN) was done per speaker.

For each of the three emitting states of each acoustic model, a mixture of 7 Gaussian distributions was used to model the output probability distribution, with the covariance matrix of each of these output distributions constrained to be diagonal, as is common practice for typical ASR system development [26]. For further performance improvement, semi-tied transforms were used. A flat phone-based language model was used in order to perform phone recognition.

For the South African English recogniser developed, Van Heerden *et al*. used a gender balanced training set of approximately 170 speakers, and a separate set for evaluation/testing, consisting of 30 randomly selected speakers. The total amount of training data used was around 260 minutes, with the testing data around 45 minutes. The discussions and results from Van Heerden *et al*., will hereafter be referred to as the *Lwazi* results, for brevity.

Three other results in literature also require mention. These are the phone recognition results from three inter-related, yet distinct corpora called TIMIT, NTIMIT and STC-TIMIT, obtained by using an architecture similar to that used by Van Heerden *et al*., specifically 39-dimensional MFCCs, CMN and 15 Gaussian mixtures [66].

The TIMIT corpus is a broadband (8 kHz bandwidth) corpus, consisting of 630 speakers, each reading ten phonetically rich sentences in various dialects of American English [67]. The NTIMIT corpus employs the same audio files as TIMIT, but with the audio files sent through a Public Switched Telephone Network (PSTN) to simulate varying telephone channel effects. The STC-TIMIT corpus also employed the same audio files as TIMIT, but instead of sending these files through a PSTN, resulting in variations in channel conditions, these files were sent via an internal telephone line through a digital switchboard. STC-TIMIT employs a Single Telephone Channel (STC) and also eliminates the artificial mouth used during the creation of the related NTIMIT corpus. Implicit in this description is the fact that these three corpora share the same orthographic transcriptions as well as the same speaker statistics [66].

However, due to the phonotactics used to construct the phonetically rich sentences of TIMIT by constraining the permissible combinations of phonemes, results obtained by using these corpora are not directly comparable to results from experiments in this study [68].

A summary of the input data and relevant results in literature is shown in Table 5.1 and a comparison with the TIMIT and its derived corpora.

Table 5.1: *English phone recognition results in literature for comparison.*

Corpus	**Total training data (minutes)**	**Training speakers (male/female)**	**Correctness (%)**	**Accuracy (%)**
Lwazi	260	85/85	62.51	54.26
TIMIT	237	326/136	75.71	71.61
NTIMIT	237	326/136	64.07	55.73
STC-TIMIT	237	326/136	69.52	62.15

5.3 OVERVIEW OF EXPERIMENTS

A number of experiments were conducted in order to validate the data collected with Woefzela. The architecture for all experiments are kept the same to facilitate easy comparison. The purpose and numbering of each experiment is summarised in Table 5.2.

In order to provide incremental results on the evaluated corpus, Experiments B, C, D, E and F gradually increased the amount of data used to train the recogniser. This also provides the opportunity to confirm the trend that an increasing amount of data leads to an increasing recognition accuracy [8, 21].

Table 5.2: *Experiment purpose and numbering summary.*

Experiment	Purpose
A	Establish primary point of comparison with results for Lwazi corpus
B	Establish broadband point of comparison with results for the Lwazi corpus
C	Increase amount of data to establish intermediate performance (step 1)
D	Increase amount of data to establish intermediate performance (step 2)
E	Increase amount of data to establish intermediate performance (step 3)
F	Establish new phone-recognition benchmark for the full NCHLT English corpus

In the first experiment a phone-based recogniser is built and trained with the same amount of data as that used for the Lwazi corpus, in order to establish the main point of validation. Since a telephone channel was used to record the Lwazi corpus, while direct recording from the microphone of the smartphone was used in Woefzela, the data for the first experiment was down-sampled from having an 8 kHz bandwidth to a 4 kHz bandwidth to facilitate a more accurate comparison.

In the second experiment the same amount of data is used, but in its original broadband (8 kHz) format. In the subsequent four experiments, the broadband data continued to be used, but the amount of data per speaker was systematically increased, in order to evaluate the impact of increasing the amount of data on the recognition accuracy. The final experiment therefore uses all of the available data to establish a new phone-recognition benchmark for the NCHLT English corpus collected with Woefzela.

In the following section all the common elements of the experimental architectures, as well as the common data selection procedures, are described.

5.3.1 RECOGNISER ARCHITECTURE

The recogniser architectures used in the various experiments are summarised in Table 5.3 with specific reference to the work done by Van Heerden *et al*. [65] in 2010 on the Lwazi corpus.

In order to perform phone recognition, a flat phone-based language model was constructed by assuming a uniform probability distribution of all mono-phones, while the pronunciations for all words were indexed in a pronunciation dictionary.

An initial, dictionary, called SAEDICT, was used to find pronunciations of words in transcriptions while additional pronunciations for words not in this initial dictionary, were predicted using a set of grapheme-to-phoneme rules trained on a South African English dictionary (SAE_OALD), by using an algorithm called *Default&Refine* [69, 70].

Table 5.3: *Recogniser architectures used in experiments. 'Mic' implies the built-in microphone on the mobile phone.*

Component	**Lwazi corpus**	**Experiments**	
		A	**B to F**
	INPUT		
Bandwidth	4 kHz	4 kHz	8 kHz
Channel	telephone	mic	mic
	RECOGNITION TYPE		
Language model	flat phone-based	✓	✓
	ACOUSTIC MODELS		
Statistical model	HMM-based	✓	✓
Features	13 MFCC+Δ + $\Delta\Delta$	✓	✓
CMVN	per-speaker clusters	✓	✓
	OUTPUT		
GMM mixtures	7	7	7
Diagonal cov matrix	Yes	✓	✓
Model transforms	semi-tied	✓	✓

5.3.2 LANGUAGE SELECTION

A brief discussion of the language chosen for this tool validation is in order. Any language corpus collected with Woefzela could in principle be used for this study, but for the following reasons South African English was selected as the target language:

- With regard to evaluating the data collection methodology, any of the recorded languages will suffice to validate this tool, since the functionality of Woefzela is not language dependent.
- By performing evaluation on a non-tone language, additional variables are not introduced that may or may not influence recognition results.
- In order to enable a wider comparability with current and future literature, South African English as a variant of English is preferred, although all corpora differ to some extent.
- Since phone recognition is compared, the actual grammar of the target language is irrelevant.

Given the above rationale, nothing precludes any of the recorded languages to be used in any future evaluations.

5.3.3 INITIAL SPEAKER FILTERING

From all available data in the English corpus collected by Woefzela, only files that passed all of the quality control criteria listed in Section 3.3.1.2 (page 25), were considered candidate files. By aggregating the amount of candidate data available per speaker, only speakers that had an amount of data (measured in parts of seconds) that was within two standard deviations from the mean of all of the candidate speakers, were further considered. This lead to a final list of speakers, called the *input speaker list* for brevity.

5.3.4 TRAINING AND TEST SET SELECTION

The *input speaker list* was divided into three mutually exclusive groups, namely, training, development and evaluation speakers. The fact that all speakers are only present in one of the three sets, ensured that speaker-independent recognition could be performed, since no data of any speaker will be present in the training and test sets. *This selection of speakers was kept constant throughout all experiments*.

In Table 5.4 these sets are shown, with the Lwazi experiments (Section 5.2) also listed for comparison. As Barnard *et al*. [25] has shown, above 50 speakers, only the amount of data per speaker plays a significant role, and as such the difference between 80 and 85 speakers in this comparison is negligible [5, 27].

Table 5.4: *Division of speakers among the training, development and evaluation data sets. A comparison with Lwazi experiments is also shown.*

Experiments	**Data set**	**No. of male speakers**	**No. of female speakers**
Experiments A to F	Training set	80	80
	Development set	10	10
	Evaluation set	10	10
	Total testing audio time (min)	405 (6.75 h)	
Lwazi results	Training set	85	85
	Evaluation set	15	15
	Total testing audio time (min)	45 (0.76 h)	

5.3.4.1 INDIVIDUAL FILE SELECTION FOR TRAINING SET

In order to achieve the target amount of total training minutes required for each experiment, this target duration was divided by the number of speakers in the training set, leading to a target duration per speaker. This was to ensure that no speaker is significantly over or under represented in the resulting acoustic models.

In order to approximate the target number of seconds per speaker as closely as possible, all files associated with a specific speaker were ordered by duration, followed by an accumulation of the total

duration for the speaker. At the point that the accumulated duration was the closest to the target number of seconds, the files were marked, and only those files aggregating to this total duration, were selected.

This file selection process has two implicit properties: Firstly, the same files selected with one experiment would subsequently be selected for all experiments having a higher per-speaker target. Secondly, since shorter prompts are typically associated with shorter audio files, these files will be selected first before longer files are considered.

5.3.4.2 METHOD OF REPORTING EXPERIMENTAL RESULTS

When reporting recognition results, HTK [55] employs a dynamic programming algorithm to optimally align two strings, similar to computing the minimum Levenshtein distance by inserting, deleting or substituting characters[1]. The first string in this instance is the 'correct' transcription of an audio file as assumed from the prompt given to the Respondent to read; typically called the reference transcription. The second string is the result string as proposed by the Viterbi algorithm when decoding the actual recorded audio file [26].

After HTK has found such a *shortest distance match*, it calculates the recognition correctness and accuracy for each utterance, combining all the results for an overall correctness and accuracy score of the complete test set. With D being the number of *deletions*, S being the number of *substitutions*, I the number of *insertions*, and N the total number of tokens in the reference transcriptions, the correctness is calculated as shown in Eq. 5.1, and the accuracy as shown in Eq. 5.2.

$$\text{Percent Correct} = \frac{N - D - S}{N} \times 100 \tag{5.1}$$

$$\text{Percent Accurate} = \frac{N - D - S - I}{N} \times 100 \tag{5.2}$$

While the calculation of the accuracy also takes into account the number of insertions, both ways of expressing recognition results are useful, and therefore often found in literature.

5.4 EXPERIMENT A: PRIMARY VALIDATION WITH RESULTS IN LITERATURE

The first experiment matches the architecture as well as the amount of training data used by Van Heerden *et al*. [65] as closely as possible. In order to achieve this point of comparison, the training and test data was band limited to a 4 kHz bandwidth rather than the original form of the Woefzela data having an 8 kHz bandwidth.

While Van Heerden *et al*. used 258 minutes of training data split across 170 speakers, in Experiment A, this total number of minutes was divided across 160 speakers, thus leading to the target of 97

[1]The *characters* in this case are the phonemes as extracted from a given pronunciation dictionary

seconds per speaker, instead of the 91 seconds per speaker for the Lwazi experiments.

By establishing this point of comparison, the data produced by Woefzela, and thus the Woefzela tool itself, will already be shown as useful for developing ASR systems. However, other interesting results are still pursued in further experiments in order to establish the performance of the full acquired data set.

5.4.1 INPUT DATA

The input data used for Experiment A, alongside that of Van Heerden *et al*. [65], is shown in Table 5.5.

Table 5.5: *Experiment A band-limited input data compared to Van Heerden* et al. *[65]*.

Data	**Van Heerden *et al*.**		**Experiment A**	
	male	**female**	**male**	**female**
No. of training speakers	85	85	80	80
No. of testing speakers	15	15	10	10
Total training audio time (min)	258 (4.31 h)		259 (4.32 h)	
Total testing audio time (min)	45 (0.76 h)		405 (6.75 h)	

5.4.2 RESULTS

The results obtained from Experiment A is contrasted with Van Heerden *et al*. [65] in Table 5.6. From this comparison it can be seen that the recognition accuracy obtained using the data acquired by means of Woefzela, is not significantly different from the accuracy reported in Van Heerden *et al*. On the one hand this extremely close correspondence should be seen as incidental as these corpora differ in various regards, yet it still sufficiently establishes the initial point of reference for the newly collected corpus.

Table 5.6: *Experiment A results compared to Van Heerden* et al. *[65]*.

Results	**Van Heerden *et al*.**	**Experiment A**
Correctness (%)	62.51	62.99
Accuracy (%)	54.26	54.19

The following experiment, Experiment B, aims to duplicate the architecture and data used for Experiment A, with the only difference being that the data used is no longer band-limited, but the original 8 kHz bandwidth data as recorded by Woefzela.

5.5 EXPERIMENTS B TO F: BROADBAND WOEFZELA DATA

The purpose of Experiment B to Experiment F is two-fold. Experiment B seeks to establish a comparison point with Experiment A, but with the original broadband data collected with Woefzela, since Experiment A band-limited the data to a 4 kHz bandwidth for earlier comparison to Lwazi results.

Experiments C to F intend to investigate the impact of increasing the amount of training data per speaker on the recognition performance of ASR systems developed with Woefzela data.

5.5.1 INPUT DATA

Table 5.7 lists the theoretical target number of seconds per speaker alongside the actual achieved training minutes for the whole training set, for each experiment. Due to the discrete length of audio files, this target per speaker cannot always be achieved, leading to a less regularly spaced total training set duration as seen in the table.

Table 5.7: *Target amount of seconds per training speaker for all experiments alongside the total amount of training data in each training set. Lwazi corpus is shown for comparison.*

Experiment	**Target seconds per speaker**	**Total training set minutes (hours)**
Experiment A	97	259 (4.32)
Experiment B	97	259 (4.32)
Experiment C	350	933 (15.55)
Experiment D	631	1,671 (27.85)
Experiment E	1166	2,881 (48.02)
Experiment F	1700	3,239 (54.00)
Lwazi corpus	91	258 (4.31)

The results from Experiments B to F, combined with previous results, will be discussed in the next section.

5.6 SUMMARY AND DISCUSSION OF RESULTS

A summary of all the experimental results as well as results from relevant literature, is shown in Table 5.8 and graphically depicted in Figure 5.1.

Table 5.8: *Summary of results from all experiments compared with results in literature.*

Experiment	**Amount of training data minutes (hours)**	**Recognition**	
		Correctness (%)	**Accuracy (%)**
Experiment A	259 (4.32)	62.99	54.19
Experiment B	259 (4.32)	65.12	56.08
Experiment C	933 (15.55)	74.30	66.83
Experiment D	1,671 (27.85)	77.58	70.40
Experiment E	2,881 (48.02)	80.18	73.79
Experiment F	3,240 (54.00)	80.35	74.23
Lwazi	258 (4.31)	62.51	54.26

A number of interesting observations can be made from Table 5.8 and Figure 5.1:

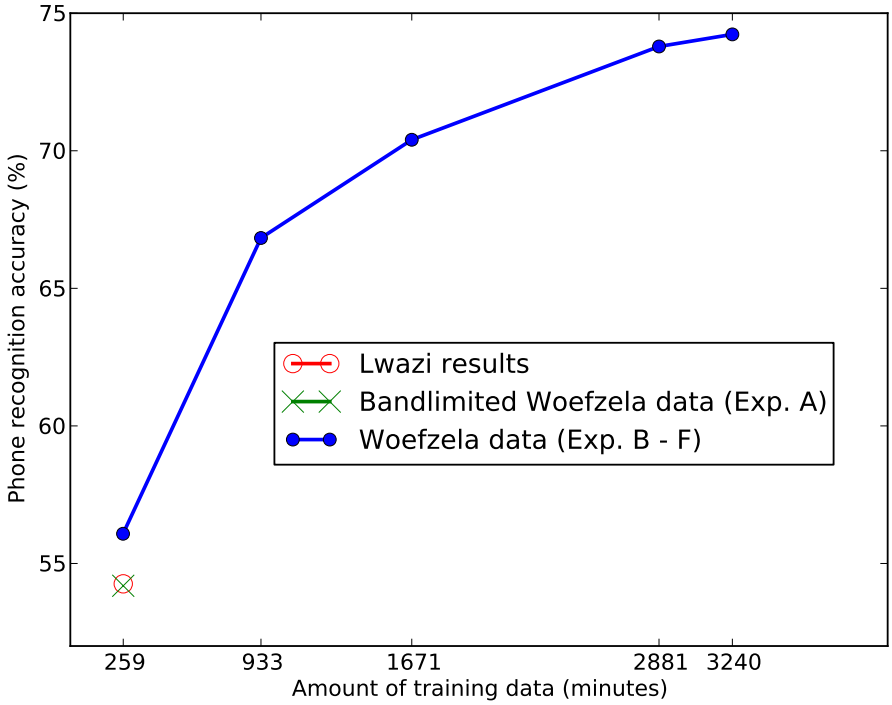

Figure 5.1: *Graphical summary of all phone recognition results.*

1. The comparison between the Lwazi accuracy achieved by Van Heerden *et al.* [65] and that of the primary validation experiment, Experiment A, serves as evidence that the data collected with Woefzela performs not significantly different from that of the Lwazi data set. As previously mentioned, the significance of the close match between these results should not be over-estimated as these corpora differ in various regards, yet this comparison still sufficiently establishes the usefulness of this data for developing ASR systems for under-resourced languages.

2. In comparing the results from Experiment B with Experiment A, an absolute increase of +1.89% is observed simply by using broadband data (Experiment B) instead of band-limited data (Experiment A). This result is not unexpected, but demonstrates the advantage of using broadband data, instead of one of the well established methods of data collection by using telephone networks, as mentioned in Section 2.2.1.3 in the literature survey.

3. The interesting trend formed by Experiments B to F as shown by the blue line in Figure 5.1, corroborates with the evidence in literature, that by increasing the amount of training data, the recognition accuracy also improves [8, 21].

   Although the absolute accuracy of the results obtained from Experiments C, D and E may vary slightly due to the explicit ordering of audio files according to size, which was not further investigated, this trend in itself confirms the consistency of this set of experiments, with the

results of Experiments B and F being the most relevant for this study.

4. The last observation relates to the new point of reference established by Experiment F. These results thus form a benchmark for the NCHLT English Corpus collected with Woefzela.

5.7 CONCLUSIONS

The following conclusions can be drawn from this chapter:

1. Woefzela is effective in collecting speech data for ASR system development; including for under-resourced languages.

2. The impact of collecting broadband data to improve phone recognition accuracy was demonstrated.

3. By using additional training data, whilst keeping the number of speakers constant, recognition performance is systematically improved. Thus *more* data is indeed better.

4. A new benchmark for the performance of the NCHLT English corpus was established.

CHAPTER SIX

CONCLUSION

6.1 INTRODUCTION

The objectives of this study as listed in Section 1.2, page 6, were to:

- Develop a software tool that is effective for collecting and annotating ASR data for under-resourced languages
- Verify that the requirements for such a tool have been met
- Validate the data produced by this tool by building an automatic speech recogniser

In this regard, Chapter 3 described the process of developing secondary requirements for such a tool, and realising these requirements through a design and implementation phase. Chapter 4 showed a comparison between the intended requirements and implemented functionality, providing more insights into certain design decisions. In Chapter 5, a number of ASR experiments were conducted, showing that the data produced by Woefzela is indeed usable for developing real ASR systems.

6.2 CONCLUSIONS

An analysis of the QC-on-the-go philosophy discussed in Section 4.12.2 showed that an increased amount of usable data can be collected from Respondents by implementing a semi-real-time algorithm on a mobile device. This maximises recording opportunity, a key attribute of effective ASR data collection for under-resourced languages. Also in Chapter 4, it was shown that the usability of Woefzela resulted in successfully recording more than 700 hours of speech data for the eleven official languages of South Africa.

The experiments in Chapter 5 showed that apart from the usability of Woefzela's data, by using data with a wider bandwidth, recognition performance could be improved. This could impact on the data collection strategy used for future data collection campaigns. Furthermore, by increasing the amount of data per speaker without increasing the number of speakers, a substantial increase in performance could be achieved.

6.3 SUMMARY OF CONTRIBUTIONS

The contributions made through this study were to:

- Demonstrate the effectiveness of this data collection methodology in collecting ASR data for under-resourced languages, in resource-constrained environments.
- Provide a free, open-source software tool for effective ASR data collection for under-resourced languages, with specific benefits in terms of maximising recording opportunity, providing support for Field Workers and data collection Contractors, and simplifying the post-processing of data.
- Introduce a novel semi-real-time QC-on-the-go philosophy which increases the amount of usable ASR data collected from speakers – without any dependence on external connectivity [63].
- Analyse the NCHLT English data collected with Woefzela, establishing benchmark phone recognition results for this corpus.

6.4 SUGGESTIONS FOR FUTURE RESEARCH

A number of ideas for future research can be proposed. A major focus of further development of this prototype could be the research and development of quality control criteria used in the semi-real time QC-on-the-go process. This study only provided arbitrarily chosen initial simple criteria as a starting point for such research, but various other advanced audio quality measurements and even basic signal-to-noise ratios could be proposed and evaluated.

Furthermore, through analysing the types and frequency of errors made by Respondents by selecting different subsets of the data based on the current quality criteria, or any newly proposed criteria, the impact of each criterion could be better understood. This may lead to revised or improved criteria and further investigation into the computational feasibility of implementing proposed criteria on smartphones with limited resources.

Also, a better understanding of the impact of these criteria on actual ASR system performance may lead to the redesign of the user interface, potentially interrupting a Respondent during a Session if an accumulation of specific errors reaches a critical point. Also, by analysing the number of errors made by a Respondent over time, the potential impact of fatigue on data quality could be better understood.

REFERENCES

[1] Etienne Barnard, Johan Schalkwyk, Charl van Heerden, and Pedro J. Moreno, "Voice Search for Development," in *Proc. INTERSPEECH*, Makuhari, Japan, September 2010, pp. 282–285.

[2] 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.

[3] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[4] Y.H. Sung, M. Jansche, and P.J. Moreno, "Deploying Google Search by Voice in Cantonese," *research.google.com*, pp. 1–4, 2011.

[5] E. Brewer, M. Demmer, B. Du, M. Ho, M. Kam, S. Nedevschi, J. Pal, R. Patra, S. Surana, and K. Fall, "The case for technology in developing regions," *Computer*, vol. 38, no. 6, pp. 25–38, May 2005.

[6] M. Kamvar and D. Beeferman, "Say What? Why users choose to speak their web queries," in *Proc. INTERSPEECH*, Makuhari, Japan, September 2010, pp. 1966–1969.

[7] M. Schuster, "Speech recognition for mobile devices at Google," *PRICAI 2010: Trends in Artificial Intelligence*, pp. 8–10, 2010.

[8] Jiulong Shan, Genqing Wu, Zhihong Hu, Xiliu Tang, Martin Jansche, and P.J. Moreno, "Search by Voice in Mandarin Chinese," in *Eleventh Annual Conference of the International Speech Communication Association*, 2010, pp. 354–357.

[9] J. Sherwani, N. Ali, S. Mirza, A. Fatma, Y. Memon, M. Karim, R. Tongia, and R. Rosenfeld, "HealthLine: Speech-based access to health information by low-literate users," *International Conference on Information and Communication Technologies and Development*, pp. 1–9, December 2007.

[10] Aditi Sharma Grover, Madelaine Plauche, Etienne Barnard, and Christiaan Kuun, "HIV health information access using spoken dialogue systems: Touchtone vs. speech," *International Conference on Information and Communication Technologies and Development (ICTD)*, pp. 95–107, April 2009.

[11] Aditi Sharma Grover and Etienne Barnard, "The Lwazi Community Communication Service: Design and Piloting of a Voice-based Information Service," in *Proceedings of the 20th International Conference on World Wide Web (WWW 2011)*, Hyderabad, India, March 2011, pp. 433–442.

[12] Etienne Barnard, Laurens Cloete, Hina Patel, and Louis Coetzee, "Towards effective telephone-based delivery of government services," in *9th International Winelands Conference, Stellenbosch*, 2003.

[13] Swaran Lata and Somnath Chandra, "Development of Linguistic Resources and Tools for providing multilingual Solutions in Indian Languages - A Report on National Initiative," in *Proceedings of the Seventh Conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May 2010, pp. 2851–2854.

[14] Etienne Barnard, M. Davel, and G. van Huyssteen, "Speech Technology for Information Access: a South African Case Study," in *AAAI Symposium on Artificial Intelligence*, 2010.

[15] Febe de Wet, Pippa Louw, and Thomas Niesler, "The design, collection and annotation of speech databases in South Africa," *Proceedings of the Pattern Recognition Association of South Africa (PRASA 2006)*, 2006.

[16] T. Schultz and K. Kirchhoff, *Multilingual Speech Processing*, Academic Press, 2006.

[17] T. Schultz and A. Waibel, "Experiments on cross-language acoustic modeling," in *Proc. Eurospeech*, 2001, vol. 54, pp. 2721–2724.

[18] Charl van Heerden, Neil Kleynhans, Etienne Barnard, and Marelie Davel, "Pooling ASR data for closely related languages," in *Proceedings of the Workshop on Spoken Languages Technologies for Under-Resourced Languages (SLTU 2010)*, Penang, Malaysia, May 2010, pp. 17–23.

[19] Marelie H. Davel, Charl van Heerden, Neil Kleynhans, and Etienne Barnard, "Efficient harvesting of Internet audio for resource-scarce ASR," in *Proc. INTERSPEECH*, Florence, Italy, August 2011, pp. 3153–3156.

[20] Febe de Wet, Alta de Waal, and Gerhard B. van Huyssteen, "Developing a broadband automatic speech recognition system for Afrikaans," in *Proc. INTERSPEECH*, Florence, Italy, August 2011, pp. 3185–3188.

[21] J. Baker, Li Deng, Sanjeev Khudanpur, Chin-hui Lee, James Glass, Nelson Morgan, and D. O'Shaughnessy, "Research Developments and Directions in Speech Recognition and Understanding, Part 1," *Signal Processing Magazine, IEEE*, vol. 26, no. 4, pp. 78–85, 2009.

[22] Steven Abney and Steven Bird, "The Human Language Project : Building a Universal Corpus of the World's Languages," in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 2010, pp. 88–97.

[23] Mike Maxwell and Baden Hughes, "Frontiers in linguistic annotation for lower-density languages," in *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora (LAC'06)*, 2006, pp. 29–37.

[24] Aditi Sharma Grover, Gerhard B. van Huyssteen, and Marthinus W. Pretorius, "The South African Human Language Technology Audit," *Language Resources and Evaluation*, vol. 45, no. 3, pp. 271–288, June 2011.

[25] Etienne Barnard, Marelie Davel, and Charl van Heerden, "ASR Corpus Design for Resource-Scarce Languages," in *Proc. INTERSPEECH*, Brighton UK, September 2010, pp. 2847–2850.

[26] M. Gales and S. Young, "The Application of Hidden Markov Models in Speech Recognition," *Foundations and Trends in Signal Processing*, vol. 1, no. 3, pp. 195–304, 2008.

[27] Jaco Badenhorst, Charl van Heerden, Marelie Davel, and Etienne Barnard, "Collecting and evaluating speech recognition corpora for 11 South African languages," *Language Resources and Evaluation*, vol. 45, no. 3, pp. 289–309, June 2011.

[28] M.H. Davel and E. Barnard, "Bootstrapping for language resource generation," in *Proc. Pattern Recognition Association of South Africa (PRASA)*, Langebaan, South Africa, November 2003, pp. 97–100.

[29] J. Baker, Li Deng, Sanjeev Khudanpur, Chin-hui Lee, James Glass, Nelson Morgan, and D. O'Shaughnessy, "Updated MINDS report on speech recognition and understanding, Part 2," *Signal Processing Magazine, IEEE*, vol. 26, no. 4, pp. 78–85, 2009.

[30] E. Brewer, M. Demmer, M. Ho, R.J. Honicky, J. Pal, M. Plauche, and S. Surana, "The Challenges of Technology Research for Developing Regions," *IEEE Pervasive Computing*, vol. 5, no. 2, pp. 15–23, April 2006.

[31] A. Pentland, R. Fletcher, and A. Hasson, "DakNet: rethinking connectivity in developing nations," *Computer*, vol. 37, no. 1, pp. 78–83, Jan 2004.

[32] Madelaine Plauche, Udhyakumar Nallasamy, Joyojeet Pal, Chuck Wooters, and Divya Ramachandran, "Speech Recognition for Illiterate Access to Information and Technology," *International Conference on Information and Communication Technologies and Development*, pp. 83–92, May 2006.

[33] Denis Burnham, Dominique Estival, Steven Fazio, Jette Viethen, Felicity Cox, Robert Dale, Steve Cassidy, Julien Epps, Roberto Togneri, Michael Wagner, Yuko Kinoshita, Roland Göcke,

Joanne Arciuli, Marc Onslow, Trent Lewis, Andy Butcher, and John Hajek, "Building an audio-visual corpus of Australian English: large corpus collection with an economical portable and replicable Black Box," in *Proc. INTERSPEECH*, Florence, Italy, August 2011, pp. 841–844.

[34] David Graff and Steven Bird, "Many Uses, Many Annotations for Large Speech Corpora: Switchboard and TDT as Case Studies," in *Proceedings of the 2nd International Conference on Language Resources & Evaluation*, April 2000, pp. 427–433.

[35] A. Moreno, B. Lindberg, C. Draxler, G. Richard, K. Choukri, S. Euler, and J. Allen, "Speechdat-car: A large speech database for automotive environments," in *Proceedings of the 2nd International Conference on Language Resources & Evaluation*, April 2000.

[36] J. S. Garofolo, Lori F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "The DARPA TIMIT Acoustic-Phonetic Continuous Speech Corpus CDROM, NIST order number PB91-100354," February 1993.

[37] Tanja Schultz, "Globalphone: a Multilingual Speech and Text Database Developed at Karlsruhe University," in *Proc. International Conference on Spoken Language Processing (ICSLP)*, Denver, Colorado, USA, September 2002.

[38] P. Bauer, D. Scheler, and T. Fingscheidt, "WTIMIT: The TIMIT Speech Corpus Transmitted Over the 3G AMR Wideband Mobile Network," in *Proc. Language Resources and Evaluation*, Valletta, Malta, May 2010, pp. 1566–1570.

[39] Gabriel Parent and Maxine Eskenazi, "Speaking to the Crowd: looking at past achievements in using crowdsourcing for speech and predicting future challenges," in *Proc. INTERSPEECH*, Florence, Italy, August 2011, pp. 3037–3040.

[40] I. McGraw, C. Lee, L. Hetherington, S. Seneff, and J. Glass, "Collecting voices from the cloud," in *Proc. LREC*, 2010, pp. 19–21.

[41] C. Callison-Burch and M. Dredze, "Creating speech and language data with Amazon's Mechanical Turk," in *Proc. NAACL HLT*. Association for Computational Linguistics, 2010, pp. 1–12.

[42] H. Gelas, S.T. Abate, L. Besacier, and F. Pellegrino, "Quality assessment of crowdsourcing transcriptions for African languages," in *Proc. INTERSPEECH*, Florence, Italy, August 2011, pp. 3065–3068.

[43] K. Audhkhasi, P.G. Georgiou, and S.S. Narayanan, "Reliability-Weighted Acoustic Model Adaptation Using Crowd Sourced Transcriptions," in *Proc. INTERSPEECH*, Florence, Italy, August 2011, pp. 3045–3048.

[44] C. Lee and J. Glass, "A Transcription Task for Crowdsourcing with Automatic Quality Control," in *Proc. INTERSPEECH*, Florence, Italy, August 2011, pp. 3041–3044.

[45] T. Hughes, K. Nakajima, L. Ha, A. Vasu, P.J. Moreno, and M. LeBeau, "Building transcribed speech corpora quickly and cheaply for many languages," in *Proc. INTERSPEECH*, Makuhari, Japan, September 2010, pp. 1914–1917.

[46] Ian Lane, Alex Waibel, M. Eck, and K. Rottmann, "Tools for collecting speech corpora via Mechanical-Turk," in *Proc. NAACL HLT*. 2010, pp. 184–187, Association for Computational Linguistics.

[47] K. Maeda, H. Lee, S. Medero, J. Medero, R. Parker, and S. Strassel, "Annotation tool development for large-scale corpus creation projects at the linguistic data consortium," *Proceedings of the Sixth International Language Resources and Evaluation*, 2008.

[48] F. Alam, S.M.M. Habib, D.A. Sultana, and M. Khan, "Development of annotated Bangla speech corpora," in *Spoken Languages Technologies for Under-Resourced Languages*, Penang, Malaysia, May 2010.

[49] P. Skrelin, N. Volskaya, D. Kocharov, K. Evgrafova, O. Glotova, and V. Evdokimova, "A Fully Annotated Corpus of Russian Speech," in *Proc. Language Resources and Evaluation*, 2010, pp. 109–112.

[50] Damjan Vlaj, A.Z. Markuš, Marko Kos, and Zdravko Kačič, "Acquisition and Annotation of Slovenian Lombard Speech Database," in *Proc. Language Resources and Evaluation*, 2000, pp. 595–600.

[51] M.L. Glenn, S.M. Strassel, and H. Lee, "XTrans: A speech annotation and transcription tool," in *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[52] C. Van Bael, L. Boves, H. van den Heuvel, and H. Strik, "Automatic phonetic transcription of large speech corpora," *Computer Speech & Language*, vol. 21, no. 4, pp. 652–668, 2007.

[53] M. Ostendorf, "Transcribing human-directed speech for spoken language processing," in *Tenth Annual Conference of the International Speech Communication Association*, 2009.

[54] J.T. Milde and U. Gut, "A prosodic corpus of non-native speech," in *Proceedings of the Speech Prosody conference*, 2002, pp. 11–13.

[55] S. Young, G. Evermann, M. J. F. Gales, T. Hain, D. Kershaw, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, *The HTK Book, Version 3.4*, Cambridge University Engineering Department, Cambridge, UK, 2009.

[56] B.C. Roy, S. Vosoughi, and D. Roy, "Automatic Estimation of Transcription Accuracy and Difficulty," in *Proc. INTERSPEECH*, Makuhari, Japan, September 2010, pp. 1902–1905.

[57] H. van den Heuvel, D. Iskra, E. Sanders, and F. de Vriend, "Validation of spoken language resources: an overview of basic aspects," *Language Resources and Evaluation*, vol. 42, no. 1, pp. 41–73, 2008.

[58] M. A. Branstad, J. C. Cherniavsky, and W. R. Adrion, "Validation, Verification, and Testing for the Individual Programmer," *Computer*, vol. 13, pp. 24–30, December 1980.

[59] Alain Abran, James W. Moore, Robert Dupuis, R.L. Dupuis, and L.L. Tripp, "Guide to the Software Engineering Body of Knowledge," *SWEBOK*, 2004.

[60] "Android developers," [Accessed: 4 Nov 2011], Available from: `http://developer.android.com`.

[61] IEEE Std 1012-2004, *IEEE Std 1012-2004 IEEE Standard for Software Verification and Validation*, IEEE Computer Society, June 2005.

[62] Febe de Wet and Nina Titmus, "National Centre for Human Language Technology (NCHLT)," *Annual Management Report 1 April 2010 - 31 March 2011*, May 2011.

[63] Nic J. de Vries, Jaco Badenhorst, Marelie H. Davel, Etienne Barnard, and Alta de Waal, "Woefzela - An open-source platform for ASR data collection in the developing world," in *Proc. INTERSPEECH*, Florence, Italy, August 2011, pp. 3177–3180.

[64] "Woefzela," [Accessed: 28 Oct 2011], Available from: `https://sites.google.com/site/woefzela`.

[65] Charl van Heerden, Etienne Barnard, and Marelie Davel, "Basic speech recognition for spoken dialogues," in *Proc. INTERSPEECH*, Brighton UK, September 2010, pp. 3003–3006.

[66] N. Morales, J. Tejedor, J. Garrido, J. Colás, and D.T. Toledano, "STC-TIMIT: Generation of a Single-channel Telephone Corpus," in *Proc. LREC*, Marrakech, Morocco, 2008, pp. 391–395.

[67] "The Linguistic Data Consortium Corpus Catalog," [Accessed: 7 Nov 2011], Available from: `http://www.ldc.upenn.edu/Catalog`.

[68] J. Badenhorst, M.H. Davel, and E. Barnard, "Analysing co-articulation using frame-based feature trajectories," in *Proc. Pattern Recognition Association of South Africa (PRASA)*, Stellenbosch, South Africa, November 2010, pp. 13–18.

[69] M. Davel and E. Barnard, "Pronunciation prediction with Default&Refine," *Computer Speech and Language*, vol. 22, pp. 374–393, 2008.

[70] Linsen Loots, Marelie Davel, Etienne Barnard, and Thomas Niesler, "Comparing manually-developed and data-driven rules for P2P learning," in *Proc. Pattern Recognition Association of South Africa (PRASA)*, Stellenbosch, South Africa, November 2009, pp. 35–40.