SPEECH-ENABLED INTERFACES FOR TRAVEL INFORMATION SYSTEMS

BAOLI ZHAO

A thesis submitted in partial fulfillment of the requirements of Nottingham Trent University for the degree of Doctor of Philosophy

This research programme was carried out in collaboration with Nottingham City Transport

June 2006

ABSTRACT

This thesis is concerned with the development of a robust and efficient speech-enabled query interface for the travel information system. The approach taken is separated into three distinct processes: i) development of a directed-dialogue speech-enabled interface for a medium grammar based bus travel information system. This interface directs the user through a sequence of questions and answers to get the enquired result; ii) development of a multimodal interface that employs a mixed-initiative grammar to overcome the usability problems identified in the medium grammar directed-dialogue system. This interface allows the system to process a more natural language style of input rather than directing the user through a rigid sequence of questions and answers; iii) development of a directed-dialogue speech-enabled interface for an equivalent large grammar based bus travel information system that uses a novel method for real-time grammar segmentation and recognition.

This thesis firstly presents the dialogue design and usability evaluation of a directed-dialogue speech–enabled query interface for a bus travel information system. The evaluation, based on a usability-engineering paradigm, analyses four human factors of the user interface: effectiveness, efficiency, user satisfaction and learnability. The initial interface design contributes a baseline specification for the construction of a speech-enabled interface and a usability test method for other speech application developers. This evaluation also highlights the usability issues associated with the use of directed-dialogue and speech-only interfaces.

A mixed-initiative dialogue combined with a multimodal interface is then presented that successfully addresses all of the usability issues identified in the directed-dialogue interface. The good usability results reported for this improved interface show that the use of a mixed-initiative dialogue combined with a multimodal interface is an effective method for building a speech-enabled phone-based HCI system.

Finally, this thesis introduces a novel last-word recognition based grammar segmentation method that is used to handle the large grammar issues associated with producing a real-time bus travel application. Large grammars tend to produce relatively slow recognition interfaces and this work shows how this limitation can be successfully addressed. This investigation therefore contributes a method for designing real-time speech-enabled interfaces that need to use very large grammars.

ACKNOWLEDGMENTS

The research has been absolutely enjoyable. This enjoyment is from my supervisors, colleagues, friends, family and all the people who have cared for me.

I am very privileged to have as my supervisors Dr. Tony Allen and Professor Andrzej Bargiela. Dr Tony Allen met with me about once a week for the last three years to discuss the problem of my research. We were also joined by Professor Bargiela for some of the discussions. To each of them I owe a great debt of gratitude for their patience and inspiration. Without their supervision, I wouldn't done all of this. I also thank my thesis committee readers Dr. Jan Noyes and Dr. Evtim Peytchev for their suggestions.

I would also like to thank Nottingham Trent University. My university has provided an excellent environment and a wonderful bursary for my research. Many companies and organizations also provided me with invaluable research materials and equipments: Nuance, Infohub, and Nottingham City Transport.

Through the years of completing this research I have talked, thought, cried, laughed and enjoyed with many friends. So many have helped that I can not mention them all here. But if I had to name just a few, I would have to start with my flat mate Yaoyao Li who has cooked for me for two years. I am very glad that I have lived together with you. During my time in UK, I have also received incredible help and support from my dearest friends: Wenqing Zhuan, Nan Zhang, Teng Cao, Yuanyuan Zhai, Dan Wang, Wei Wang, Belinda Liu, Betty Sun, Chris Roland and his girlfriend Raquel. Thank you very much. I love you guys. I always believe that, with friends, you will never walk alone.

Special thanks go to my family who have been extremely understanding and supporting my studies. I feel that I am the luckiest person in this world to have such a warm family. I have no doubt that the nurture, care, and love given by my family makes me who I am today. Mummy, Daddy, Baoxia, Jiaojiao, Thank you very much; I appreciate it forever!

TABLE OF CONTENTS

Chapter 1 - Introduction

Chapter 2 – Literature Study

Chapter 3 – A Medium Vocabulary Directed-Dialogue Interface

Chapter 4 – A Mixed-initiative Dialogue Interface

Chapter 5 – Speech Interface with a Large Grammar

Chapter 6 – Conclusions and Further Research

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

This chapter describes the background and framework for this research and introduces the reader to the subject area and related knowledge.

1.1 Background

This research work is based on the previous research project ATTAIN (Advanced Traffic and Travel Information System) [NTU 2002] which is a mobile telephone based traffic information system that enables the travelling public to make enquiries, using a mobile telephone, about selected bus routes in the Nottingham city conurbation. The system includes over 140 bus routes and 1355 bus stops location references to provide a comprehensive coverage of the Nottingham city and selected out-of-city locations. This system was developed within the School of Computing and Informatics as part of an earlier collaboration with the Nottingham City Transport department. Currently, the ATTAIN system relies on text based messaging from mobile phones that responds to specific requests for bus time information. The provision of customised information prevents information overload while affording more thorough examination of alternatives that are pertinent to a specific journey. The requests are made by means of SMS (Short Message Service) text messages and the replies are provided in the same way. The requests for information must adhere to some simple rules to ensure that the messages are easily understood while the amount of text that needs to be entered is as small as possible. The general format of the 'request for information message' is as follows: [origin code] to [destination code] [before/after] [time], for example: Arnold to Beeston after 10.30. This restricts the use of the system to those people who own and can operate the text-based interface on a mobile phone. This research intended to develop a speech-enabled query interface for the existing ATTAIN system. This will immediately open up the system facilities to those people who are either not technically literate enough to efficiently operate the text facilities of a mobile phone or who only have access to land based phones. In addition, the provision of a speech-enabled interface alongside the existing text based interface will offer the technically literate, mobile phone owning ATTAIN system users a choice of input modalities.

In recent years, the use of speech and natural language interface technologies has significantly improved the usability of many computer based interactive applications. These include [Miller 2002]: driving navigation systems, traffic, weather and stock market information systems as well as telephone banking and email checking. There are many reasons for this new focus but one of the main reasons is the recent introduction of reasonably effective speaker-independent speech recognition technologies [Torre 2002]. Voice is a natural interface that the majority of people are capable of using without any technical training because it enables the user to speak and listen using skills learned during childhood [W3C 2000A]. Thus the creation of speaker-independent speech-enabled interface systems, especially one that can provide a natural language type of speech dialogue, are likely be of increasing benefit to users of computer-based information systems. However, the performance of the speech recognition algorithms is not the whole story. There is still the question of how speech recognition can and should actually be used in terms of the human factors associated with using speech based Human Computer Interactions (HCI). Related to this is the issue of tools for the development of speech-enabled interfaces. The speech input technology is available but the question of how to build effective speech-enabled interfaces still remains. This project investigated some of the methods and problems associated with building effective speech-enabled interfaces.

In May 2000, a voice programming standard was introduced that is endorsed by The W3C (World Wide Web Consortium). Known as VoiceXML (Voice eXtensible Markup Language) [W3C 2000B], this language is based on web standards and can be used to easily create speaker-independent speech-enabled applications. The latest version, VoiceXML 2.0 [W3C 2004], can create audio dialogues that feature synthesised speech, digitized audio, recognition of spoken and DTMF (Dual Tone Multi-Frequency) key input as well as being able to record spoken input, telephony and mixed-initiative conversations. It brings the advantages of web-based development and content delivery to interactive voice response applications. Another markup language that is currently proposed for developing multimodal web applications is the SALT (Speech Application Language Tags) language [Microsoft 2005A]. Developed by Microsoft, SALT is a set of extensions to existing markup languages, in particular HTML and XHTML. This language enables multimodal and telephony access to information, applications and Web services from PCs, telephones, and mobile devices. The applications developed using SALT can be implemented using the client model with speech processing done on the Microsoft Speech Server [Moraes 2002]. Microsoft has a speech SDK (Software Development Kit) for .NET developers, which uses SALT, so developers would need to be familiar with ASP.NET (Active Server Page) before attempting to work with SALT. VoiceXML is the larger of the two standards because it is a complete standalone markup specification whereas SALT depends more on existing functionality handled by other Web application specifications. VoiceXML also has the advantage of being a more widely supported standard, with more than 150 companies involved in the VoiceXML Forum [VoiceXML Forum 2005]. This may be because VoiceXML has roots in an earlier Web era (unlike SALT which is based in more modern Web development architecture). As a result, VoiceXML could be more appealing to developers with a background in traditional web or telephony applications. VoiceXML will be used as the developing language in this research project.

During the course of this research project, the ATTAIN travel information system has also been extended to apply to the London Metropolitan area where over 45,000 location references are needed to provide a comprehensive coverage of the London city locations. Compared with the speech-enabled ATTAIN interface for Nottingham (which only contains 1355 location references), the London speech-enabled ATTAIN interface will require far larger vocabularies and a variety of grammars. In order to support this extension, the dialogue of the speech-enabled interface must be optimised to handle grammar and lexicon elements that contain references to more sophisticated grammar and lexicon modules. The impact of this on dialogue management and system usability needed to be determined, and mechanisms for dealing with any differences devised.

1.2 Overview and Contributions

As discussed at 1.1, the ATTAIN travel information system is in need of an efficient and robust speech-enabled interface. However, current implementations of such technology still face problems in terms of their performance and usability factors. Speaker-independent (as opposed to speaker-dependent) continuous speech recognition is still relatively error prone [Torre 2002]. Consequently, all general purpose speech-enabled systems currently need to be designed to be tolerant of these errors. As a result, the first speech-enabled interface developed for the Nottingham travel information system used a directed-dialogue style of interaction [Zhao 2004A]. This development took into consideration aspects of user interface design and speech dialogue management including error recovery. The recognition performance of this interface is quite good for the user group tested. However, the usability test does indicate that the interaction style is not yet natural or fast enough for public acceptance. To overcome these critical difficulties it was necessary to enhance the underlying speech recognition accuracy and improve the dialogue management. That said, the initial interface design contributes a baseline specification for the construction of a speech-enabled interface and a usability test method for other speech application developers.

In order to achieve a higher system performance and a more "natural" interaction style in human-computer interaction, speech recognition has been research intensely over the past few decades. As cited by Oviatt [Oviatt 1991] and Cohen [Cohen 1995], speech is widely used for functions like describing objects and events, sets and subsets of objects, out-of-view objects, conjoined information, past and future temporal states, as well as for issuing commands for actions or iterative actions. The current research efforts in Human Computer Interaction (HCI) are moving towards using multimodal interfaces that employ the use of different input methods [Neti 2000] [Wang 1995]. Tan defined naturalness as the regular way, by which one human being would pass information to another human being (e.g. Person A tell his/her mobile number to person B) [Tan 2003]. It is to be noted that natural is different from familiar. Familiarity is defined as the most common way a particular task is carried out currently, for example, users are familiar with keyboard to input the data into computer. Noyes argued that if a human machine interaction is natural, it will meet the requirements of the usability framework (learning, effectiveness, attitude, flexibility) [Noyes 2001]. For example, in a talking to a machine application could be viewed as unnatural, and likewise, the task of issuing single word commands.

In this research project increasing the "natural" aspect of the interaction between the user and the Nottingham travel information system, the use of a second multimodal interface that employs a mixed-initiative grammar was investigated [Zhao 2004B]. This grammar allows the interface to process a more natural language style of input, rather than directing the user through a rigid sequence of questions and answers. The novel multimodal aspect of the new speech-enabled system still requires speech to be used to input the required journey information but now uses text messages, as well as audio feedback, to present the results of the search back to the user. This overcomes the human short-term memory problem present in the initial version of the interface. The good usability results reported indicate that the use of a mixed-initiative dialogue combined with a multimodal interface is an effective method for building a speech-enabled HCI system.

Based on the encouraging results obtained from the speech-enabled Nottingham travel information system, a directed-dialogue speech-enabled interface has been developed for an equivalent London travel information system. The initial version of this system used a large grammar file that simply contained all of the bus names. The experimental results show that the system took up to 13 seconds to process one bus stop name. Such a latency would obviously be unacceptable to users. A second version interface was therefore developed that used many small grammar files grouped according to the starting letter of the bus stop names. This system had to ask the user to speak the first letter of their origin or destination before asking the user to speak their origin or destination bus stop name. After the system had recognised the first letter of the bus stop names, the system could recognise the full bus stop names using the small grammar file that only contained bus stop names with the same start letter as the recognised first letter. Performing grammar segmentation in this way significantly changed the system's performance particularly in terms of processing time which now only takes 1.67-2.17 seconds to process one user's entry. Unfortunately, the necessity of obtaining this extra first letter information from the user introduced usability issues into the system. In human to human communication, asking people to speak the first letter of a word before trying to recognise a word is not a natural form of interaction.

To ensure that the user can naturally communicate with the system, the 'unnatural' questions needed to be removed. Ideally, if the system was able to record the user speaking the bus stop name, then the system could extract automatically the first letter from the recorded audio and thereby eliminating the need for the unnecessary communication. Consequently, a First Phoneme Processor has been investigated that, theoretically, should have been able to find automatically the first phoneme from a user's spoken input. Unfortunately, the first phoneme recognition results were not encouraging due, primarily, to the complexity of phoneme recognition. An alternative novel grammar reduction technique was therefore developed that is based on the idea of dividing the large London grammar into smaller grammar files grouped according to the end word of a bus stop name. It was anticipated that this approach would work better than the first phoneme approach because the sound segments of words are longer than those of phonemes and the end word sounds are fairly distinct. A Last Word Processor has been developed which can find automatically the last word of a user's spoken input. The experiment results show that the user perceived latency is now 4.01-5.94 seconds. Because the last word recognition system does not need to ask any 'unnatural' questions, the users can naturally communicate with the system so recovering the previous version's good usability ratings. This last set of experimental results thus show that the last word based grammar reduction method is more effective (accurate) and efficient (faster) than either an equivalent first phoneme based grammar reduction system or, indeed, a single grammar based system. This investigation therefore contributes a method for designing real-time speech-enabled interfaces that need to use very large grammars.

1.3 Outline of the Thesis

This thesis consists of 6 chapters which are as follows:

Chapter 1 gives a general introduction to the subject area within which this project is set as well as outlining the motivation for this work and discussing the problems associated with building a speech-enabled interface. This chapter also summarises the original contributions achieved by this work.

Chapter 2 presents a literature study of previous work in the field of speech technology including speaker-independent speech recognition, grammar production, and speech synthesis technologies. The problems associated with speech dialogue management, error recovery, multimodal interfaces and usability studies are also introduced in this chapter.

Chapter 3 provides a detailed discussion regarding the ways in which speech can be utilised to overcome the limitations of the current ATTAIN system. Issues surrounding the design of a directed-dialogue speech-enabled query interface using VoiceXML and how the dialogue management and error recovery was organised are discussed. Finally experimental results showing the usability and HCI factors of this prototype interface are given.

Chapter 4 presents the idea of building a multimodal interface that uses a mixed-initiative grammar. This grammar allows the interface to process natural language input, rather than directing the user through a rigid sequence of questions and answers. The novel multimodal aspect of this new speech-enabled system is shown to overcome the human short-term memory problem inherent in any speech-only interactive HCI system. This chapter then presents the speech recognition accuracy testing and usability evaluation of this mixed-initiative dialogue multimodal interface for the ATTAIN travel information system.

Chapter 5 considers how speech-enabled interfaces can deal with the problems of using a very large grammar. Two novel methodologies, the automatic recognition of the first phoneme and last word from a user's speech input, are presented. A comparison between the method of grammar sub-division based on first letter recognition and the method of grammar sub-division based on last word recognition is presented. The method of grammar sub-division based on recognising automatically the last word is shown to be a reliable way of dealing with large grammars in real-time constrained speech-enabled interfaces.

Chapter 6 presents the final remarks and conclusions of this project as well as discussing the potential avenues for further work.

CHAPTER 2
LITERATURE STUDY

In this chapter, a detailed discussion on the disciplines of speech technology, speech dialogue management, error recovery, multimodal interface and speech-enabled systems usability is presented.

Section 2.1 discusses the techniques and problems associated with speaker-independent speech recognition, grammar production, speech synthesis and the advantages and applications of natural speech input methods, together with the work done to handle these issues.

Sections 2.2 and 2.3 discuss speech dialogue management and the different error recovery strategies that can be employed to deal with the problems of speech recognition errors. These sections also explain the different types of dialogue that can be used to deal with different user groups and different speech applications.

Section 2.4 surveys prior research and commercial projects for speech-enabled systems. The hands-free dialling of mobile phones, personalised driving directions, and similar services through telephone interaction with a computer are introduced. Flight reservation, control of audio, climate and phone accessories in a car by spoken commands as well as the use of voice commands in video game playing are also surveyed.

Section 2.5 will provide a discussion on multimodal interfaces, which includes the definition, advantages, example systems and use of multimodal interfaces to carry out error recovery processes in a speech-enabled system.

Finally, Section 2.6 provides a literature survey on usability, in terms of its definitions, measures and the reasons for selecting each usability metric used in this work.

2.1 Speech Recognition Technology

In 1952, through the research work carried out at Bell Lab by Davis and others [Davis 1952], a fully operational speech recognition system that could recognise isolated digits spoken over a telephone by a single user was developed. Since then, intensive research into automatic speech recognition by machine has resulted in research based speech recognition systems that now have the capability to deal with continuous dictation without have the user having to speak in a discrete manner. These same systems are also capable of handling large vocabularies. Minami has successfully developed an accurate and efficient algorithm for very-large-vocabulary continuous speech recognition based on an HMM (Hidden Markov Models) algorithm which has been applied to a telephone directory assistance system that recognises spontaneous speech containing the names and addresses of more than 70,000 subscribers (vocabulary size is about 80,000) [Minami 1993].

Over the last decade, a number of commercial systems have been successfully developed. NeuVoice uses its noise-robust small-footprint speech recognition engine ported to the Nokia 9200 series communicators as a Voice Dialler [NeuVoice 2005]. This Voice Dialler has a quoted accuracy above 98% in a quiet environment but, being essentially a speaker-dependent system, it has to ask the user to provide a speech template for each word that the dialler is required to remember. ViaVoice is a desktop dictation speech recognition engine that has been developed by IBM (International Business Machines). ViaVoice has quoted recognition rates of up to 96% with speaker-dependent continuous speech [IBM 2005], but takes between 20 and 40 minutes to train depending on the reviewer. Despite these advances, true natural language speech interfaces are still several years away. This is mainly due to the inability of current speech recognition systems to deal successfully with speaker-independent continuous speech recognition. Current speech-enabled systems overcome this limitation by getting the user to train the recognition engine to their voice.

Speaker-dependent systems must be trained by each individual user, but typically have much higher accuracy rates than speaker-independent systems [Davis 1996]. Speaker-independent systems, on the other hand, have the ability to recognise speech from any speaker without any training. The typical achievable recognition rate for large-vocabulary speaker-independent systems is claimed to be about 80%-90% for a clear environment, but can be as low as 50% for scenarios like cellular phone with background noise [Wikipedia 2005]. In addition to the problems associated with making interfaces speaker-independent, speech-enabled systems also have to contend with the problems associated with continuous, large vocabulary speech recognition. Continuous speech systems can recognise words spoken in a natural rhythm rather than isolated words. Non-continuous speech systems require a considered pause between each word. Although continuous systems are more desirable, continuous speech is harder to process, because of the difficulty in detecting word boundaries [Grasso 1997]. Vocabulary size can also vary from 2 words to more than 40,000 words. Large vocabularies have more errors in speech recognition accuracy, but small vocabularies can force unwanted restrictions (out of vocabulary error) on the naturalness of communication [Bazzi 2000]. Often the vocabulary must also be constrained by grammar rules which identify how words can be spoken in context [Peacocke 1990].

2.1.1 Why Use Speech

[Weiser 1996] states that in computing and mobile environments the user's interaction with the distributed services and embedded appliances should be as natural as possible. Intuitiveness, ease of use and seamless access are some of the major desired features. The user should be able to concentrate upon the task to be performed and not be forced to cope with interface issues. With these requirements, in mind, speech based natural language interfaces appear to be a desirable choice. On the other hand, Intuitiveness raises expectations, and if the system cannot cope with natural language, the user is better off to have a non-intuitive system and no preconceptions. Speech is a powerful communications medium that is rich and expressive. Speech is natural, portable, and can be used while doing other things. It is faster to speak than it is to write or type [Gould 1982]. Speech interfaces are especially advantageous in mobile situations and/or when hands and eyes are busy [Cohen 1995]. Among other things, speech input offers speed, high-bandwidth information, and relative ease of use [Oviatt 2000A].

Sawhney and others discussed the suitability of speech interfaces for mobile environments [Sawhney 1998]. They conclude that there are at least three fundamental advantages for using speech.

- "Speech is an ambient medium rather than an attentive one". Visual interfaces require the users focused attention while speech allows us to interact whilst using our other faculties to do something else.
- "Speech is descriptive rather than referential. People describe objects in terms of their roles and attributes". In visual situations people point to or grasp the objects of interest. For this reason, speech is to a large extent complementary, and can often be combined with other mediums to great effect.
- "Speech requires cheap physical resources (i.e. microphone, speaker, telephone or mobile)". Speech-based interactions can be scaled down to much smaller and much cheaper input/output forms than visual or manual modalities. Speech interaction requires only audio I/O devices, which are already quite small and cheap. In addition, the increasing mobile and telephone market allows speech services to be accessed from just about anywhere, anytime and from multiple devices.

Spoken language is the natural instinct of humans for information-rich expression [Burke 2000]. Without a great deal of training, most humans can express themselves in an extensive variety of fields. As an expression medium, speech works for a much wider range of people than typing, drawing or gesture because it is a natural part of human existence.

That said, speech interfaces are only just beginning to make an impact on computer use and information access, for example, computer command, consumer, data entry, speech-to-text, telephone, and voice verification is using on business and personal computing [Oberteuffer 1995]. Such as, Hauser and others developed a command and control speech interface for their medical article records system [Hauser 1999]. The impact thus far being limited to those places where current speech recognition technology, even with its limitations, provides an advantage over existing interfaces. The research project described in this thesis has been designed as a speaker-independent telephone based information access system, because it can provide a high input-bandwidth in an environment where the alternative input mode is DTMF (i.e. Touch-Tone telephone buttons). DTMF based communication applications are becoming increasingly complex to the point where users are becoming frustrated and are looking for a more natural way to get things done [Intel 2004]. The use of speech can simplify the user's interaction with the system so that more complex functionality can be added by service providers. On the other hand, [Rosenfeld 2001] states that speech will only achieve a much higher penetration as an interface technology if certain fundamental limitations are addressed. In particular:

- Highly accurate speech recognition performance
- Excellent usability properties (for the users)
- Ease of development (for the implementers)

2.1.2 Speech Recognition

Speech recognition technology has been available for more than five decades. However, due to technical limitations, few successful speech recognition applications have been developed. Meanwhile, the technological development in mobile computing devices has dramatically increased the market need for speech-enabled interfaces in order to simplify the user interface and free the user's hands and eyes. So while speech recognition technology continues to make only incremental improvements, the potential for speech-enabled interfaces is expanding rapidly. For example to enhance existing mobile interfaces by providing end-users an alternative mode of interaction wherein they can “speak into” their device and have it “type”. Some applications of such interface include: SMS (Short Messaging Service), IM (Instant Messaging), Email, MMS (Multimedia Messaging Service), Wireless-internet-browsing and large vocabulary speech interface for document creation. IBM is developing a speech-to-speech translation system that can translate spontaneous free-form speech in real-time on both laptop and hand-held PDAs (Personal Digital Assistant). Building such a speech system, they have to face these challenges include speech recognition, machine translation in adverse environments, and designing algorithms and building models in a scalable manner to perform well even on memory and CPU deficient hand-held computers [Zhou 2004].

Speech recognition technology faces major problems in terms of recognition accuracy. The accuracy rate of speech recognition is dependent on the following major elements:

- Lexicon and Grammar
- Speaker-dependent and Speaker-independent recognition
- Equipment and Environment

2.1.2.1 Lexicon and Grammar

A lexicon is a pronunciation dictionary that includes information about the phoneme units in each word [Rabiner 1993]. The Automatic Speech Recognition (ASR) engine compares the user’s speech to the phoneme strings stored in the lexicon and makes decisions about what word was spoken. Most large lexicon speech recognisers employ subwords as basic recognition units [Svendsen 1995]. This implies that in order to obtain word (or sentence) recognition, a lexicon that defines the composition rules of the words in terms of basic units must be made available to the recogniser. [Yun 1999] suggests linguistically defined subwords should be used as recognition units; typically phonemes or phone-like units. In general, the lexicon is commonly created by the use of expert knowledge or a standard pronunciation dictionary.

When a speech-enabled interface is developed, the words and phrases that users can speak to the application must also be specified. These words and phrases are presented to the speech recognition engine as a grammar and are used in the recognition process. A grammar [Rabiner 1993] uses a particular syntax, or set of rules, to define the words and phrases that can be recognised by the engine. A grammar can be as simple as a list of words, or can be flexible enough to allow variability in what can be said such that it approaches natural language capability. Generally, smaller grammars are easier for a computer to recognise, while larger grammars are more difficult because of increased ambiguity between words.

The commonly obtained error rates in laboratories on speaker-independent isolated word databases are around 1% for 100 words grammar, 3% for 600 words and 10% for 8000 words [Deroo 1998]. For a speaker-independent continuous speech recognition database, error rates of around 15% for a 65000 word vocabulary have been quoted [Young 1997], although this is under laboratory condition. Recognition grammars are either static (created at design time) or dynamic (dependent on database lookup at run time). [Schalkwyk 2003] presents an efficient technique that addresses dynamic changes of a grammar.

VoiceXML applications use grammars to specify sets of valid user utterances at particular points in an interaction with the application [W3C 2004]. For example, at the beginning of the application, the designer may wish the system to ask the user to select among a set of predefined options. The VoiceXML document uses a grammar to identify the set of possible things a user can say for each option. The speech-recognition engine uses the grammar to identify which option the user is selecting. In addition, the grammar can (optionally) specify how to interpret a valid expression in terms of values for input variables.

Type	Description
Alternation	A set of alternatives ("cat" or "dog" or "goat"). Enclosed in square brackets ([ ]).
Sequence	Multiple expressions that must all be said in a particular order ("great dane"). Enclosed in parentheses (( )).
Repetition	Repeat a single expression some number of times ("very" or "very very" or...). The * and + operators.
Optional	Special case of repeat 0 or 1 times (the "kitty" in match "kitty cat" or "cat"). The ? operator.
Weighting	Likelihood that an expression will be said. Indicated with ~ operator.

Table 2.1 - The Basic Types of Combination in GSL (Grammar Specification Language) Grammar

The Nuance Grammar Specification Language (GSL) for VoiceXML grammars are used in this project. GSL grammars combine tokens and rule references into more complex expressions. See Table 2.1.

The size of grammar of a speech recognition system affects the complexity, processing requirements and the accuracy of the system. There are no established definitions or accepted standard of grammar size [Venkatagiri 2002]. In this research project, the speech-enabled ATTAIN interface for Nottingham is required to recognise only 1355 bus stop names. This application is defined as a medium size grammar application. The London speech-enabled ATTAIN interface will requires a far larger grammar (27792 bus stop names). This application is defined as a large size grammar application. It is clear that the larger the vocabulary the more opportunities there are for the system to make errors and the slower will be the system when processing the user's spoken input. The London speech-enabled ATTAIN interface must, therefore, be optimised to handle the problems caused by using such a large grammar if it is to achieve practical real-time application status.

2.1.2.2 Speaker-Dependent and Speaker-Independent Recognition

Speaker dependence [Fontaine 1996] describes the degree to which a speech recognition system requires knowledge of a speaker's individual voice characteristics to successfully process speech. A dedicated speech recognition engine can "learn" how you speak words and phrases; it can be trained to your voice. Speech recognition systems that require a user to train the system to his/her voice are known as speaker-dependent systems. Most desktop dictation systems are speaker-dependent (i.e. IBM ViaVoice [IBM 2005], Philips Dictation Systems [Philips 2005]). Because they operate using very large vocabularies, dictation systems perform much better when the speaker has spent the time to train the system to his/her voice. Speaker-dependent systems generally are fairly accurate for the trained speaker, but much less accurate for other speakers. They also assume the speaker will speak in a consistent voice and tempo.

Speech recognition systems that do not require a user to train the system are known as speaker-independent systems [Rabiner 1993]. Speaker-independent systems are designed for use by a variety of speakers. Adaptive systems usually start as speaker-independent systems and utilise training techniques to adapt to the speaker to increase their recognition accuracy. Google Labs deployed a speaker-independent system as a demonstration of a telephone interface for its popular search engine [Franz 2002]. However, their system is constrained such that it only gives a reasonable search result for keyword queries from the users. Voice search users who prefer to ask full questions or make other types of natural language queries would prove difficult for their system to model and recognise.

HMM (Hidden Markov Models) or ANN (Artificial Neural Networks) are usually used as the speaker-independent acoustic voice models. Hussien and others developed an Amharic speaker independent continuous speech recognizer based on an HMM and ANN hybrid approach [Hussien 2005]. The model was constructed at a context dependent phone part sub-word level. A promising result of 74.28% word and 39.70% sentence recognition rate was achieved. Carnegie Mellon University and

Sun Microsystems together developed a speaker-independent speech recogniser: SPHINX [Walker 2004]. Sphinx explored variants of HMMs such as discrete HMMs [Lee 1990], semicontinuous HMMs [Huang 1993], and continuous HMMs [Placeway 1996]. SPHINX-4 can recognise continuous speech using a large vocabulary. On a vocabulary of over 21,000 words, SPHINX-4 achieves speaker-independent word recognition accuracies of 71-96%, depending on the complexity of the grammatical structure in the sentences. However, the current development version of Sphinx 4 is still imperfect in many senses, particularly in terms of its speed and its ability to deal with complex sentences.

Speech recognition in this project must be speaker-independent. Hundreds, maybe thousands of users could call the ATTAIN travel information system. It would be impossible to require that each caller train the system to his or her voice. The speech recognition in the ATTAIN interface must successfully process the speech of many different callers without having been trained on the individual voice characteristics of each caller. This fact alone will inherently limit the recognition performance of any developed system.

2.1.2.3 Equipment and Environment

Speech recognition engines need to maintain good recognition accuracy even when the quality of the input speech is degraded, or when the phonetic characteristics of speech are poor. This project will involve speech recognition over the mobile telephone network, and this introduces some unique challenges. First and foremost is the bandwidth of the audio stream. The standard telephone system uses an 8 kHz audio sampling rate. This is a much lower bandwidth than, say, the desktop, which uses a 22 kHz sampling rate. The quality of the audio stream is thus considerably degraded in the telephony environment, making the recognition process much more difficult [Kemble 2001]. Another serious difficulty in the deployment of telephone speech recognition systems for narrowband speech recognition is the expense in both time and cost of obtaining sufficient training data. [Malkin 2004] uses the NYNEX PhoneBook [Pitrelli 1995] database for training and testing their ASR system for low bandwidth speech input. This system is designed for isolated-word recognition tasks and users access this system over a telephone channel with an 8,000 Hz sampling rate. However, this generally results in sub-optimal performance. [Seltzer 2005] propose a new algorithm for training wideband acoustic models using only a small amount of wideband speech augmented by a larger amount of narrowband speech. The algorithm operates by first converting the narrowband features to wideband features through a process called Feature Bandwidth Extension. The bandwidth extended features are then combined with available wideband data to train the acoustic models using a modified version of the conventional forward-backward algorithm. Experiments performed using wideband speech and telephone speech demonstrate that the proposed mixed-bandwidth training algorithm results in significant improvements in recognition accuracy over conventional training strategies when the amount of wideband data is limited.

The telephony environment can also be quite noisy, and the equipment is quite variable. Users may be calling from their homes, their offices, the shopping centre, the airport, or their cars. They may also call from mobile or regular phones. The speech recognition in this interface must be robust enough to cope with all these different environments. Much research has been focused on characterizing and estimating the frequently changing acoustic conditions for speech recognisers, and on identifying and compensating for major sources of recognition performance degradation. In general, the performance of existing speech recognition systems, whose designs are predicated on relatively noise free conditions, degrades rapidly in the presence of adverse conditions. It was found that recognition accuracy for a typical speech recogniser drops from 96% for clean speech to 73% as the signal-to-noise ratio (SNR) is decreased to 20 dB, and to 31% at 10 dB SNR [Junqua 1996]. However, a recogniser can provide good performance, even in very noisy background conditions, if the training material from which the reference patterns of the vocabulary are obtained is carefully chosen so that its background conditions match that of the best condition. Aurora [Hirsch 2000] provides an excellent platform in which to research noise robustness techniques and the preliminary results are encouraging. The average performance of recognition accuracy between 0 and 20dB takes a value of 87.81% for a typical speech recogniser.

2.1.3 Speech Synthesis

Speech synthesis [Dutoit 1997] is a basic requirement of all interactive speech-enabled interface systems. Like speech recognition, speech synthesis is a critical component of a speech-enabled interface system. A Text-To-Speech (TTS) synthesiser is a computer-based system that should be able to read any text aloud. In a manner similar to human reading, a TTS synthesiser comprises a Natural Language Processing module (NLP) that is capable of producing a phonetic transcription of the text with the desired intonation and rhythm. A Digital Signal Processing module (DSP), then transforms this symbolic information into speech.

The first full text-to-speech system for English was developed in 1968 by Umeda and Teranishi [Umeda 1975]. The speech was very intelligible but humdrum and far away from the quality of present systems. High quality TTS synthesis was developed in the mid eighties, as a result of important developments in speech synthesis and natural language processing techniques. For example: Telesensory Systems Inc's commercial TTS system [Klatt 1987], DECtalk and Infovox SA-101 [Allen 1987]. Such systems make it possible to access textual information over the telephone and AT&T has organised a series of consumer tests for promising TTS based telephone services [Levinson 1993]. It found that TTS could speed up queries in telecommunications services. For this, TTS has to deliver highly intelligible output for general text, whilst also sounding natural and intelligible. A TTS comparison, conducted by ESCA/COCOSDA [Beutnagel 1999], tested a total of 17 English language systems (13 US and 4 UK English systems representing female and male voices; among these systems were Microsoft's Whistler, British Telecom's Laureate, Lucent Bell Labs' TTS, and AT&T's Next-Gen TTS). These systems were compared in two categories: naturalness and intelligibility. The result showed that the intelligibility problem in TTS might be close to being solved while naturalness still has plenty of room for improvement. As cited by Schroeter, TTS quality is characterized by two factors; namely the intelligibility of the speech that is produced, and the naturalness of the overall message that is spoken [Schroeter 2001]. The major evaluation criteria of TTS quality are intelligibility and naturalness of speech [Kamm1997]. The naturalness of speech is a subjective factor which is user perceived speech naturalness comparing to human speech. To let TTS reach naturalness, some techniques still need to be improved [Dutoit 2003]:

- Formalizing the relationship between syntax, semantics, pragmatics and prosody
- Deriving natural sounding intonation and duration from abstract prosodic patterns
- Accounting for speaker and speaking style effects

One way to overcome the limitations of speech synthesis systems is to incorporate real audio output. VoiceXML facilitates this by allowing references to be made to stored audio files (e.g. wave files) that are linked inline. The disadvantages of this method, is that all possible output message need to be known when the system is designed. This tends to limit such messages to menu and numerical value style systems [Inria 2004]. In this research project, the speech-enabled query interface for the London travel information system contains 27792 bus stop names. It would be a very large amount of work to record prompts containing all of these bus stop names. Only using recorded audio was therefore impracticable for giving prompts and conversational feedback to the user in the ATTAIN system. Mixing TTS and recorded audio was one possible solution for resolving this problem, e.g. Do you want to go from (audio) *Oxford Street to Royal Arcade* (TTS)? However, because sound changes from a real human voice to TTS in one sentence may affect the usability of the developed interface it was decided that only TTS prompts would be used in this work. Chapter 3 and Chapter 4 details how TTS is used in this research work.

2.2 Speech Dialogue Management

As described by [Turunen 2001A], speech-enabled communication can differ greatly between individual users and situations. For example, some people prefer that the computer takes the initiative, whilst others may prefer a more user-initiated style of communication. Speech is also very language and culture dependent and differences between user groups (e.g. age, gender) can be large [Darves 2002]. Natural interaction therefore requires flexible interaction models that need to be supported by the system architecture [Allen 2001]. In order to construct an efficient speech-enabled interface the interaction method must be able to adapt to the different users and situations and facilitate the dialogue between a user and the speech-enabled information system. Consequently, the interaction should not necessarily be based on sequential control. It must be handled in a more flexible way. All these needs imply some kind of coordination, selection and evaluation of different possibilities. The interface should have the ability to calculate predictions for how to continue a dialogue using either knowledge sources (e.g. dialogue grammar and history) or different kinds of dialogue (e.g. directed-dialogue or mixed-initiative dialogue). In most speech-enabled interfaces, dialogues are structured in a pipeline fashion (directed-dialogue); that is, they are executed in a sequential order. This kind of dialogue is considered suboptimal for interactive systems [Garian 1993]. In order to facilitate the development of advanced speech-enabled interfaces (i.e. mixed-initiative), more advanced techniques, models, methodology, and tools are needed.

[Pakucs 2001] suggests a user-centred, application independent model for speech dialogue management, where every user is expected to use a single, highly personalised speech interface to access speech appliances. The user-centred dialogue management method is one way of providing widely available speech services. This solution relies on and supports personalisation, adaptability, context awareness and user modelling. However, to do this it is necessary to gather data on the user's behaviour and speech patterns, in order to build the user models. These can then be used in dialogue management to predict the user intent given the current dialogue context. Through adapting the speech recognition to individual users, the speech recognition error-rate can be decreased substantially. The ATTAIN is a travel information system designed for general public usage. This research project has to make the service available to all users, in spite of dialects, non-native accents and even speaking disorders. Building user models for all these user groups is impracticable. Consequently user models are not used in this work.

Earlier work in speech-enabled system architectures used client-server approaches where users can communicate with the system from light-weight clients while specialized servers handle the computationally heavy tasks such as speech recognition, language understanding, database access and speech synthesis [Seneff 1998]. The best known speech-specific client-server architecture is Galaxy-II [Seneff 1999], which was developed by MIT (Massachusetts Institute of Technology) and has been used successfully in several applications such as the Jupiter, Voyager and Orion systems. The Galaxy's architecture is a general agent architecture that has been used in the construction of many speech applications. This kind of architecture offers the necessary infrastructure components for speech applications, but it does not support adaptation in any particular manner [Martin 1999]. There are three particularly interesting recent examples using the agenda-based dialogue management architecture theory [Rudnicky 1999]. These are: RavenClaw extension [Bohus 2003], Queen's Communicator [O'Neill 2003], and SesaME [Pakucs 2003]. The purpose of these approaches is not to provide a complete speech architecture but instead, a model for dialogue management. IBM's Jaspis [Turunen 2005] introduces a new paradigm for interactive systems that focuses on speech-based applications. It is possible to construct highly adaptive systems suitable for different user groups and to support accessibility by using the jaspis architecture and its principles.

Any automated speech-enabled system will inevitably be compared by the users to a human operator offering a similar service. By analysing human-human and (simulated) human-computer dialogues, [Giachin 1996] found that while users expect reduced linguistic competence from a system (and reflect this in a simplification of their own linguistic behaviour), they still expect the system to retain many of the characteristics of a human operator, whilst compensating for the systems performance limitations. Users expect the system to repair failures which may arise from performance limitations, especially speech recognition, as well as limitations in system knowledge - linguistic knowledge, semantic knowledge, task knowledge and dialogue knowledge. In short, the system needs to have necessary strategies to avoid dialogue failure. Error recovery stategies are therefore essential in any human-computer interface; particularly one that use an error prone recognition technology such as speech recognition.

2.3 Error Detection and Correction

Most errors in speech are a result of the system's reliance on a statistical interpretation of the user's utterances [Berglund 2003]. Speech recognition systems prioritize hypotheses based on language models derived from statistical techniques such as n-grams and Hidden Markov Models [Lieberman 2005]. Speech recognition systems produce a list of weighted candidate hypotheses for a given audio segment, and choose the "best" candidate to distinguish between words and phrases that sound similar but have different meanings. If the choice is incorrect, how a system handles recognition errors can dramatically affect the performance of a speech-enabled interface. If either the system or the user detects an error, an effective speech-enabled interface should provide one or more mechanisms for recovering from the error. Error management has usually been separated into error detection and error correction [Turunen 2001B]. Research in the field of linguistics has investigated strategies people use in dealing with communication problems in conversations [Schegloff 1977], [Turunen 2001B] and [Skantze 2003]. The three main strategies used in human to human conversation are avoiding communication problems, initiation of a repair dialogue as soon as a communication problem has been detected and collaborative work on the repair [Clark 1989]. Applied to speech user interface design, these strategies correspond to the following two approaches for dealing with interpretation errors:

- ❖ Reduce the number of interpretation errors by training or guiding the user towards speaking styles and formats which the automatic interpretation system can interpret more accurately. See 2.3.1
- ❖ Detect the interpretation errors through context sensitive feedback messages and recover from these interpretation errors by involving the user in interactive error recovery dialogues. See 2.3.2.

The major speech recognition errors are rejections and substitutions [Peissner 2001]. The system can detect rejection errors using confidence measures based on comparing the recognised word's 'score' to some predefined threshold. The most common use of confidence scores in error handing in speech-enabled systems is to choose between rejection (low confidence, rejection errors), levels of confirmation (medium confidence, substitution errors) and acceptance (high confidence) [Larsson 2002]. Skantze suggests that the threshold used should be tuned according to empirical data [Skantze 2003]. There is a trade-off that has to be made between the number of correct rejections and correct acceptances. If the threshold is too high, it will increase the number of false rejections. Conversely, if the threshold is too low, it will increase the number of false acceptances. The tuning should aim to finding the lowest number of false acceptances and false rejections. This break-even point is often close to the so-called Equal Error Rate, where the number of false acceptances equals the number of false rejections [Skantze 2003].

2.3.1 Rejection Errors

When handling rejection errors [Martin 1980] developed a method based on repetition. The speech recogniser stores the prior interpretation results of the user input in a buffer. Then, the buffer can be actively and interactively edited by the users when repeating their spoken input. The idea of just letting the user repeat the erroneous output has also been used, with some modifications such as eliminating elements from the buffer that are known to be incorrect [Ainsworth 1992 and Murray 1993]. The drawback of repetition is that users have to repeat their input over and over if the speech recogniser's interpretation is wrong. [Frankish 1992] has shown that repetitions tend to have lower recognition accuracy. This is because, if the user is asked to repeat a word or phrase again, the same error is likely to be repeated. In addition to this, users tend to adjust their way of speaking to what they believe is easier for the recogniser to interpret, which often has the opposite effect. Although [Danis 1989] showed that it is possible to increase speech recognition accuracy by training the user (i.e. trying to eliminate the speaking behaviours which tend to cause recognition error) this method is a trade off between accuracy and naturalness. However, [Suhm 1999] argues that if accuracy is significantly improved, an error recovery strategy based on repetition can outperform other strategies, although it is not advisable to simply repeat the same error recovery procedure; otherwise the user may experience more than one rejection error in a row. Instead, progressive assistant prompt messages and/or alternate input modalities should be used to recover from the recognition error. For example, the user could be prompted to press a key on the telephone pad as an alternative to speaking. All that said, none of the existing studies have considered a mobile-setting where the input methods are limited to speech and DTMF keypad input.

2.3.2 Substitution Errors

The techniques for handling substitution errors are somewhat different to those for handling rejection. Substitutions errors are harder to detect, thus harder to handle. One possible strategy is to apply natural language based processing technique to a list of the most likely utterances for a given user input [Bernhard 1996]. For example, with the flight booking interface developed by MIT [Filisko 2004], the system could assume that an error has occurred if the user appears to want to travel from Boston to Boston. Although the word "Boston" is given the highest confidence, the system could choose a lower confidence word that is more probable. Another way would be to include flexible correction mechanisms that allow a user to correct a portion of the input. For example, if the user asks for the flight information from Boston to New York, the system might respond with "ok, you want to travel from Boston to New York…" A flexible correction mechanism would allow the user to just correct the destination by saying: “No, I said to New Jersey.” Another method based on choosing an item from a list of alternative words was introduced by [Murray 1993]. The list contains all the alternative interpretations the speech recogniser makes from the user utterances; this list is also called an n-best list. The disadvantage of the n-best list strategy is that it is a trade off between speech recognition accuracy and naturalness. When the system asks the users to choose their input from the n-best list presented in form of a spoken menu, it is an unnatural interaction from the human user’s point of view that also tends to increase the cognitive load placed on the listener by the interface.

A major problem with speech recognition in dialogue systems is that the speech recogniser only can recognise words that are in the predefined vocabulary. If the user uses a word that is out of vocabulary (OOV), the recogniser will still make a “guess”, using the words in vocabulary. This may result in a substitution error. [Hazen 2001] suggests that one solution is to use “filler” or “garbage” models in the speech recognition that should catch OOV words. Another option is to use the confidence scores to identify those recognised words that represent OOV words. These two methods were compared by [Hazen 2001]. The comparison showed that the OOV model performed better than the confidence scoring model. However, the best result was gained by combining the two methods.

2.3.3 Error Recovery Methodologies

Given a collection of error recovery methods it is often difficult for the designer to predict which interaction method the users will prefer. Applying the principle of least collaborative effort which governs error recovery in human to human conversations, [Clark 1986] found that users prefer methods which minimise the effort necessary to recover from any errors. [Suhm 1996] states that the effort required is determined by the following three dimensions:

- ❖ Time required by the user to provide the input and by the system to process it
- ❖ Accuracy of the system in interpreting the input

- ❖ "Naturalness" of the interactions

After surveying existing commercial and research systems that utilise speech recognition, [Mankoff 2000] found that two error resolution strategies are the most commonly used. Repetition is usually done in the same modality as the original input. Offering a choice of best alternative is usually done by selecting one of the alternatives from the n-best list presented for the user in the form of a menu. This selection could be done using a different modality to that used in the original input, i.e. selecting an item by pressing the keypad rather than re-speaking. It has also been shown that combining recognition results from multiple input modalities can be effective in reducing the occurrence of recognition error [McGee 1998 and Oviatt 1999A]. Furthermore, [Suhm 2001] studied four multimodal error resolution methods for a dictation task. The results showed that multimodal error correction was more efficient than unimodal correction; i.e. error correction was more efficient if the users changed their modalities. For example, using speech for input and keypad input for correction was more efficient than using speech for both input and correction. In addition to these two strategies, they found that the use of confidence levels (a measure of the recogniser's interpretation confidence) to decide which interpretation to use provided an efficient error recovery mechanism. In this case the user was not involved in choosing the error resolution strategy.

In another survey of commercial and research spoken dialogue systems, [Walker 2000] found that the repetition and n-best strategies or not effective for identifying and repairing problems that arise in the conversation. In this situation, preventing the problems arising is an efficient way for dialogue management. [Walker 2002] trains a problematic dialogue classifier using automatically obtainable features that can identify problematic dialogues significantly better (23%) than the baseline (67.1%). The problematic dialogue classifier learns to automatically identify and predict problematic human-computer dialogues in a corpus of 4774 dialogues collected with the How May I Help You (HMIHY) spoken dialogue system [Gorin 1999]. The HMIHY system was installed at an AT&T customer care centre. HMIHY answered calls from live customer traffic and successfully automated a large number of customer requests. The classifier has the ability to predict problematic dialogues which allowed the system's dialogue manager to modify its behaviour to repair problems, and even perhaps, to prevent them. In this research project, only repetition and the n-best list error resolution strategies are used. Compared with the other strategies, these two strategies require the least amount of time and cognitive effort on the part of the user when interacting with the system and on the part of the system when processing the user inputs. From a usability point view, asking people to repeat what they said is a natural form of interaction [Raux 2005]. Chapter 3 and Chapter 4 details how these two error recovery mechanism are used in the system.

2.4 Speech Enabled Interfaces

Speech recognition technology is becoming popular for a variety of applications. In navigation or command and control applications, speech recognition is used to operate appliances and machinery, including computers. The hands-free dialling of a mobile phone is a common example of a command and control speech recognition application. ScanSoft embeds a voice card into the mobile phone to provide English, German and French speech-enabled operation [ScanSoft 2004]. Advanced voice capabilities, available within the voice card, make the mobile easier, more intuitive and faster to use, and provide hands-free interaction with the device so enabling motorists to access phone numbers using only their voice [Turunen 2005]. This speech recognition application does not need a complex speech interface design, because the mobile users only access phone numbers by simply speaking the phone number or name stored in the mobiles phone book memory. Voice commands are also used in video game playing [Mindmaker 2005 and VoconGames 2005]. These systems simplify the gaming process by allowing any gamer to control the game using their voice rather than memorizing hundreds of keyboard commands. However, such systems still tend to restrict heavily the user to a group of valid commands and simple sentences.

Speech recognition technology is being tentatively used, and researched, in the car industry [White 2004]. The major development concentrates on speech instructions to navigation systems, e.g. GPS (Global Positioning Systems) connected digital maps: "How far is it to London?" and climate controls, radio/CD (Compact Disc) controls, e.g. "Please turn to BBC 1". Although there are advantages for speech recognition in cars, such as being able to keep both hands on the steering wheel while telling the "car" to do something, there are considerable obstacles in terms of in-car noise and passenger interruption. The Project54 system, developed by the University of New Hampshire, integrates electronic devices in police cruisers [Kun 2004]. This integration allows officers to have control over all of the electronic devices using speech. Project54 has a quoted speech recognition accuracy of 86%. Unfortunately, this system can only be used by trained officers.

The existing public telephony infrastructure provides users with ubiquitous access to information and communication services. Several telephone-based systems provide advanced speech and audio interfaces to such services. Phoneshell allows users to access email, voice mail, calendar, and a variety of news, weather and traffic reports using a touch tone telephone [Schmandt 1993]. Users hear messages feedback via synthetic speech and audio playback. Phoneshell is constrained by the limitations of the telephone keypad and it does not offer speech interaction; however this system did introduce a new form of terminal in HCI research; namely the phone. [Ly 1994] attempted to extend the functionality of Phoneshell using continuous, speaker-independent speech recognition. This system focuses on communication within a workgroup, i.e. reading email, sending voice messages to workgroup members etc. The major contribution of this system is the incorporation of memory-based reasoning techniques. These make predictions about the user's next action based on an analysis of the user's prior history. By reducing much of the interaction to "yes" and "no" responses, recognition errors were reduced. That said, since this early system had a relatively poor speech recognition accuracy, many mistakes were observed during use, e.g. deleting messages by mistake or unpredictably hanging up on the user during a session.

Huang developed an integrated computer and telephone-accessed WWW (World Wide Web) system (CTW) to provide a ubiquitous web access service [Huang 2000]. The CTW system serves as an intermediary between the telephone user and web sites by providing an alternative mechanism for people accessing a WWW form-based service via a telephone set. A web page designer employing HPML (Hyper Phone Markup Language) with the CGI (Common Gateway Interface) extension and the symbolic pronouncement extension has the flexibility to compose comprehensive and dialog-based WWW services for telephone users. HPML allows a set of web pages embedded in a CGI form to be accessible to both computer and telephone users. The CTW system provides a ubiquitous web access service for users at any degree of web-literacy by integrating well-installed telephone networks with the Internet. The disadvantage of this system is that the user can only give single word commands (rather than full sentences) with most of the user's input being in the form of DTMF keypad inputs.

Another example of a speech-enabled telephone based system is the flight arrival and departure information service introduced by MIT [Seneff 2002]. This system (Mercury) is a speaker-independent continuous system which uses a very sophisticate dialogue model to make the system intuitive to use and robust in handling the wide range of different ways in which users can express their flight constraints and select the flights of the itinerary. They have showed that it is possible to design a telephone access spoken dialogue system that people would be willing to use to plan their air travel. The disadvantage of this system is that Mercury can only recognise a vocabulary of approximately 1,000 words.

In this research project, the speech-enabled interface for the ATTAIN travel information has to deal with a potentially very large vocabulary. This raised many problems in terms of recognition accuracy and system response time and Chapter 5 details the novel grammar segmentation approach taken to resolve these difficulties.

2.5 Multimodal Interface

Yuen states that a multimodal system supports interaction with the user through the use of more than one modality [Yuen 2002]. The term modality refers to a form of sensory perception: hearing, vision, touch, taste and smell. [ManÉ 1996] defines modality as a communication channel between the user and the mobile. The modes above can be combined in a multimodal interface, containing audio (e.g. in the form of speech), vision (in the form of text and graphics, or moving video), and touch. [ManÉ 1996] defined three types of multimodal interfaces:

- ❖ Complementary - Different modalities are used to accomplish different tasks. The system designer decides on the most appropriate modality for each task.
- ❖ Concurrent - Different modalities (such as speech and touch tone, or speech and mouse) can be used interchangeably to accomplish the same task.
- ❖ Separate - The same application can be driven with speech or with some other modality (such as a GUI), but not at the same time.

A major design challenge is to select the appropriate modality for each application. This involves matching the strengths of a particular modality to the task. For example, speech is probably not a good choice if the user's task involves interacting with a computer while talking to another person. In many applications, however, the expressive power of language can be a valuable complement to other modalities. Another challenge is to set users' expectations appropriately. A system that uses speech output and sounds very naturally may be misleading to the user if the system can only accept limited speech input. It is not in a person's common experience to encounter a person who can speak fluently, but not understand very well. Non-native speakers, for example, typically have the opposite problem. They can often understand the language better than they can produce it. Oviatt examined how people might combine input from different devices in a multimodal computer interface [Oviatt 1994]. They used a simulated service transaction system with verbal, temporal, and computational input tasks using both structured and unstructured interactions. Participants were free to use handwriting, speech, or both during testing. [Oviatt 1994] evaluated user preferences in modality integration using spoken and written input. Among the findings, it was noted that simultaneous input with both pen and voice was rare and that digits, proper names, and structured interactions were more likely to be written.

Accessibility is mentioned as one of the major motivations for the development of multimodal interface [Turunen 2005]. For example, there has been much work in speech-enabled interfaces to allow visually impaired users to access existing graphical interfaces [Mynatt 1994]. Multiple modalities have also been used to make human-computer interaction accessible for people with disabilities [Edwards 2002]. In the ideal case, a speech-enabled interface should take the needs of different users and usage conditions into account and interaction should be adapted to each user/usage situation. Ito developed a multimodal interface that allows use of voice and pointing gesture by facing as commands for controlling distributed home appliances used by people with disabilities [Ito 2001]. However, current speech-enabled interfaces development architectures tend to lack the flexibility necessary to adapt to a variety of users and usage conditions.

In this research project, the existing ATTAIN system is based on a text message interaction only. There are many people whose disabilities prevent them from making full use of a text message only interface - i.e. people who can not use their hands, or who have difficulty using their hands, or who can not see the display well enough to use a text message only interface. The decreasing size of mobile devices is also making keypads and screens smaller and more difficult to manipulate, consequently designers need to consider alternative communication channels when interacting with these terminals. Speech is a natural and convenient way to express complex questions to a service [Oviatt 1991] [Cohen 1995] [W3C 2000A]. However, when the speech recogniser fails, the user may want to use other more familiar input methods. Input methods such as touch keypad are therefore also required. The terminals' output channels should also support visual (text and graphics) presentation and audio. [Gould 1982] states that it is faster to speak than it is to write or type. However, [Gollner 1994] also asserts that it is slower to listen than it is to read. Together therefore these two findings suggest that speech is efficient for the talker, but hearing speech is usually a burden on the listener. Listening is a relatively slow way of obtaining information. The most significant disadvantage of speech is that it is temporary. Once speech has been uttered, the auditory information is no longer available. This can place extra memory burdens on the user and severely limit the ability to scan, review, and cross-reference information. It is often difficult to remember anything more than a few bus stop names or numbers. Screens are needed to display graphics and information that it is tedious to listen to. The multimodal version of the ATTAIN speech-enabled system developed as part of this project allows the use of speech to input the journey information required but uses text messages to present the results of the search back to the user. This multimodal method thus helps overcome the human short-term memory problem present in other speech only interfaces.

2.6 Usability Issues

Optimised speech-enabled interfaces require a systematic approach to the design process. However, to ensure optimum performance, usability testing of the developed system is also required. [Nielsen 1994] views usability as a complement to utility. Utility refers to whether a system is able to perform what it is supposed to do and usability refers to how well the users are able to use the system. Unfortunately, such a usage of usability tends to lead to systems that are built only using functions that can be easily used by the user. Although the system is able to carry out the task it has been designed for, it disregards the "context of use" (e.g. users needs) of a system. As argued by Trauboth, the usage of usability in such a way will only create systems that are usable but not necessarily useful [Trauboth 1996].

2.6.1 Usability Definition

The word usability is often defined by different people in different ways. [Nielsen 1994] states that usability relates to psychology, human factors and ergonomics. In this defination, usability is different from the rest of design which focuses on the human issues. Usability is most often interpreted by software engineers as relating to skills in interface design which complement other design objectives such as functionality, efficiency and reliability. [Bevan 1995] states this is a narrow product-oriented view of usability which suggests that usability can be designed into a product. In this sense, usability is closely related to ease of use, which is probably the most common way the term is misused. What really counts is whether a user can achieve their intended goal when they use the product. The answer depends not only on usability as ease of use, but also utility, reliability and computer efficiency. In designing to enable a user to achieve their goals one needs to make a trade-off between these properties.

Usability was also defined in this broad sense, long ago, by [Whiteside 1988]. This definition has the advantages that:

- It is a business-oriented view which focuses on the real objectives of design;
- It is relatively easy to measure.

The definition of usability used in ISO 9241-11 and the MUSiC project [Bevan 1994] is: The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

Effectiveness refers to whether the users are able to carry out the intended tasks (e.g. accuracy). Efficiency refers to the effort needed to carry out the required tasks (e.g. time to carry out the tasks). Meanwhile satisfaction refers to whether the users feel comfortable with the system or whether they prefer a particular system over another.

Note that the ISO's definition is different from Nielsen's definition on usability. ISO's usability definition measures the quality between the system, user, tasks and the environment it is being used in, whereas Nielsen's usability definition concentrates on whether the functions available in a system are usable by the user. Through the ISO's definition of usability it is possible to develop a system that allows the user to complete their tasks effectively and efficiently. User satisfaction is achieved to the extent that the users are satisfied once they have completed their tasks.

This broad definition of usability turns out to be synonymous with "quality of use" [Bevan 1995], i.e. the higher level quality objective that the product not only meets its specification, but also works well in the real world! In software engineering, the conventional objective for quality is to build a software product which meets the specification. However, this alone is rarely sufficient to ensure quality of use that the product can be used for its intended purpose in the real world.

In this project, the speech-enabled interfaces will employ a range of conversational modes from simple directed-dialogue to complex mixed-initiative dialogue systems. The ATTAIN project will use the ISO's definition on usability as the main basis for carrying out usability studies on these two approaches. The usability metrics in the ISO's definition of usability (effectiveness, efficiency and user satisfaction) will be included.

2.6.2 Usability Measures

While carrying out usability tests on the ATTAIN system, it is important to measure each usability aspect individually. However, deriving conclusions based on one of them to reflect on total usability may prove to be incorrect. It is very common to assume that if a system is able to achieve a high accuracy and allows the users to complete the task at hand quickly, then user satisfaction/preference will also be high. However, the observation of Walker has shown that there is a weak correlation between effectiveness, efficiency and user satisfaction [Walker 1998]. In Walker's experiment where he compared a system initiated and mixed initiated speech-enabled system, it was found that despite the fact that the mixed initiated system achieved a higher performance in terms of effectiveness and efficiency for the expert users, it did not lead to an increase in user preference. This illustrates the importance of measuring effectiveness, efficiency and satisfaction together. Hence it is of utmost importance that the usability of the proposed interface is measured in all of these three aspects and that the correlation between these aspects be investigated.

2.6.2.1 Choice of Measures

Because the relative importance of the components of usability depends on the context of use and the purposes for which usability is being described, there is no general rule for how measures should be chosen or combined. The choice of measures and the level of detail of each measure are dependent on the objectives of the parties involved in the measurement. The relative importance of each measure to the goals should be considered. For example where usage is infrequent, high importance may be given to measures of learning and re-learning. If it is not possible to obtain objective measures of effectiveness and efficiency, subjective measures based on the user's perception can provide an indication of effectiveness and efficiency.

2.6.2.2 Effectiveness Measures

[ISO 1998] states that effectiveness measures relate the goals or subgoals of the user to the accuracy and completeness with which these goals can be achieved. For example if the desired goal is to reproduce accurately a two-page document in a specified format, then accuracy could be specified or measured by the number of spelling mistakes and the number of deviations from the specified format. Completeness could be measured by the number of words of the document transcribed divided by the number of words in the source document.

2.6.2.3 Efficiency Measures

[Bevan 1997] states that measures of efficiency relate to the level of effectiveness achieved by the expenditure of resources. Relevant resources can include mental or physical effort, time, materials or financial cost. For example, human efficiency could be measured as effectiveness divided by human effort, temporal efficiency as effectiveness divided by time, or economic efficiency as effectiveness divided by cost.

If the desired goal is to print copies of a report, then efficiency could be specified or measured by the number of usable copies of the report printed, divided by the resources spent on the task such as labour hours, process expense and materials consumed.

2.6.2.4 Satisfaction Measures

[Wachowic 2003] states that satisfaction measures give an indication of how comfortably users are when they use the system, and their attitudes towards the use of the product.

Satisfaction can be specified and measured by subjective rating on scales such as comfort experienced, liking for the product, satisfaction with product use, as well as the acceptability of the workload when carrying out different tasks or the extent to which particular usability objectives (such as efficiency or learnability) have been met [ISO 1998]. Other measures of satisfaction might include the number of positive and negative comments recorded during use. Additional information can be obtained from longer-term measures such as rate of absenteeism, observation of overloading or underloading of the user's cognitive or physical workload, as well as health problem reports or the frequency with which users request transfer to another job.

2.7 Summary

This Chapter has outlined the established research issues associated with designing, implementing and testing an error tolerant, narrowband, speaker-independent, countinuous speech recognition interface. The next three chapters detail the results of applying this knowledge to produce directed-dialogue and mixed-intitiative speech-enabled interfaces for medium and large sized vocabulary systems.

CHAPTER 3
A MEDIUM VOCABULARY DIRECTED-DIALOGUE INTERFACE

This chapter discusses ways in which speech can be utilised to overcome the identified limitations of the current text-based ATTAIN system. Speech-enabled interfaces can employ a range of conversational modes from simple directed-dialogue to complex mixed-initiative dialogue systems but, in this initial phase of the research project, it was decided that a directed-dialogue speech-enabled query interface should be developed because this would produce an interface capable of providing the best speech recognition accuracy whilst at the same time providing a baseline usability performance. Issues surrounding the design of the speech-enabled interface and how the dialogue management was organised will be discussed.

3.1 The System Architecture

The VoiceXML based speech-enabled interface was developed which can handle 8 users simultaneously interacting with system. The users connect, using either land based or wireless telephones, to a VoiceXML Gateway through an ISDN (Integrated Services Digital Networks) line. Currently, the Motorola's Voice Developer Gateway is used, which provides the power to create and deliver the voice application in a single, self-contained unit. The telephone service provider NTL provides the 30 line ISDN link. Figure 3.1 shows the architecture of the prototype speech-enabled ATTAIN system interface.

A Dialogic Telephony Interface (DTI) Card connects the Gateway to the ISDN. This connection or port is the channel by which an outside caller reaches the Gateway. The DTI card uses the ISDN PRI (Primary Rate Interface) signalling protocol and currently supports 8 of the possible 30 ports. A Digital Signal Processor (DSP) card is used in the Gateway for Automatic Speech Recognition (ASR) and text-to-speech (TTS) capabilities. The DSP offers a powerful signal processing foundation on which to deploy the signal-processing algorithms. The ASR and TTS software operate in conjunction with the DSP card.

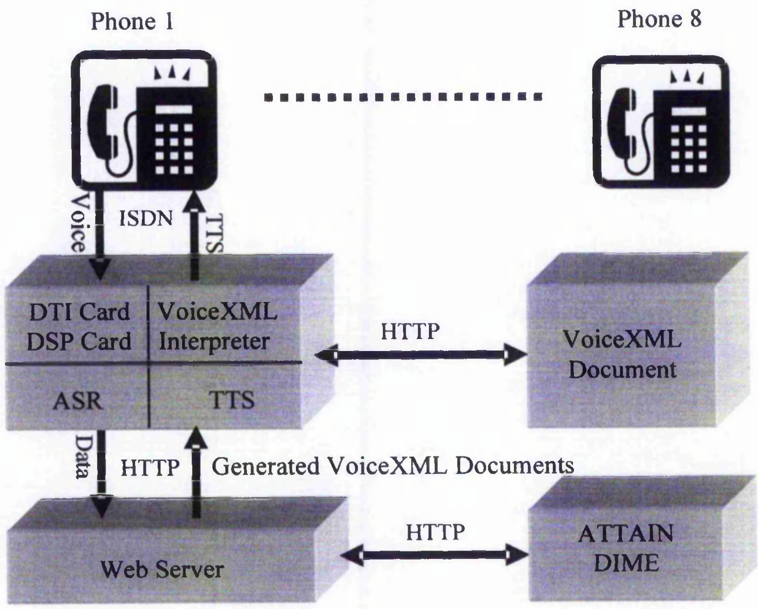


Figure 3.1 - The ATTAIN Speech Interface Architecture

The VoiceXML browser software in the Gateway allows a user to navigate through a voice driven application via voice menus or commands. The software responds to the spoken commands from the user by presenting system information back to the user in an audible format. This is similar in operation to a PC environment, where a graphical Web browser collects and presents Web information to the user. The VoiceXML browser interacts with the user via the telephone network and a Web server.

A VoiceXML application is different to HTML applications. An HTML application is intended for a standard Web browser whereas a VoiceXML application is intended for a user's telephone. The standard Web browser runs locally on a user' machine, whereas the VoiceXML interpreter runs remotely on a VoiceXML hosting site. On every call to a VoiceXML application, all of the resources (grammar files) needed by that application may need to be fetched from a location other than where the VoiceXML interpreter runs. These resources may then be stored locally at the hosting site (i.e. cached) by the VoiceXML interpreter for later use by the same or a different application on another call.

The system can pre-fetch the grammar files into one proxy catch which is the same as a Proxy Server. For example, once the first user has called the system, if someone else then calls the system, the system can use the same grammar file for all subsequent users. This grammar may be stored in the proxy cache and be available to these subsequent users from that cache instead of going out over the Web.

The VoiceXML document application is delivered to the Voice Gateway from a VoiceXML document server. Hosting VoiceXML documents on an external web server keeps the Gateway isolated from any other software packages that may impact its performance. Any Internet web server that can be accessed by the Gateway is appropriate for this purpose. Making the relationship between the web server and the Gateway as close as possible will minimise latency.

The system also arranges (via the voice gateway) to have a Servlet interpreter of the VoiceXML language used in the Voice Gateway and the ATTAIN system Web server. This is to convert the VoiceXML query language into the query format required by the existing ATTAIN system text based query format. After the VoiceXML Gateway has accepted the voice input and recognised the words, it sends the user's query to the Servlet. This Servlet may use a variety of different mechanisms CGI, ASP, JSP (JavaServer Page), Miva Script etc, but in this system it uses Java. The Servlet communicates with DIME (Distributed Memory Environment) [Kosonen 1999], which contains the real bus timetable information, to get the required travel information. It then generates a new VoiceXML document to pass this information back to the user in audio form.

The system uses Nuance V7.0.4 for Speech Recognition. The Nuance System is based on an open client/server architecture that is reliable, robust, and scalable from small to very large grammars. The Nuance ASR maximizes flexibility and efficiency by taking advantage of a distributed architecture, which features the following components:

- The RecClient process is responsible for speech acquisition (including preprocessing of the audio input), the application interface, and optional functions such as telephony control or audio prompt playback.
- The RecServer process is responsible for speech recognition, speaker verification, and natural language understanding.
- The Resource Manager manages the efficient distribution of recognition tasks to a pool of RecServer processes.

Figure 3.2 shows the Nuance System's distributed architecture.

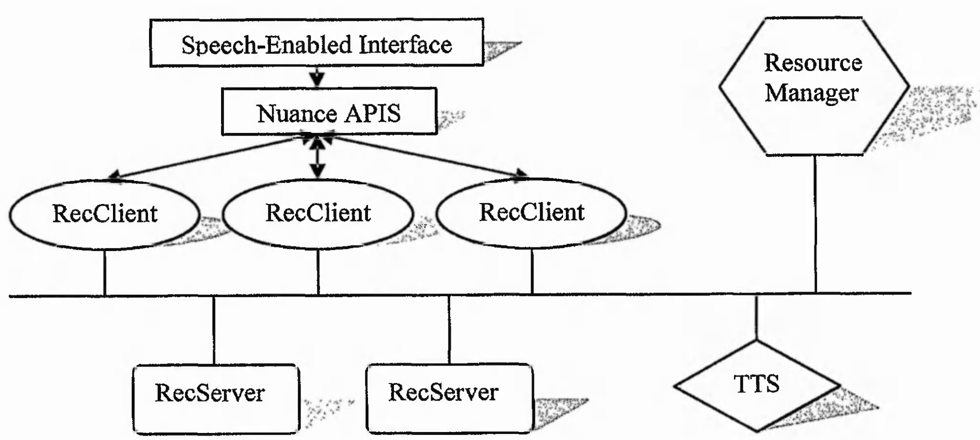


Figure 3.2 - The Nuance System Distributed Architecture

The speech recogniser works through the following steps (See Figure 3.2):

1) A call arrives at the RecClient, which notifies the Gateway. The Gateway detects and picks up the call.

2) The speech-enabled interface asks the RecClient to play the first prompt, to which the caller responds. For TTS prompts, the RecClient sends the text to be synthesised over a socket to the TTS server and receives samples back.
3) To perform recognition on the caller response, the RecClient sends a server request to the Resource Manager (while buffering the audio data) and the Resource Manager points the RecClient to the most appropriate RecServer.
4) The RecClient sends a recognition request to the RecServer. Each request consists of the audio stream and a reference to the grammar to use. The reference to the grammar implicitly includes the acoustic model, because both are built into the recognition package loaded on the RecServer.
5) The RecServer receives the request, performs the recognition tasks, and returns the recognition results to the RecClient.
6) The RecClient sends the recognition results to the interface.
7) The interface responds appropriately, for example, by making a DIME lookup or by requesting that the RecClient play another prompt as response to the user.
8) The caller responds and the RecClient sends out the next request for recognition.

3.2 Dialogue Design

Creating a robust speech-enabled interface is important. As defined by Stern, robustness in speech recognition refers to the need to maintain good recognition accuracy even when the quality of the input speech is degraded, or when the acoustical, articulatory, or phonetic characteristics of speech in the training and testing environments differ [Stern 1996]. In this research project, speech recognition has been transferred to real application, as the need for greater robustness in recognition technology is becoming increasingly apparent. However, if users do not like or do not use the system, then all that effort will be wasted. Effective speech solutions rely on a thorough understanding of what users are trying to accomplish and how to create an engaging dialogue that helps them get the information they require. The dialogue design must consider a broad range of user factors that are concerned with making the interaction between the system and user as successful and effortless as possible for the user. In this project, the goal is to design a simple, consistent, unambiguous, helpful and representative speech-enabled interface that the user should feel to be both natural and familiar.

3.2.1 Pre-Design Studies

The design of a speech-enabled interface is often left entirely to the system designer. The designer's chosen command vocabularies, as well as chosen interaction structure are thus very important to the success or failure of the system. Furnas [Furnas 1987] has coined the phrase "armchair approach" to describe speech-enabled interfaces that are mostly designed using a designer-selected vocabulary [Prodanov 2003 and Adams 1999], as well the designer's "conceptual model" for interaction structure [Norman 1983].

The armchair approach refers to the designer's particular use of technical words that may seem natural to the designer, but which may be obscure or even meaningless and misleading to the novice target user of the speech-enabled interface. Studies conducted with typed input show that only 10%-20% of a novice user's first attempts at using their own command vocabulary matched the designer's intended commands [Furnas 1987]. Users tend to use a surprisingly large number of synonymous terms to indicate the same object or function. According to Zipf's theory [Zipf 1949], one of the basic principles of language is that, when giving names to objects and functions, a few (usually 2-3) synonymous high frequency alternatives are used as well as a large number of low frequency alternatives. This implies that for any given interaction there is a broad but positively skewed distribution of user commands available. Furthermore, the probability that the designer picks a low-frequency command to use in the interface is quite high, given that the archetypal designers' everyday use of technical terms are not typical of the sort of terms spontaneously used by the average user.

Thus, with the designer's 'armchair approach' 8-9 out of every 10 initial user attempts will fail to identify the designer's intended command. If the designer called the feature "Paste", then nine out of ten users would expect it to be called something else [Furnas 1987]. This implies that the learning time for a new interface will be prolonged, since the novice user has to learn and use the designer's way of denoting objects and functions. The alternative is the empirical user centred design approach, where the potential users of the service are asked to define the command vocabulary themselves. By using the 2-3 most frequently suggested user commands as synonyms in the speech-enabled interface's vocabulary, [Furnas 1987] and [MacDermid 1996] showed that this method was up to five times as successful as the armchair approach. This figure refers to system control using both typed and spoken input (for voice-controlled applications).

Yankelovich suggests that to avoid the need for synonymous alternatives and achieve naturalism and familiarity, any speech-enabled interface design should start with a pre-design study [Yankelovich 1998]. Other researchers suggest using the Wizard-of-Oz method as the principal mechanism for evaluating and getting input for dialogue design [Dahlbäck 1993 and Bretan 1995]. As described by [Yankelovich 1998], the pre-design study differs significantly from the Wizard-of-Oz studies that have been used extensively by others in the design of speech-only and multimodal systems. The Wizard-of-Oz is a technique to simulate a computer system that takes spoken natural language input, processes it in some principled way, and generates spoken natural language responses [Klemmer 2000]. In addition, the pre-condition for conducting a Wizard-of-Oz study is that before the experiments are begun it should be possible to formulate a detailed specification of how the future system is expected to behave [Yankelovich 1998]. The pre-design study method, on the other hand, takes place prior to any system design or functional specification. The purpose is to launch the design process.

A pre-design [Yankelovich 1998] was included in the earliest stages of the ATTAIN speech-enabled interface life cycle. It helped to define the interface functional requirements, see the task from a user's view, develop a feeling for the tone of the conversations, discover effective interaction patterns, design the prompts and acquire specific knowledge about the vocabulary and grammar used when people access the system by speaking in a true environment. Like other researchers, the pre-design studies in this project use what Norman [Norman 1991] refers to as a scenario-based approach. The pre-design study is designed to capture human-human interactions in the true environment. This data is then used as the basis for the interface design in order to ensure that the speech-enabled system dialogue closely matches the natural human dialogue. In theory, this should make the system more usable. In this research project, the ATTAIN system was intended to provide users with bus travel information. Therefore, before designing the interface, how people enquired about their journey is needed to discover. A pre-design study experiment was carried out. 12 users were arranged who ranged in age and nationality to enquire about bus information with a system operator. Because most of the users had no experience of using such a system, the system operator controlled the interaction. The users gave the permission to record their completely natural conversations with the system operator.

After collecting common ways of answering place questions (e.g. "Where do you want to travel to?"), it was found that people do not always respond with full sentences (e.g. "I would like to go to Carlton."). They often just give a short answer (e.g. "To Carlton" or "Carlton"). Some users even gave answers that are effectively meaningless to the system (e.g. "I want to go back to home" or "I want to travel to my office"), even though these answers do have meaning from the users perspective. During the interface design, these special responses would have to be handled in an intelligent way. Most speech-enabled interfaces tend to give a default answer (e.g. "Sorry, I could not understand you") in such situations. Instead of rejecting the user's response in this way, the system could be designed to ask the user clarifying questions to extract further information from the user (e.g. "ok, where is your home then?" or "I understood that you want to go to work. Please tell me where your office is."). Because the users are not being directly refused by the system, they would be likely to be less frustrated when continually using the system. In addition, to these meaningless responses there were also occasions when some of the users were confused by some of the system questions and thus gave unexpected responses (e.g. "Sorry, what do you mean?" or "Pardon"). In such cases the user had obviously had difficulty understanding the system operator's formal question. When designing the speech interface it is important that 'help handlers' be included to make the questions as clear as possible in order to guide the user though the dialogue.

During the pre-design study, the system operator often confirms the user's answer with "Ok" or "Aha" were also learned. The frequent OK's in the conversion provide the speaker with evidence of understanding in a natural way. However, it is not essential to provide the user with clear evidence that the system has correctly interpreted the user's input. Users are more likely to accept a system that makes relatively frequent errors if the system exhibits cooperative behaviours. This type of behaviour can be achieved by providing users with frequent, but unobtrusive, feedback, and allowing users to make partial corrections when they detect an error (e.g. "Ok, you want to travel from 'Arnold' to 'Beeston'…" "No, I said from 'Alford'…").

Speech-enabled interface designing in this project was an iterative process: started with the best guess, collected data from the pre-design studies, refined the grammar, obtained more user data, refined further, and so on. As designing the interface by adding and removing phrases, the dialogue more closely approximated the way users speak to a human operator.

3.2.2 Directed-Dialogue Interface Operational Overview

After learning from the pre-design study and considering all aspects of user interface design including usability testing [Nielsen 1994] and speech dialogue management (including error recovery) [Seneff 2000], a directed-dialogue speech-enabled query interface was developed for the ATTAIN travel information system using VoiceXML. The design diagram for the directed-dialogue can be found in Figure 3.3.

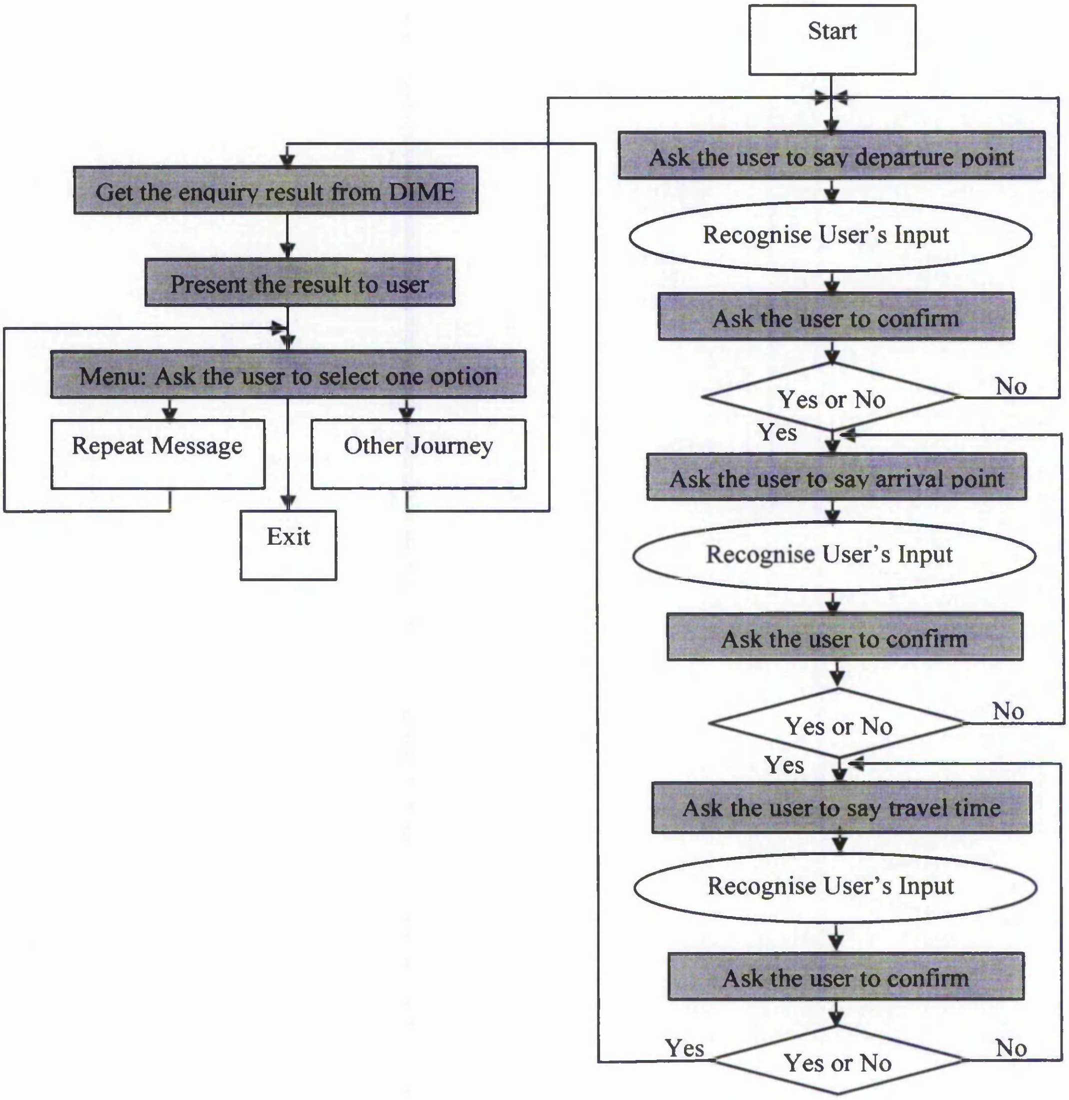


Figure 3.3 - The Design Diagram for the Directed-Dialogue Interface

This directed-dialogue interface gives the user explicit directions about what to say and when to say it. There are two primary sources for giving user prompts and conversational feedback: synthesised speech and audio. Because sound effects are important for improving the usability of telephony user interface, audio files were originally inserted for the prompts in the ATTAIN VoiceXML application. Voice user interfaces with personality have the power to make telephone-based services more human like, conversational, efficient, effective, and enjoyable. Research has found that the more familiar users are with the type of person the voice represented; the more successful they were in completing the task [CCIR 2005]. Unfortunately these audio prompts later needed to be changed to TTS only prompts because of the long delays experienced during the loading of each audio file. All of the audio files were stored in a Document Server. Before the system could play any audio file, the VoiceXML interpreter had to fetch the specified audio files from the server. The longest prompt audio file could take up to 4 seconds to fetch. This latency was detectable by the users and was felt to have an impact on the usability of the prototype interface. Recorded audio may still be the most desired output, but synthesised speech was found to be an effective substitute that maintained the important real-time usability aspect of the interface.

The user is therefore introduced to the system through a TTS welcome message and then taken through a serious of dialogues to communicate with the system. See Dialogue Figure 3.4.

System: *Welcome to the ATTAIN travel information system in Nottingham.*

System: *Please specify the bus stop code or the name of the area where you want to start your journey from.*

User: *Beeston.*

System: *I heard 'Beeston', is this correct?*

User: *Yes.*

System: *Please specify the bus stop code or the name of the area where you want to finish your journey.*

User: *The Albert Hall*

System: *Did you say 'The Albert Hall'?*

User: *Yes.*

System: *You can specify either the earliest time of departure or the latest time of arrival. Please say either 'Departure' 'Arrival' or 'Now'.*

User: *Departure.*

Figure 3.4 - A Typical Conversation (to be continued)

System: *Please specify the time of the earliest departure from Beeston.*

User: *20 past 10.*

System: *Do you want to travel after 20 past 10?*

User: *Yes.*

System: *Thank you. Please wait while I get the information you require. This may take a few moments.*

System: *Ok, you can catch Bus 36 at 10.23; it will arrive at City Centre at 10.46.*

System: *If you want to listen to this message again, please say 'Repeat Message';*
If you want to enquiry about another journey, please say 'Journey';
If you want to exit, please say 'Exit'.

User: *Repeat Message*

System: *Ok, you can catch Bus 36 at 10.23; it will arrive at City Centre at 10.46.*

System: *If you want to listen to this message again, please say 'Repeat Message';*
If you want to enquiry about another journey, please say 'Journey';
If you want to exit, please say 'Exit'.

User: *Exit.*

Figure 3.4 Continued - A Typical Conversation

Users do not have to wait for the prompt to finish before speaking. Instead, they can "*barge in*" and speak a response at any time, thereby saving time. When the interpreter detects incoming speech, it interrupts the prompt. See Figure 3.5.

System: *Please specify the bus stop code or the name of the area where you want to……(Barge in)*

User: *The Albert Hall*

System: *Did you say 'The Albert Hall'?*

……

Figure 3.5 - A Barge in Example

In the event that the user has difficulty understanding the system prompts, help handlers are included in individual form fields to make the messages as specific as possible in order to guide the user though the dialogue. See Figure 3.6.

System: *Please specify the bus stop code or the name of the area where you want to finish your journey.*

User: *Help*

System: *Please say where you want to finish your journey, for example say 'City Centre'.*

......

Figure 3.6 - A Help Example

When non-experienced users communicate with any system, they often make errors. For example, with the speech-enabled ATTAIN system the user may stay quiet for too long after a prompt or the user may say something that the interpreter cannot recognise against any active grammars. The system handlers are designed to help the user out of these difficult situations. The VoiceXML interpreter can throw a number of predefined events based on errors, telephone disconnects or user requests. A range of self-defined throw events in the system were also specified (e.g. "Exit", "Cancel", "Go Back", "Hang Up"). When an event is thrown, the associated event handler is invoked. Execution then resumes in the element that was being executed when the event was thrown.

3.3 Usability Test for the Directed-Dialogue Interface

3.3.1 Experiment Design and Materials

Two interfaces were implemented for this experiment: a conventional Text Message Interface and a Speech-enabled Interface. A script of instructions was read to each participant and they were given an overview of the system. The users were informed of the purpose of the experiment which was testing two interface's usability and the result can be used to indicate what advantages or disadvantages the speech-enabled interface has compared to the existing text based system. The questionnaires were shown so that the users knew the key aspects they were expected to notice during the experiment (Questionnaires see Appendix 1). Before using the system, the users were not taught how to use the system. All the users were required to find out one bus information enquiry on each interface, including start journey time, arrival time, bus service number and bus changing information.

The required origin and destination bus stops were randomly pre-selected from a list of 1355 bus stop names. Each of the tasks were timed and recorded along with any comments made during the completion of the tasks. A questionnaire of usability test was given to participants afterwards. The questionnaire asked the users to evaluate the features of the system according to a 1-9 type Likert rating scale [Komorita 1963]. This process was repeated for both the text message interface and the speech-enabled interface. At the end of the experiment the users were asked to select the interface they preferred.

For this experiment the confidence level of the speech recogniser was configured to 0.5. This is the confidence threshold required for the speech-recognition engine to decide whether the input speech matches a sentence from the grammar. The Timeout property was set to 5 seconds; this property specifies how long the interpreter allows the users to remain silent before they start to talk. The Complete Time Out property was set to 1 second; this property specifies how long the interpreter waits after the user stops talking before processing the speech input.

3.3.2 Participant Selection

Nielsen [Nielsen 2005] and Landaruer's previous research [Landaruer 1993] have shown that the number of usability problems found (T) in a usability test with n users is given by:

$$T=N(1-(1-L)^n) \quad \textbf{Equation 3.1}$$

Here N is the total number of usability problems in the design, L is the proportion of usability problems discovered while testing by a single user and n is the number of users. The typical value of L, found by Nielsen and Landaruer, is 31%; averaged across the large number of projects they studied. Plotting the curve for L=31% gives the following result. See Figure 3.7.

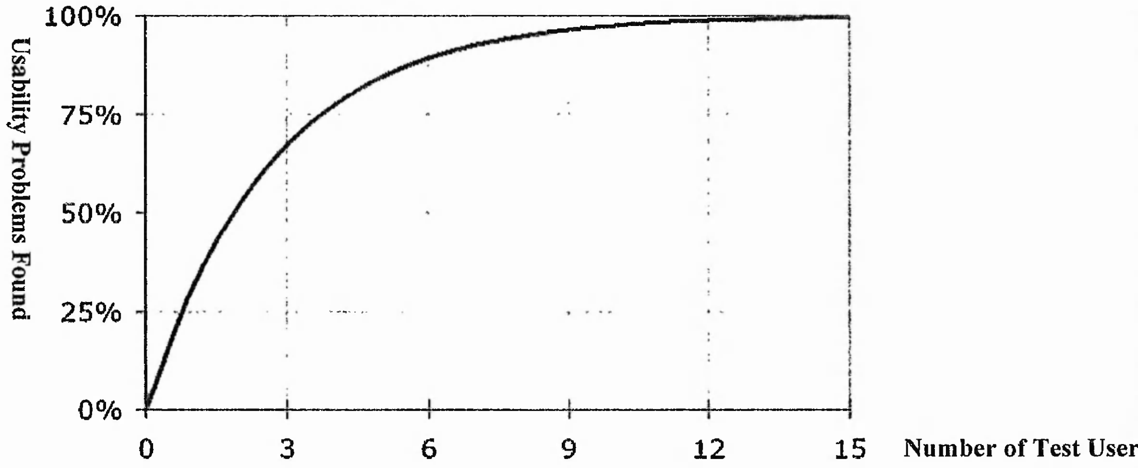


Figure 3.7 - The Usability Problems Found Curve

Figure 3.7 clearly shows that at least 15 users are needed to be tested to discover virtually all (99.62%) of the usability problems in a typical system [Nielsen 2005]. For the first usability experiment, therefore 20 users were asked to experiment with the system (none of them had previous experience of using such a system). They ranged in age from 18 to 29 years of age. According to Nielsen [Nielsen 2005] and Landaruer's theory [Landaruer 1993], 20 users should discover 99.99% of the usability's problems of this system.

3.3.3 Statistics Methods for Analysing the Usability Results

As described by [Sauro 2005A], adding confidence intervals to completion rates in usability tests tempers both excessive skepticism and overstated usability findings. Confidence intervals make testing more efficient by quickly revealing unusable tasks with very small samples. The risk of acceptance can be described using two elements: the Confidence Level, and the Width of the Confidence Level.

To create the confidence intervals for the user's preference (prefer text message interface or prefer speech interface) and complete task (complete, did not complete) responses, the binomial distribution is used, since the event is binary/binomial.

Because the rates of user's preference and complete task are not symmetrical, a technique to create asymmetric binomial confidence intervals is needed. There is a technique that uses the incomplete beta function and the F distribution. [Harte 2002] states the formula of this technique.

In order to evaluate the analytical data for the other continuous valued response items, the true mean of a population using the formula for standard deviation and the student t distribution was estimated. As described by [Sauro 2005B], this method is best for small sample size results. Equation 3.2 shows the expression used to calculate the true population mean from the results given in Table 3.1:

$$\mu = \overline{x} \pm t^{*}\left(\frac{s}{\sqrt{n}}\right)$$ **Equation 3.2**

$\overline{\mathbf{X}}$ Mean of the sample

μ True mean of the entire population of users

n Number of users in the sample

s The standard deviation of the sample

t* t statistic =TINV(.05,n-1) [confidence level(.05) and degrees of freedom n-1]

3.3.4 Results and Discussion

The mean intelligible scores for the two interfaces are presented in Table 3.1. The sample accuracy of the speech-enabled interface was manually determined by recording the user responses and comparing them to the system recognised response. The average completion time was recorded as the time taken from the user starting to interact with the system (dialling the number in the case of the speech-enabled system and entering the first text character in the case of the text based system) to completing the task. In addition, the interaction efficiency (user remembered data) was assessed by asking the users to write down what they had remembered from the audio response to their query. Also, as described in 3.3, the evaluation of the questionnaire results used the Likert rating scale to analyse four human factors of the user interface: i.e. Perceived Efficiency, Learnability, Re-learnability and User

Satisfaction. Finally, the user preference records whether the user would prefer to use the text based or speech based interface and the complete task records whether or not the user was able to get a response from the system.

	Text	Speech
Sample Accuracy	*	88.5%±5.8%
User remembered Data	+	52%±16.2%
Average Completion Time (secs)	139	111
Perceived Efficiency Level	8±0.8	5.7±0.9
Satisfaction Level	7±0.7	5.3±1.0
Learnability Level	7.8±0.7	6.2±0.8
Re-learnability Level	7.7±0.7	6.4±0.8
Users Preferences	15	5
Complete Task	20	20

Table 3.1 - Performance of Text Message Interface (Text) and Speech-enabled Interface (Speech) Systems

3.3.4.1 Effectiveness Test

The effectiveness of a system from Shackel's point of view is about whether a task can be accomplished with the specified system within a given time frame [Shackel 1986]. In this experiment, all of the users were able to 'successfully' get the bus information that they required using both interfaces. However, if measuring 'success' in terms of the output quality, the speech-interface did not give an acceptable performance because most of the users could not remember the full enquiry result. After each user had heard the bus information and disconnected the phone, the users were immediately asked whether they could remember the following five data items: Bus Service Number, Departure Time, Departure Bus Stop Name, Arrival Time and Arrival Bus Stop Name. The average user in the sample could only give 52% (μ=52%±16.2%) of the correct data. In general, this

* It is difficult to define the accuracy of text message input. It is therefore assumed the accuracy rate of the text message input to be 100%.

+ Because the users can read the enquiry result from their mobile, they do not need to remember the data.

was not enough to help the user complete the journey. [Oulasvirta 2004] states that extracting information by listening to a speech recording is much more difficult than visually scanning a document because of the transient and temporal nature of audio.

Using a binomial confidence intervals analysis of the complete task results with 95% confidence, that although 100% of the sample could get the bus information the true population figure could be as low as 83%. In addition to this, the user remembered data results shows that the audio memory capacity of humans is such that the output quality of this initial interface is too low. It is therefore difficult to say that this interface is an effective one.

3.3.4.2 Efficiency Test

The ISO efficiency definition [ISO 1998] represents how much effort is required in order to accomplish the task. An efficient system ought to require as little effort as possible. Time spent using the system can be considered as one way of measuring the efficiency of the system. To accomplish the task using the speech-enabled interface, the average user spent less time (111 seconds) than using the text message interface (139 seconds). In text message interface, the users can save their enquiry messages on their mobiles, so the user will spend much less time during future usage. In this experiment, the users were not allowed to use saved message. Interestingly, from a binomial confidence intervals analysis of the user preference comments (for Questions see Appendix 1) with 95% confidence, 51% to 91% ($\bar{X}$=70%) of the sample thought that the speech-enabled interface was actually slower than the text message interface. The reason for this false impression is probably that the user is interacting with the speech-enabled system all of the time whereas input data is entered offline in the text interface system. In addition to the perceived slowness, the users also thought that they were required to perform too many actions with the speech-enabled interface in order to accomplish the task, e.g. the users needed to make a phone call using one hand, while using their other hand to write down the result.

As cited by Bevan, the efficiency can be measured by provide further evidence of easy of use [Bevan 1991]. In some circumstances adequate performance may be achieved by the user only at the cost of considerable mental and physical effort. Therefore, in this experiment, the user perceived efficiency level of each interface was also measured using the question, "Do you think this interface is easy to use?" (1= Very Easy to 9=Very Difficulty). The users only gave an average score of 5.7 (μ=5.7±0.9) for the speech interface. The major reason, given by the users, for this efficiency level is the poor dialogue design. In the directed-dialogue interface, the system asked the user to confirm whether or not the recognised data is correct after every single user utterance. As the system recognised the user input with close to 90% recognition accuracy, the majority of users felt that they wasted time confirming the correct answers.

3.3.4.3 User Satisfaction Test

In reality, how can we say whether or not a system is effective, efficient or satisfying? User satisfaction with the interface is a very subjective measure. However, we can ask them to tell us how content they are with the system and the way in which it operates and try to convert these responses into some sort of overall measure of satisfaction with the system. Unfortunately, it still had no guarantee that one person's 10 rating was not another person's 5 or 8 rating. In this experiment, the user satisfaction measurements are based on observations of user attitudes towards the system [Faulkner 2000]. The aim in designing the interface is to promote continued and enhanced use of the interface by the user. Thus the aim of usability testing is to ensure that the user has positive feelings towards the system. It is possible to measure user attitudes using a question, such as: "How would you rate your opinion of the interface?" (1=Very Bad to 9=Very Good). People should also enjoy using the system; they should have fun. Fun is something that computerised systems can often be lacked. It is also very subjective. The users were asked to give a score (1=Strongly Disagree to 9=Strongly Agree) for the enjoyability level of each interface. Two scores were calculated from these two questions and simply averaged them to obtain the satisfaction score; which for the speech interface is only 5.3

(μ=5.3±1.0). The users gave a satisfaction score of 7.0 (μ=7.0±0.7) for the text message interface. Comparing the speech-enabled interface and text message interface satisfaction scores, the users are not very satisfied with the speech-enabled interface can be seen.

3.3.4.4 Learnability Test

It was hypothesised that the speech-interface should be easy for the user to learn so making it possible to use the system effectively as quickly as possible. Almost any system will require some amount of time for a new user to learn how to use it. This interface can be said to aim for minimum learning time, because most of the users will have knowledge of using the telephone. The measurement of learnability involved an objective measurement of just how easy the interface was to learn. The users were asked to give scores for the statement: "The system is easy to learn." (1=Strongly Disagree to 9=Strongly Agree). It can be seen from Table 3.1 that the user's response to this question for the speech interface is a mean score of 6.2 (μ=6.2±0.8). Comparing this to the text based interface score (μ=7.8±0.7), it can be deduced that the users thought that they needed to spend more time learning how to use the speech-enabled system than they did the text based system.

Learnability also contains the idea of how much effort is required to maintain the skills and the concept of re-learnability given time away from the system. This attribute can be determined by asking the users to give an opinion on the statement: 'I am able to use the system for a second time without referring to any instruction?' (1=Strongly Disagree to 9=Strongly Agree). The users gave a mean score of 6.4 (μ=6.4±0.8) for the speech-enabled system indicating that if this interface is not used regularly it could well be that the user has to constantly re-learn how to use it. Again, this was also a lower score than achieved by the text based system.

In conclusion then, the fact that the mean learnability and re-learnability scores of the speech interface are lower than the equivalent scores for the text interface implies that the users found the text based system easier to use than the speech based system. This goes against the original hypothesis that a speech-enabled interface would be more natural to use (i.e. easier to learn and remember) than a text based interface.

3.3.4.5 Feedback from Interview with Users

After each experiment, the users were interviewed and asked them to give feedback. Most of the users felt that the speech-enabled interface was inflexible. This was because the interface only allowed a single way to input data. If the ASR had a problem understanding, it simply asked them to repeat what they have said again and again. This led to frustration on the part of users who were repeatedly misunderstood. In such a situation, they would have liked to be able to use the interface in a different way. For example, they could input data using the keypad rather speech. They also do not like the way the application controlled the interaction all of the time; the user could not volunteer information. For example, they may have wished to give the information in a natural single sentence (i.e. "I want to go from Arnold to Beeston."), but the directed-dialogue interface could not understand the volunteered information. This probably explains why the users preferred the text based system. In addition, because most of participants used in this study had had experience of using text messages on a mobile phone for many years, the familiarity of such an interface tended to encourage the users to prefer the text based system.

Most of the users also felt that it was difficult for them to remember the enquiry result. This is because of the temporary nature of speech and the limitations of human memory. In any future version of the speech-enabled interface, the enquiry result must be presented in a format that can be easily assimilated and acted upon by the listener. The best way this could be done would be to send the enquiry information to the user as a text message that can be permanently saved in the user's mobile. Finally some of the users do not like the American accent in the TTS prompts; they would prefer a local voice.

3.4 Conclusions

In this chapter, the implementation of a directed-dialogue speech-enabled interface for the ATTAIN system has been discussed which focused the discussion on the characteristics and implementation issues of the directed-dialogue interface, as well as describing the complex travel task application.

From the first usability test results and user feedback, the speech-enabled interface only gave the lowest level of performance that is acceptable to the users can be seen. Although speech is assumed to be the most natural input method for communication with computer based information systems [Cox 1998], the speech-enabled interface had many problems in terms of usability performance and user satisfaction. To overcome these critical difficulties it was necessary to improve the dialogue management and information feedback mechanism. Consequently, an investigation was carried out into supporting VoiceXML based speech recognition using a multimodal interface with mixed-initiative grammar. This is described in Chapter 4.

CHAPTER 4
A MIXED-INITIATIVE DIALOGUE INTERFACE

The previous research has focussed on a creating a directed-dialogue speech-enabled interface for the ATTAIN travel information system. As a result of this research, a directed-dialogue interface was developed as detailed in Chapter 3. Unfortunately, the previous usability test results show that the directed-dialogue interface only gave a minimal level of performance. Although speech is assumed to be the most natural input method for communication with computer based information systems [Cox 1998], the previous interface had many problems in terms of usability performance and user satisfaction. In order to overcome the identifiable difficulties of the directed-dialogue interface, a multimodal interface has been developed that uses a mixed-initiative grammar. This grammar allows the interface to process natural language style input, rather than directing the user through a rigid sequence of questions and answers. The multimodal aspect of the new speech-enabled system still requires speech to be used to input the required journey information but now uses text messages, as well as audio feedback, to present the results of the search back to the user. This overcomes the human short-term memory problem present in the initial version of the interface. The design diagram of the mixed-initiative dialogue can be found in Figure 4.1.

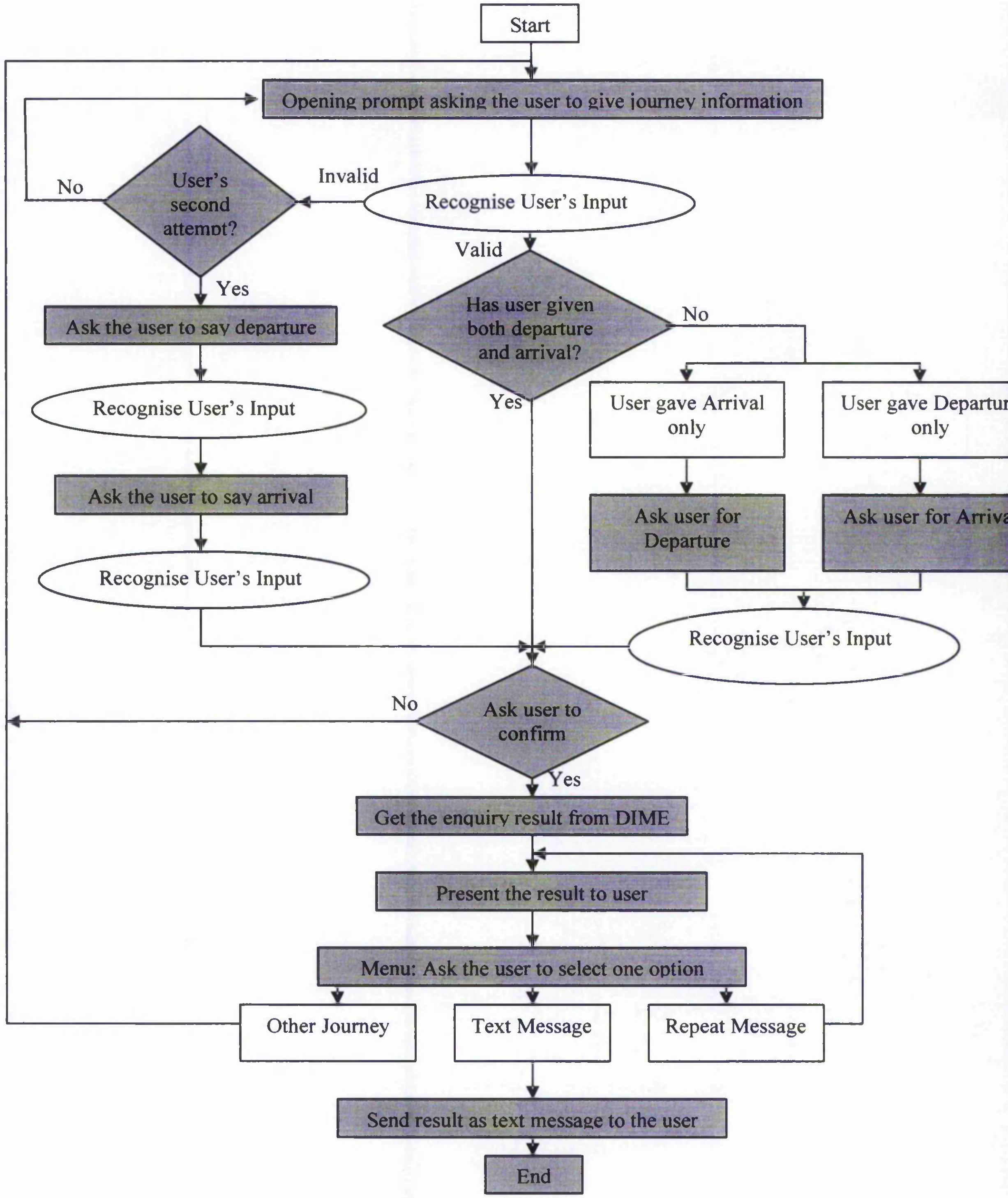


Figure 4.1 - The Design Diagram for the Mixed-Initiative Dialogue Interface

4.1 Operational Overview

The architecture and overall operation of the mixed-initiative dialogue interface are the same as the directed-dialogue interface; the user dials a telephone number to access the system using either a fixed or mobile telephone. The number provides the user access to a VoiceXML browser that supports a mixed-initiative grammar, see figure 3.1 (Chapter 3 Page 41).

The user is introduced to the system through a welcome message and an opening prompt. The user can give a very flexible answer. See Figure 4.2.

System: *Welcome to the ATTAIN travel information system in Nottingham.*

System: *Please state your journey.*

User: *I want to go from Arnold to Beeston.*

or User: *I am going to Beeston.*

or User: *To Beeston.*

……

Figure 4.2 – The Opening Prompt

In the event that the user fails to give both origin and destination in their response, the system is able to incorporate this into its state and request automatically the unfilled data. See Figure 4.3.

System: *Welcome to the ATTAIN travel information system in Nottingham.*

System: *Please state your journey.*

User: *I am going to Beeston.*

System: *Ok, you want to go to Beeston. Please state where you would like to travel from.*

User: *City Centre.*

……

Figure 4.3 –The Mixed-initiative Detection 1

Likewise, in the event that the system only recognises one of the two given journey points, the system is able to incorporate this into its state and request automatically the unrecognised data. See Figure 4.4.

System: *Welcome to the ATTAIN travel information system in Nottingham.*

System: *Please state your journey.*

User: *I want to go from Arnold to Beeston.* ***(Unrecognised Origin)***

System: *Ok, you want to go to Beeston. Please state where you would like to travel from.*

User: *Arnold.*

......

Figure 4.4 - The Mixed-initiative Detection 2

Once the system has gathered all the mandatory data (origin and destination), it asks the user for confirmation and then launches a DIME (Distributed Memory Environment) [Kosonen 1999] query. After the system has presented the enquiry result verbally, the system gives four menu options for different user preferences. See Figure 4.5.

System: *I understand that you want to go from City Centre to Beeston, Is this correct?*

User: *Yes*

System: *Thank you. Please wait while I get the information you require. This may take a few moments.*

System: *Ok, you can get bus 36 at 10.23; it will arrive at City Centre at 10.46.*

System: *If you want to listen to this message again, please say 'Repeat Message';*
If you want to enquiry about another journey, please say 'Journey';
If you want to send the result as a text message, please say 'Text Message';
If you want to exit, please say 'exit'.

User: *Text Message*

System: *Ok, the result will be sent to your mobile 07901536567. Thanks for calling the ATTAIN travel information system. Goodbye.*

Figure 4.5 - The Sending Text Message Dialogue

When a user calls from a mobile phone the user's telephone number is known automatically to the system and can be used to send the text message without any further input from the user. However, when a user calls the system from a fixed phone, the system requires the user to input his/her mobile number using either speech or keypad (i.e. DTMF).

The mixed-initiative interface also supplies event handler for 'help', 'no match'...etc in the same manner as the directed-dialogue interface.

4.2 Comparison between Mixed-initiative and Directed-Dialogue Interfaces

In the directed-dialogue interface, the users did not like the way the application controlled the interaction all of the time; the user could not volunteer information. For example, they may wish to give the information in a natural single sentence (i.e. "I want to go from Arnold to Beeston."), but the directed-dialogue interface could not understand the volunteered information. In the mixed-initiative dialogue interface, the user inputs are not constrained to single data input as in the directed-dialogue interface. Users are able to respond with multiple data. The flow of the interaction is not controlled by the system, but is controlled by the user. In a mixed-initiative dialogue, the users are allowed to respond to the system prompts in a natural language format. The challenge for the recogniser is to understand the different ways in which the goal can be expressed.

The mixed-initiative dialogue is more natural [Ferguson 2005] and lets the user perform a task in only one or two steps. For example, the user can give their origin, destination and travel time in one sentence rather than being asked to give just one piece of information at a time as in the directed-dialogue interface. However, this paradigm can be challenging for speech recognition technology and can therefore lead to a risk of higher error rates. It is also difficult to teach callers what they can and cannot say. The directed-dialogue interface is easier to manage from the system's point of view but it is cumbersome and inefficient for users, especially those with more experience of the system. The implementation of a mixed-initiative dialogue is a tree wherein the root node is the opening prompt (Please state your journey). The number of branches from each node corresponds to the number of different types of response that the system allows the user to input at that point in the dialogue. In the directed-dialogue interface, the sequential decision process model and the system controls all of the communications are used.

In the mixed-initiative interface, dealing with errors is one of the most critical factors for user acceptance [Schiaffino 2004]. The system deals with both user errors and system errors gracefully. If the system detects error conditions (e.g. a repeated request for the same attribute), the system gradually reduces the allowed scope of user input by applying more and more input constraints and specific prompts. For example, after two attempts, the system will switch to a directed-dialogue mode where it will prompt the user for one piece of information at a time. See Figure 4.6

System: *Please state your journey.*

User: *I want to go from City Centre to Beeston*

System: *Sorry, I misunderstood what you said. Could you repeat your journey using your normal tone, and speed of talking. Please state your journey again.*

User: *From City Centre to Beeston.*

System: *I am still having a problem understanding you. I will ask you for information one piece at a time, just hold on for a moment.*

System: *please state where you would like to travel from.*

User: *City Centre*

...... **(Continues similar to the directed-dialogue interface, see Figure 3.4)**

Figure 4.6 - The Error Recovery Dialogue

This mixed-initiative interface is also a multimodal interface in that it allows the use of speech to input the required journey information but uses text messages to present the results of the search back to the user. When a user calls the system from a fixed phone, the system also allows the user to input his/her mobile number using either speech or keypad. The uni-modal directed-dialogue interface could only respond using verbal messages.

4.3 Usability and Accuracy Test

As the interface had been redesigned to produce a multimodal mixed-initiative interface, a second usability study and accuracy test was carried out so as to allow a direct comparison to be made between the two speech-enabled interfaces.

4.3.1 Experimental Design and Materials

A multimodal mixed-initiative interface was implemented for this experiment. A script of instructions was read to each participant and they were given an overview of the system. The users were informed of the purpose of the experiment which was testing the interface's usability and the speech recognition accuracy. The questionnaires were shown so that the users knew the key aspects they were expected to notice during the experiment. Before using the system, the users were not taught how to use the system. All the users were required to find out bus information enquiry including start journey time, arrival time, bus service number and bus changing information. A questionnaire was given to participants at the end of each experiment. The questionnaire was similar to the one used in directed-dialogue usability experiment. The origin and destination bus stops were randomly pre-selected from a 1355 bus stop names list. Each of the tasks were timed and recorded along with any comments made during the completion of the tasks. Because the test environment could directly affect system performance, the experiment in two different environments was performed: office and noisy shopping centre.

For this experiment the confidence level of the speech recogniser was configured to 0.5. This is the confidence threshold required for the speech-recognition engine to decide whether the input speech matches a sentence from the grammar. The Timeout property was set to 5 seconds; this property specifies how long the interpreter allows the users to remain silent before they start to talk. The Complete Time Out property was set to 1 second; this property specifies how long the interpreter waits after the user stops talking before processing the speech input.

4.3.2 Participant Selection

There are 10 native and 10 non-native United Kingdom English speakers participated in the experiment. The users ranged in age from 18 to 70 years of age.

It was also felt that it would be interesting to perform an analysis of variance in performance due to age. Participates were chosen to be representative of a broad range of ages. The users were divided into three groups, ages under 25, ages between 25-45 and ages above 45. It is defined "older participations" as people over the age of 45 rather than over the age 65 as defined by other researchers [Nielsen 2002] in other HCI usability testing. In November 2002, [Chattractichar 2003] carried out a survey with 326 mobile phone owners of different age groups. After studying the survey results, they concluded that they could divide all mobile users into three age groups: below 26, 26-40, and above 40 years old according to the different user needs and responses to the mobile functions. In this experiment, the division of users into three age groups is broadly similar to Chattractichar's. There are 5 participants whose ages under 25, 9 participants whose ages between 25-45 and 6 participants whose ages over 65.

4.3.3 Results and Discussion

4.3.3.1 Accuracy

The measurement of recognition accuracy depended on whether the desired end result occurred. For example, if the user said "yes", the engine returned "yes", and the "YES" action was executed, it is clear that the desired end result was achieved. However, if the engine returns text that does not exactly match the utterance (the user said "Yeah" and the engine returned "Yes"), yet the "YES" action was executed, it can be said that the speech was recognised correctly because the dialogue processing was successful and system achieved the desired end result.

	Sample	Mean	95% Confidence Interval	Standard Deviation	t* (.95)
Accuracy Rate	20	74.5%	10.2%	21.8%	2.093
Rejection Rate	20	21.1%	10.2%	21.8%	2.093
Substitution Rate	20	4.4%	3.8%	8.2%	2.093

Table 4.1 - The Speech Recognition Accuracy Rate

Table 4.1 shows the accuracy rate in detail for the 20 users in the usability test. To evaluate the results the standard deviation with student t distribution method is used as described in Section 3.3.3. Overall there was a mean sample rejection rate of 21.1% and a mean sample substitution rate of 4.4%. This gives a mean sample accuracy rate (μ=74.5%±10.2%) that is worse than the directed-dialogue interface's average sample accuracy rate (μ=88.5%±5.8%). Even though it can be claimed with 95% confidence that the best true population mean accuracy rate of the mixed-initiative interface could be 84.7% (74.5%+10.2%), this is still lower than the directed-dialogue sample mean. As both interfaces have the same number of bus stop names (1355), it is necessary to consider what caused this drop in recognition accuracy. In the directed-dialogue interface, the grammar was constrained to one keyword input at each point in the dialogue thereby maximising the underlying ASR performance. The mixed-initiative dialogue interface is required to process a wider range of inputs capturing all possible user initiatives. This requires a larger and more complex grammar that produces a reduced ASR performance. A mixed-initiative system is a trade-off between the degree of initiative allowed and the ASR performance.

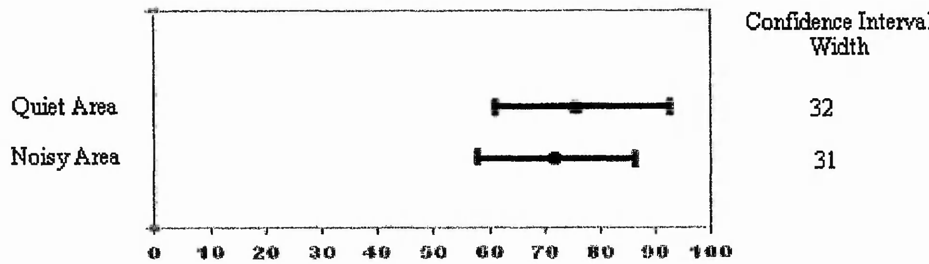


Figure 4.7 - The Accuracy Rate with 95% Confidence Intervals in Different Test Environment

To analyse the results further, the results were separated on the basis of environment, as shown in Figure 4.7. From the experiment results, the true population mean accuracy rate is between 60.4% and 92.4% in the quiet environment and between 56.8% and 87.8% in the noisy area with 95% confidence. By comparing these data values, the ASR performance are not strongly affected by the users' calling environment can be seen. It is therefore possible for the users to call the system from any environment (e.g. their homes, their offices, the mall, the airport, or their cars) without any impact on the system performance.

4.3.3.2 Usability Attributes

	Mixed-Initiative Interface	**Directed Dialogue Interface**
Sample Accuracy	74.5%	88.5%
Average Completion Time (sec)	126	111
Perceived Efficiency Level	7.6	5.7
Satisfaction Level	7.0	5.3
Learnability Level	6.8	6.2
Re-learnabiltiy Level	7.1	6.4

Table 4.2 - Performance of Directed and Mixed-Initiative Dialogue Interface

The mean intelligibility scores for the two speech-enabled interfaces are presented in Table 4.2. The results given for the directed-dialogue interface are those given in Table 3.1. They are reproduced here for ease of comparison. To accomplish the task using the mixed-initiative dialogue interface, the average user spent more time (126s) than using the directed-dialogue interface (111s). However, by analysing the efficiency level ratings of the two interfaces it can be inferred that the average user thought that they accomplished the task more efficiently using the mixed-initiative dialogue interface. The users gave a mean score of 7.6 (μ=7.6±0.7) for the efficiency level of the mixed-initiative dialogue interface compared to the directed-dialogue interface's 5.7 (μ=5.7±0.9). It would appear that time to complete and recognition rate are not critical factors as far as user perceived efficiency is concerned. A better alternative is to relate efficiency to the number of operations performed during the task. For example, allowing single user utterances containing multiple data inputs gives an impression of greater efficiency even if the recognition rate is lowered and the average time to complete is increased.

Looking at the satisfaction scores, it is apparent that the users were more satisfied with the mixed-initiative interface. All the users scored over the neutral satisfaction threshold and the average score was 7.0 (μ=7.0±0.6). This confirms the hypothesis that a mixed-initiative dialogue interface would be better accepted by the users than the original directed-dialogue interface 5.3 (μ=5.3±1.0).

[Peissner 2002] states that the usability of the system is mainly determined by the functioning of the ASR engine; accurate recognition is a reliable guarantee for efficient and satisfying use. In Peissner's view, a system with a very accurate speech recogniser should have a high level of correlation to user satisfaction. Conversely, this high correlation also means that low recognition rates will be associated with poor usability. However, the mixed-initiative speech interface experiments presented here indicate that the time to complete task and recognition rate are not critical factors for user's satisfaction. I argue that usability of the system is not solely determined by the ASR performance. A poor ASR performance does not necessarily imply mean a low probability for task completion and low user satisfaction levels. As long as the system has effective ways of detecting, correcting and recovering from recognition errors, a low ASR performing mixed-initiative interface is preferred over a higher ASR performing directed-dialogue interface.

The results also indicate that the mixed-initiative dialogue interface was easier for a user to learn than the directed-dialogue interface. With a score of 6.8 (μ=6.8±0.7) compared to 6.2 (μ=6.2±0.8), it is can be seen that it is easier for a new user to become acquainted with the mixed-initiative dialogue interface than with the directed-dialogue one. By questioning the users, several user behaviours were implicitly learned during system usage (e.g. voice loudness, absence of extraneous utterances, etc.) were able to identify. Consequently the users gave the mixed-initiative interface a re-learnability score of 7.1 (μ=7.1±0.8). This is also an improvement on the directed-dialogue interface.

Finally, the mixed-initiative dialogue interface can be seen to be more effective in accomplishing the task. In both experiments, all of the users 'successfully' achieved the bus information that they were required to obtain. However, in the case of the mixed-initiative interface, 100% of the users said they could complete their journey following the enquiry result. The mixed-initiative interface users did not need to remember or write down the result, because all of the result was saved on their mobile. This multimodal approach completely overcame the short term human memory problem evident in the directed-dialogue interface.

The mixed-initiative dialogue also has the capability of dropping down, or moving up, to an appropriate interaction level on the basis of users' interactive behavior. This approach shows that one way to minimise user difficulty with speech-enabled interface is to give control of the interactive process and in particular the presentation of auditory information, to the user. The users can ask for the result to be repeated, so the memory problems arising from the use of speech might seem to be obviated. But problems remain. How, for example, are users to be informed of these capabilities and the way to invoke them?

4.3.3.3 Results According to User Age

From Figure 4.8, it can be seen that the older participants have lower speech recognition accuracy than either the younger individuals and the middle age people. With 95% confident, the true population mean accuracy rate is between 47.3% and 75.1% for this older age group. This is much lower than the middle age group

(68.8%-99%) and the younger individuals (51.4%-96.2%). Older people who spoke too slowly and ambiguously caused this problem. The timeout property of the speech recogniser was not set long enough to wait for the older people to finish a long utterance. [Rybczynski 1995] states that, throughout history, people have been afraid of accepting new technology and during these experiments, it was found that some of the over 45 group refused to use this system because of a fear of technology. Of those old people who did take part in the experiments, it was also found some who had never talked on a mobile phone before. These reasons also contributed to the low speech recognition accuracy rate for this group. When people design speech-enabled interfaces, they need to consider the potential hearing loss, reduced memory capacity and slow talking speed of older individuals.

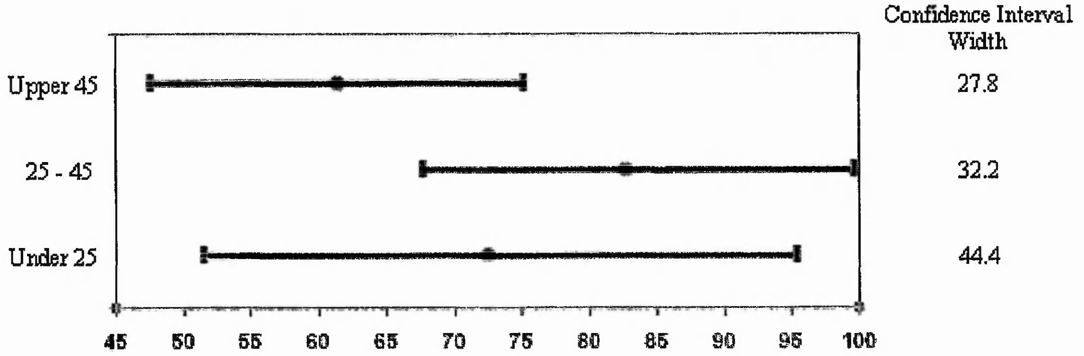


Figure 4.8 - Speech Recognition Accuracy Based on Users Age

From figure 4.8, with also 95% confident, the true population mean accuracy rate is between 51.4% and 96.2% for the younger individuals. This is a very wide spread and needs some explanation. From the observations during the experiment, it was found that some of the younger individuals did not speak in a normal manner. After they discovered that the mixed-initiative dialogue interface could understand a full sentence, some of them tried to challenge the system with special English sentences that the system failed to recognise (e.g. "I hate changing buses, please tell me a direct bus from Arnold to Beeston,"). In addition, some of younger individual spoke very quickly with strong accents. Because this is their natural way of speaking, the system cannot expect that they will change their way of talking. Therefore in any future interface, the system should give clear prompts to avoid the young users giving over long sentences.

Overall, these results are at variance with the investigation [Wilpen 1996] who showed that the word error rate of speech recognition is stable within the range of 15 to 70 years. In their experiment, the performance was only degraded for children and elderly users above 70. In this experiment, it can be seen that the performance of speech-enabled interface is significantly reduced above a user age of 45. In real life, the majority of buses users are likely to be the elderly and young people, so the system should give the behaviour of these two groups of people more consideration in any future dialogue design.

4.3.3.4 Error Recovery

[Boyce 1999] suggests that if a system does not understand the user's utterance, then it is best to simply ask the user to repeat his or her request rather than asking for confirmation of information. They refer to this step as a reprompt. Often human-computer dialogues take the form of "Sorry, please repeat". The system admits culpability, then as quickly as possible asks for the information to be repeated.

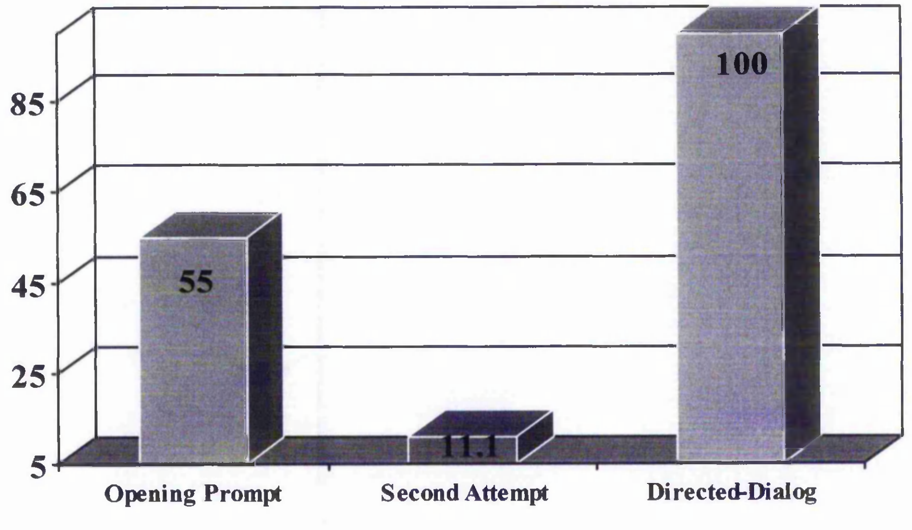


Figure 4.9 - Results in Error Recovery

In the mixed-initiative interface, if the system detects a recognition error in the opening prompt (Please state your journey.), the system will ask the user to repeat what they said by playing a different prompt (See Figure 4.6 on Page 67). Changing the prompts is likely to lead the user to respond differently. If the system still has difficulty recognising the second attempt, the system will switch to a directed-dialogue mode where it will prompt the user for information one piece at a time.

Figure 4.9 shows the percentage of users who completed the tasks in the different error recovery levels. It can be seen that 55% (11 out of 20) of the users were able to complete their query at the initial prompt level. However of the 45% of users who failed the opening prompt stage, the system could only recognise 11.1% (1 out of 9) of user utterances in the second attempt level. The same recognition error was repeated when the user spoke the similar words or phrases again; even though the system has supplied a progressive assistant prompt message. However, when the system reverts to the directed-dialogue format following the second attempt failure it can be seen that all of the remaining user responses were correctly recognised. In contrast to the finding of Boyce, it can be seen that asking the user to repeat a sentence level response is not an effective error recovery method. This stage will be removed in future versions and the system will revert to a directed-dialogue as the first level error recovery method.

4.4 Conclusion

In this chapter the implementation of a mixed-initiative dialogue multimodal interface has been discussed. The users' usability test results and feedback according to the users show that the mixed-initiative speech-enabled interface gives a more acceptable level of performance than the directed-dialogue interface. In the mixed-initiative interface, both the users and the system can play an active role. Users can respond to the system prompts by speaking freely and naturally, providing relevant information in the order they choose, whilst the system can dynamically re-configure its dialogue to respond accordingly. The multimodal aspect of the interface also overcomes the human short term audio memory problem by providing the information in the form of a text message. That said, the mixed-initiative interface does still have problems in terms of its reduced ASR performance and potential lack of repeated error recovery methods. These could do with being addressed should any commercial application of the system be undertaken.

In the mixed-initiative dialogue interface, the mean sample ASR accuracy rate is 74.5% (μ=74.5%±10.2%). This has the potential to be improved upon. The current interface only returns a single utterance string as the result of each speech-recognition event. If the user's response is not clear, the result will be the one utterance that the speech-recognition engine judges to be the most likely. However, instead of returning the single most likely utterance, it is possible for the speech-recognition engine to return a list of the most likely utterances. Confidence-scoring post-processing can then be investigated as a way of improving the speech recognition error handling. As described by [Pitrelli 2003], confidence-scoring post-processing uses a recognition verification strategy: the computer calculates a measure of recognition confidence, and if it is above a threshold it can accept the recognition result, thereby automating the processing. When the confidence value is below the threshold, the result can be "rejected" automatically, meaning that the recognition result is assumed to be unreliable and the item needs to be re-entered. Confidence scoring may consist of a function of appropriate parameters drawn directly from the recognition process, or it may be considered a learning task in which a classifier is trained to use an array of such parameters to distinguish correct recognition from incorrect. In future research, aiming to use a post-processing approach to confidence scoring, confidence is measured following recognition. It may also be possible to use confidence measures to eliminate the need for the confirmation stage in the dialogue. If the system can determine automatically that it is 'confident' in its recognition results the there is no need for the system to ask the user if the understood message is correct (See Figure 4.5 on Page 65). Subsequent usability experiments on the new interface will be carried out after the system is optimised in terms of performance.

CHAPTER 5
SPEECH INTERFACE WITH A LARGE GRAMMAR

With a system response (recognition) time of 1.2 seconds, the mixed-initiative speech-enabled query interface for the Nottingham travel information system could be accepted by the users as a real time application. However, this interface is based on a medium sized grammar file which contains 1355 bus stop names. If this interface were to use a large grammar file, both the response time and the recognition performance could be significantly degraded. In recent years, PC based automatic speech recognition (ASR) systems using large vocabulary continuous ASR have claimed significant improvements [Tang 2005] and there are now several commercial systems on the market (ViaVoice of IBM, SAPI of Microsoft and NaturallySpeaking of Dragon etc.). However, all of these systems only achieve their optimum performance when used in certain environments (i.e. speaker-dependent etc). Currently, there is little published research on the use of large grammar files in VoiceXML based systems, especially for real time applications.

[ManÉ 1996] gives an explanation as to why there has been such a lag between the impressive developments in research labs and the release of viable speech products capable of real time application. A major factor is that the research systems often performed superbly in the laboratory on the tasks and domains for which they were designed, but encounter several difficulties when used to create commercial products. For example, ManÉ argues that accuracy on complex tasks with large vocabularies is still not high enough for real time systems. This is because: research systems are often trained using speech collected under unrealistic conditions (e.g., “read” speech recorded in a studio environment); the emphasis of research systems is on accuracy and not rejection - critically important for real world applications; and there is little emphasis placed on developing research systems that could run in real time. This chapter aims to address some of these shortfalls by investigating techniques for enabling real time applications development of large vocabulary VoiceXML based systems.

5.1 Large Grammar Issues in a London Bus Travel Application

There are 27792 bus stop names in the London area. This is much bigger than the Nottingham system which contains only 1355 bus stop names. This increase in the number of bus stop names was expected to produce an increase in processing time and a reduction in recognition accuracy. The experimental results of the Nottingham travel information speech-enabled system also show that the mixed-initiative interface gives a mean sample accuracy rate of 74.5% compared to the directed-dialogue interface's average sample accuracy rate of 88.5%. The ASR performance of mixed-initiative interface is lower than the directed-dialogue interface because the mixed-initiative dialogue interface is required to process a wider range of inputs capturing all possible user initiatives. A directed-dialogue was therefore chosen for the speech interface to the London system rather than a mixed-dialogue because using a directed-dialogue would help minimise both these critical performance parameters.

The first step in designing on effective large grammar system was to simplify the grammar file. Initially, 27792 bus stop names with their associated postcodes were collected into a single large grammar file for London. In order to provide the processing time of a single bus stop speech recognition using this grammar, an initial experiment was implemented. The system recognised 100 bus stops from a male age 26 speakers. The bus stops were randomly pre-selected from the bus stop names list in the large grammar file. The experiment indicated that single bus stop name recognition would take 16 seconds using this grammar.

By analysing the bus stop names, it was easy to find that many bus stop names appear more than once with different postcodes (i.e. Abbey Road appears 5 time - see Table 5.1). It is usual in densely populated areas a street have more than one postcode and some same bus stop names indicate the different geographical place. These repeated bus stop names could be simplified by replacing them with single entries in the grammar file. When the users then enquire about a journey using these bus stop names (i.e. "I want to go from Abbey Road.", "I want to go to York Road"), the system has to detect which Abbey Road or York Road the user means. As these bus stop names are differentiated by their postcodes, the system was designed to ask the user to give the bus stop postcodes in order to distinguish between them. Flags were used to indicate bus stops names that had multiple occurrences in the grammar file. After the repeated bus stop names were deleted, a 23337 bus stop name grammar remained.

Bus Stop Name	Postcode	Bus Stop Name	Postcode	Bus Stop Name	Postcode
Abbey Road	E15	York Road	E7	York Road	SE1
Abbey Road	NW6	York Road	E10	York Road	SW11
Abbey Road	NW8	York Road	E17	York Road	SW18
Abbey Road	SE2	York Road	N11	York Road	SW19
Abbey Road	SW19	York Road	N18	York Road	W3
York Road	E4	York Road	N21	York Road	W5

Table 5.1 - Example of Same Bus Stop Name with Different Postcodes

The system needed a significant amount of time (8 minutes) to compile this large grammar file. However, this only affects the system the first time references are made to a particular source grammar file in a VoiceXML document. Once the compiled grammar file is cached by the interpreter, subsequent references to the same source grammar file then use the compiled file. The grammar file is not recompiled unless it is modified.

In a single large grammar file London bus travel application, any user perceived latency would be mainly caused by the automatic speech recogniser needing to match the voice input against every entry in the large grammar file. The best way to minimise this delay would be to reduce the grammar file size, so minimising the recognition time. Avoiding repeated words in the grammar file is one useful method for reducing the grammar file size. Many bus stop names start with the same first words but have different endings. See Table 5.2.

Bus Stop Names	Bus Stop Names	Bus Stop Names	Bus Stop Names
Abbey Gardens	Abbey Parade	Abbey View	Abbots Place
Abbey Grove	Abbey Road	Abbots Gardens	Abbots Road
Abbey Lane	Abbey Street	Abbots Manor	Abbots Terrace
Abbey Mews	Abbey Terrace	Abbots Park	Abbots Walk

Table 5.2 - Example of Bus Stop Names that have the Same First Words and Different Word Endings

It is not necessary to write every single word into the grammar file. It just needs to have one entry for each start word in the grammar file with separate entries for the different endings. For example [...(Abbey Road) (Abbey Gardens) (Abbey Grove)...] Because the bold word Abbey is repeated in other bus stop names, it just needs to write one Abbey in the grammar file. See Table 5.3.

Old Grammar File	New Grammar File
Names[([...... (abbey drive) (abbey gardens) (abbey grove) (abbey lane) (abbey mews) (abbey parade) (abbey road) (abbey street) (abbey terrace) (abbey view) (abbots gardens) (abbots manor) (abbots park) (abbots place) (abbots road) (abbots terrace) (abbots walk) ......])]	Names[([...... abbey abbots......] [...... Drive gardens grove lane mews parade road street manor park place terrace walk ......])]

Table 5.3 - Rebuilding the Grammar with One Same Start Word

When users call this large single grammar file directed-dialogue system, the speech recogniser takes approximately 13 seconds to process one user's single input (i.e. one bus stop name). During this processing, the users hear nothing from the system. Ideally, the demand an application places on the network should be transparent to the caller, and the system should appear to be instantaneously responsive regardless of the amount of data being processed. In reality, speech recognition is computationally intensive, and its demands increase with the complexity and size of the grammar. With a 23337 name grammar the latency of the system's response time resulting from the extensive computation required for just one single input is easily perceived and is unacceptable.

When designing speech interfaces, a common HCI problem that emerges involves the users' inability to interpret silence. In speech-only systems, silence can either mean that the speech recogniser did not hear an utterance or that it is processing the user's input. In such situations the users tend to assume that a lengthy silence means that the system did not hear the request. Researchers state that the users can not tolerant the latencies longer than 8 seconds [Levow 1997]. The default duration of the hourglass (latency of system processing) consists of two intervals: See figure 5.1.

- ❖ From end of speech detection to end of recognition.
- ❖ From end of recognition until the next prompt is reached.

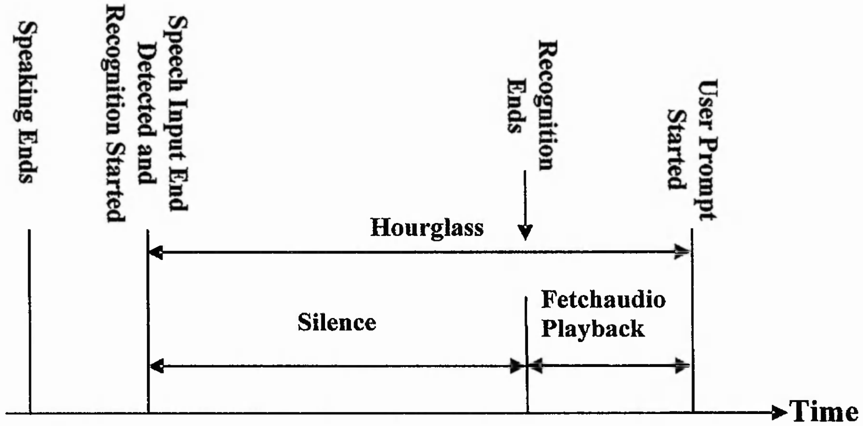


Figure 5.1 - Recognition Response

In most small or medium grammar file applications, where the recognition time is small, the caller perceived latencies would usually occur during the second interval when the interpreter is typically fetching VoiceXML documents over the Internet. For this reason, the VoiceXML specification includes a fetchaudio attribute on elements that perform fetches. This feature allows the application to specify audio that can be played from the time the fetch is attempted until the time another fetch or the next listen state is reached. In this way, the hourglass is terminated as soon as the fetchaudio begins playing. By employing fetchaudio, the user perceived duration of the hourglass only occurs during the first interval - the fetchaudio covers the second interval. The system can cover this remaining latency with a percolation sound that provides the user with a hint that the system is processing the input. However, in the London bus information system case, the resulting recognition takes such a long time, that the use of a percolation audio is no help. The users are unlikely to spend 13 seconds listening to a percolation audio whilst waiting for the system to respond to their input. Researchers state that latencies longer than 8 seconds cannot be effectively masked [Levow 1997]. As it is not possible to simplify the grammar any further another more efficient grammar construct needs to be built to cover this long recognition latency problem.

In order to provide a baseline of speech recognition accuracy the large grammar, an initial experiment was implemented. The system recognised 100 bus stops from a male age 26 speakers. The bus stops were randomly pre-selected from the bus stop names list in the large grammar file. The experiment indicated that single bus stop name recognition would take 16 seconds using this grammar. In terms of recognition accuracy, the initial experimental results show that there was a mean sample accuracy rate of 53% (μ=53%±4.9%) in the large grammar file (23337 bus stop names) directed-dialogue system (The experimental description see Section 5.1). The London interface recognition accuracy is therefore significantly worse than the Nottingham directed-dialogue interface's (1355 bus stop name grammar) average sample accuracy rate of 88.5% (μ=88.5%±5.8%). Even though it can be claimed with 95% confidence that the best true population mean accuracy rate of the large

London grammar interface could be 57.9%, this is still substantially lower than the directed-dialogue sample mean of 88.5%. However, this result is reasonable because the number of bus stop names in the London system is 17 times more than for the Nottingham system. Smaller grammars are easier for a computer to recognise, while larger grammars are more difficult because of increased ambiguity between words. Deroo obtained common error rates in laboratory experiments on speaker-independent isolated word databases of around 1% for a 100 words grammar, 3% for 600 words and 10% for 8000 words [Deroo 1998]. Young obtained error rates of around 15% for a 65000 word vocabulary with a speaker-independent continuous speech recognition system [Young 1997]. This is obviously significantly better than the London bus travel information system, but the phonetic sound of the bus stop names are more similar than the words used in Young's dictation system. This leads to an increase in substitution, making the recognition accuracy worse than in a system such as Young's which uses a common word grammar. Young's experimental result was also under laboratory condition which is another reason for that system having better recognition results than the London real time system. If the public were to accept the London speech-enabled bus travel information system, the accuracy rate of the system would have to be improved as well as the system response time.

5.2 First Letter Based Grammar Reduction System

In the large single grammar London application, the caller perceived latencies occur whilst the speech recogniser is processing the user's input against the large grammar. Dividing a large grammar file into many smaller grammar files is one possible way of reducing this latency. These smaller grammar files can then be processed individually by selecting only one of the smaller files using contextual information.

5.2.1 Dividing a Large Grammar File into Many Small Files

The large London application grammar file can be divided into 26 small grammar files with each grammar file only containing the bus stop names that have the same first letter. See Table 5.4. The design diagram for the directed-dialogue can be found in Figure 5.2.

Large Grammar File	Small Grammar Files
Large[([(aaron hill road) (abberley mews) (abbess close) …… (baalbec road) (babmaes street) (bailey mews) …… (zander court) (zennor road) (zetland street)])]	**A Grammar File** A[([(aaron hill road) (abberley mews) (abbess close) ……])] **B Grammar File** B[([(baalbec road) (babmaes street) (bailey mews) ……])] …… **Z Grammar File** Z[([(zander court) (zennor road) (zetland street) ……])]

Table 5.4 - Example of How the Large Grammar File could be Divided into Many Small Grammar Files

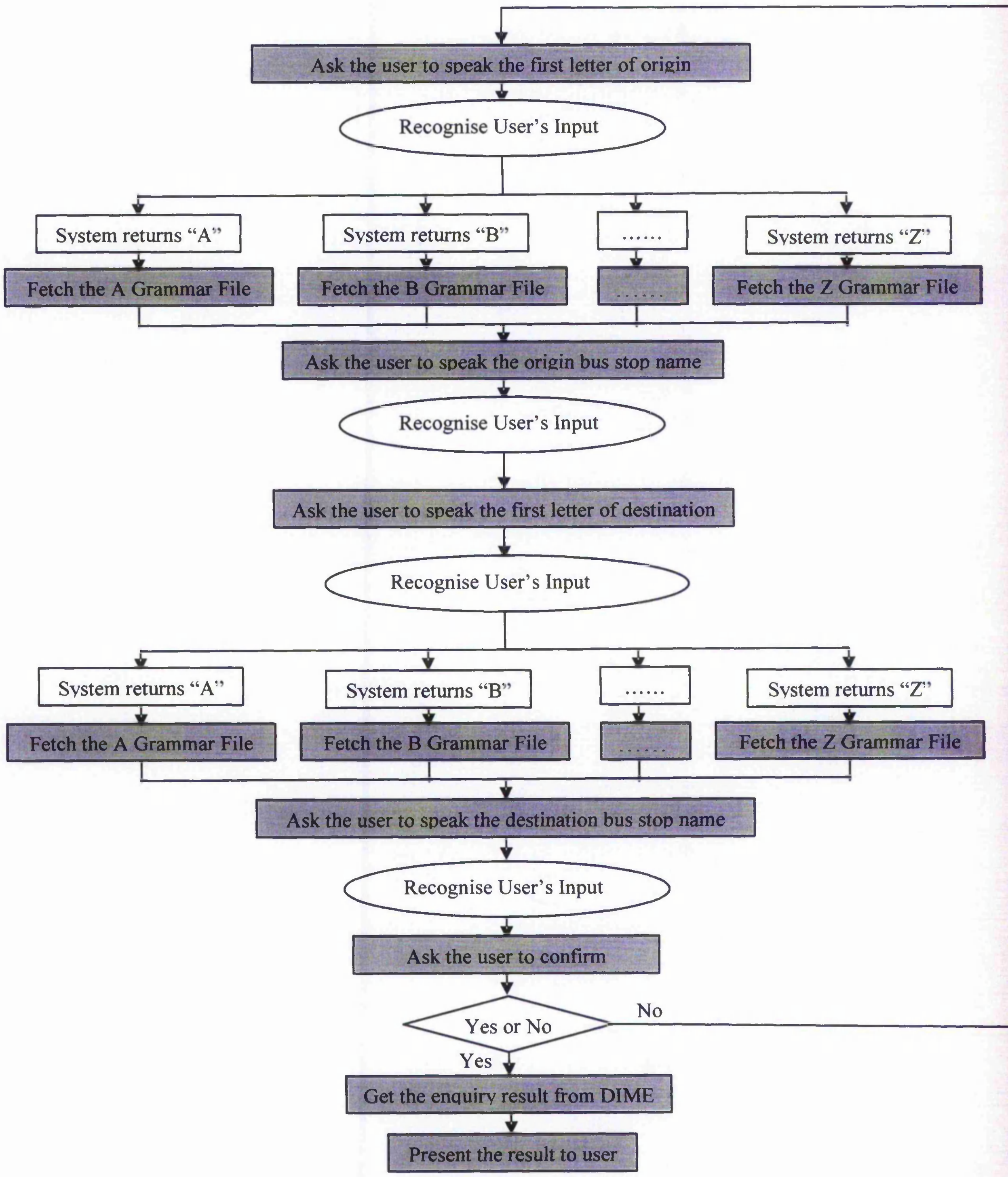


Figure 5.2 - The Design Diagram for the First Letter Based London Interface

The directed-dialogue speech-enabled query interface for a first letter based ATTAIN travel information system using VoiceXML is shown in Figure 5.3. First, the users need to supply the first letter of their origin or destination to the system. After the system has recognised the first letter of the bus stop names, the system fetches the one relevant small grammar file corresponding to this recognised first letter. The system then asks the user to give the full bus stop name and processes the user's input against this fetched grammar file.

***System**: Please say the first letter of your origin.*

***User:** A*

(System fetches the "A" grammar file.)

***System**: Please say your origin.*

***User:** Abbess Close*

(System recognises the user input using the "A" grammar file.)

***System**: Please say the first letter of your Destination.*

***User**: B*

(System fetches the "B" grammar file.)

***System:** Please say your destination.*

***User:** Bailey Mews*

(System recognises the user input using "B" grammar file.)

......

Figure 5.3 - First Letter Based Speech Interface for London Application

5.2.2 Confusion Matrix for First Letter Recognition Experiment

In the first letter recognition system, a critical problem is that the system has to be 100% accurate in its recognition of the first letter of the user's origin or destination. If the system wrongly recognises the first letter, then the system will fetch the wrong small grammar file. Using this incorrect grammar file, the user's input will produce either an out of grammar (rejection) error or a substitution error. Because the phonetic sound of single letters is fairly ambiguous, the speech recogniser can easily misrecognise them. For example, the letter "D" sounds very like "T", whilst "F" sounds like "S" etc. If users were prepared to learn the police letter alphabet, Alpha, Bravo, Charlie, Delta etc, this would be a solution. However, in terms of usability, this method is not a good one, because the users need to put extra effort into using such a system. In addition, not everybody would know these alphabets and not everybody could remember all of them. It was therefore necessary to find a reliable way of making sure that the first letter could be 100% correctly recognised without affecting the user's motivation. Combining the small grammar files of the easily confused letters into one medium sized grammar file was considered a good solution. This approach could be considered as negating some of the advantage of separating the original single grammar file on the basis of first letter, however, as long as there is some separation of bus stop names then there will be some improvement in recognition accuracy and processing time. Unfortunately, there is little research showing which letters are easily confused by a speech recognition system processing narrowband speech input. The following experiment was performed to construct a first letter confusability matrix.

5.2.2.1 Experimental Design and Materials

A script of instructions was read to each participant and they were given an overview of the system. The users were informed of the purpose of the experiment which was testing the speech recognition accuracy of letters. Each speaker was simply asked to speak the letters A, B, C......Z. The system recognised these letters using a simple grammar file containing only the 26 letters. Each of the tests was recorded along with any comments made during the completion of the tasks. Because the test environment could directly affect system performance, the experiment in two different environments was performed: office and noisy shopping centre.

For this experiment the confidence level of the speech recogniser was configured to 0.5. This is the confidence threshold required for the speech-recognition engine to decide whether the input speech matches a letter from the grammar. The Timeout property was set to 5 seconds; this property specifies how long the interpreter allows the users to remain silent before they start to talk. The Complete Time Out property was set to 1 second; this property specifies how long the interpreter waits after the user stops talking before processing the speech input.

5.2.2.2 Participant Selection

There are 10 native and 10 non-native United Kingdom English speakers participated in the experiment. The users ranged in age from 18 to 70 years of age.

5.2.2.3 Results and Discussion

Figure 5.4 clearly shows that the most difficult letter to recognise is "T"; which was only recognised 5 times during 20 tests. The Letter "W" was recognised correctly 100% of the time.

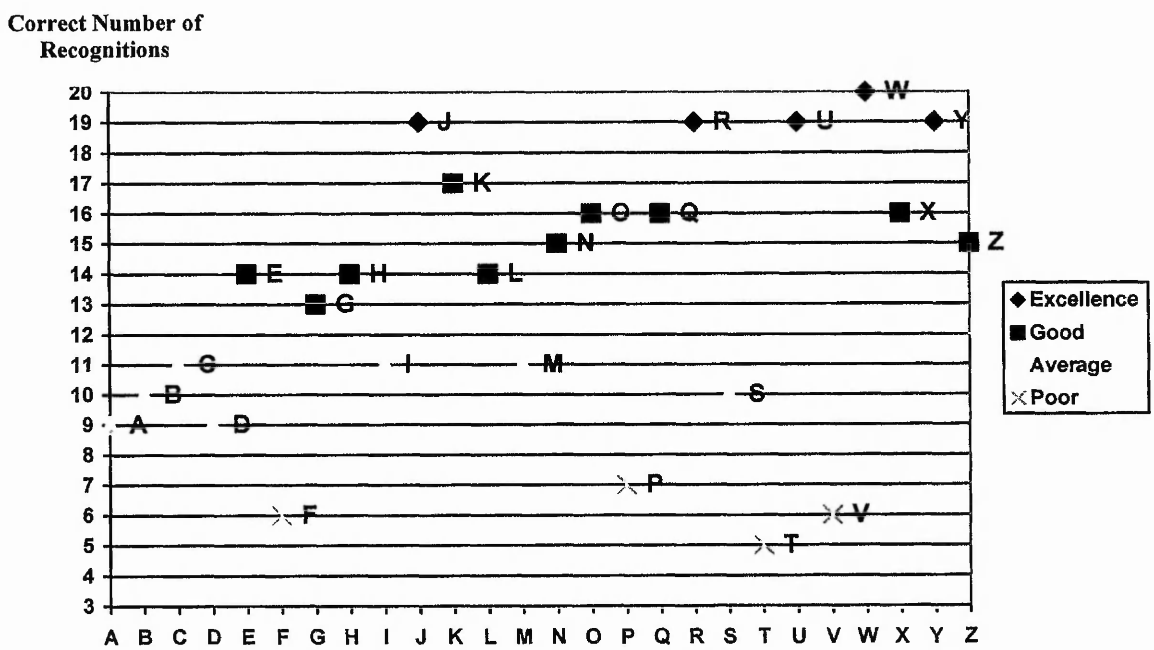


Figure 5.4 - The Accuracy Rates of Letters Recognition

The common substitutions for the 26 letters from the experimental results can be found in table 5.5. For example, when the user spoke the letter "A", the system correctly recognised "A" 9 times during the 20 tests and gave the substitution letter "K" 6 times and the substitution letter "H" 5 times. When the user spoke the letter "L", the system returned the command "HELP" (which is the default command from the system) 4 times. That apart, most of the substitutions for the specific letters are with their confusability partners. For example, to a speech recogniser, the letters "B", "E" and "V" sound very alike.

Spoken Letter	Correctly Recognised		The Most Substitution		Other Substitutions	
	Letter	Times	Letter	Times	Letter	Times
A	A	9/20	K	6/20	H	5/20
B	B	10/20	E	6/20	V	4/20
C	C	11/20	E	5/2	D	4/20
D	D	9/20	E	7/20	B	420
E	E	14/20	D	4/20	C,B	2/20
F	F	6/20	S	7/20	X,N	7/20
G	G	13/20	J	5/20	D	2/20
H	H	14/20	L	4/20	K	2/20
I	I	11/20	Y	7/20	R	2/20
J	J	19/20	K	1/20		
K	K	17/20	J	2/20	A	1/20
L	L	14/20	HELP	4/20	O	2/20
M	M	11/20	N	7/20	G	2/20
N	N	15/20	M	5/20		
O	O	16/20	M	3/20	L	1/20
P	P	7/20	B	7/20	D	6/20
Q	Q	16/20	U	4/20		
R	R	19/20	M	1/20		
S	S	10/20	X	6/20	F	4/20
T	T	4/20	C	5/20	E, D, G,P	11/20
U	U	19/20	Q	1/20		
V	V	6/20	B	6/20	B,C,E,D	8/20
W	W	20/20				
X	X	16/20	EXIT	4/20		
Y	Y	19/20	I	1/20		
Z	Z	15/20	N	4/20	EXIT	1/20

Table 5.5 - The Recognition Results for 26 Letters

To further evaluate the experimental result, the correlation between the letter recognition results was found. For example, when the user speaks the letter "F", the speech recogniser returns "S" 7 times of 20 and returns the correct result "F" 6 times of 20. This means that when the speech recogniser gives the result as a letter "S", the users could have spoken "F". The letters "F" and "S" are easily confused by the speech recogniser and these two letters can be putted into one confusing letter matrix. The other letters have similar correlations. See Table 5.6.

Recognised Letter	Substituted (confused) for							Recognised Letter	Substituted (confused) for		
A	J	K						**N**	M	F	Z
B	D	E	V	P				**O**	L		
C	E	T						**P**			
D	B	C	E	G	P	T	V	**Q**	U		
E	B	C	D					**R**	I		
F	S							**S**	F		
G	M							**T**			
H	A							**U**	Q	W	
I	Y							**V**	B		
J	G							**W**			
K	A	H						**X**	F		
L	H	O						**Y**	I		
M	N	O						**Z**			

Table 5.6 - The Confusing Letter Matrix

Table 5.6 shows that when the speech recogniser gives the result as the letter "K", it could be that the user has spoken the letter "A" or letter "H". The worst substitution result is letter "D". This is because a "D" recognition could have resulted from the user speaking the letters "B", "C", "E", "G", "P", "T" or "V". The best recognition results are for the letters "P", "T", "W" and "Z". When the speech recogniser returns one of these four letters, the users had not spoken any other letter. In constructing the confusion matrix shown in Table 5.6, it should be noted that the experimental results shown in Table 5.5 are only based on the first returned letter from the speech recogniser. When the experimental results were collected, the best two recognition results from the n-best list of each spoken letter were also recorded. For example, when the users spoke the letter "J", the system correctly recognised "J" 19 times during the 20 tests (the best recognition result from the n-best list) and gave "A" as the second recognition result from the n-best list 13 times. Although "A" was never returned as the best recognition result when the user spoke the letter "J", the letters "A" and "J" were still putted into the same confusion matrix because the system returned letter "A" as the second recognition results many times for the "J" input. In a similar manner "D" is included in the "B" confusion matrix even though "D" was never produced as the top recognition result for a "B" input. On the hand, it can be seen from Table 5.5 that letter the "N" was recognised when the user spoke the letter "F". However, the reason this substitution occurred was found to be that the user coughed during the experiment. In this situation, not to put the letters "N" and "F" into the same confusion matrix are chose. After consideration of all of these exceptions, the final confusing letter matrix is as shown in Table 5.6.

This matrix was used to help build new grammar files made up of bus stop name that began with the same first letter and its confusion matrix first letter equivalents. In this way, even if the speech recogniser incorrectly recognised the first letter, the system could still fetch the correct grammar file for bus stop names recognition, as long as the recognised letter is in the same matrix as the user spoken letter. For example, the system combined the bus stop names starting with letters "A", "J" and "K" (they are in same matrix.) into one grammar file (AJK.grammar). If a user wants to go from King's Cross and gives the first letter "K" to the system, the system could return the first letter result as letter "A" thereby fetching the AJK.grammar. Because the bus stop name "King's Cross" is still in the AJK.grammar, the system could correctly recognise it even though it had incorrectly recognised the first letter. See Figure 5.5.

***System**: Please say the first letter of your origin.*

User: *K*

System: Recognised as "A". System fetches the grammar AJK.grammar.

***System**: Please say your origin.*

User: *King's Cross*

System: Recognised as "King's Cross"

……

Figure 5.5 - Using Combined First Letter Grammar

Combining the confusable letters into a set of medium sized grammars, allows the system to theoretically fetch the correct grammar 100% of the time. However, this is a trade off between the recognition accuracy and system processing time. For example, when the speech recogniser returns the result as a letter "D", the system has to fetch the BCDEGPTV.grammar file which contains all bus stop names starting with letters "B", "C", "D", "E", "G", "P", "T" or "V". This will be a large grammar file. The experimental results show that the system has to take 9 seconds to recognise one bus stop name using this grammar. This is an improvement over the 13 seconds it takes to process the single large grammar file but the user would still perceive such a delay. The same problem happens when the system produces a recognition result of the letter "B" (BDEVP.grammar), the letter "E" (EBCD.grammar) and the letter "N" (NMFZ.grammar). The system could, therefore, use the above large grammar files to process the user's input, but it would not then be a usable real-time system.

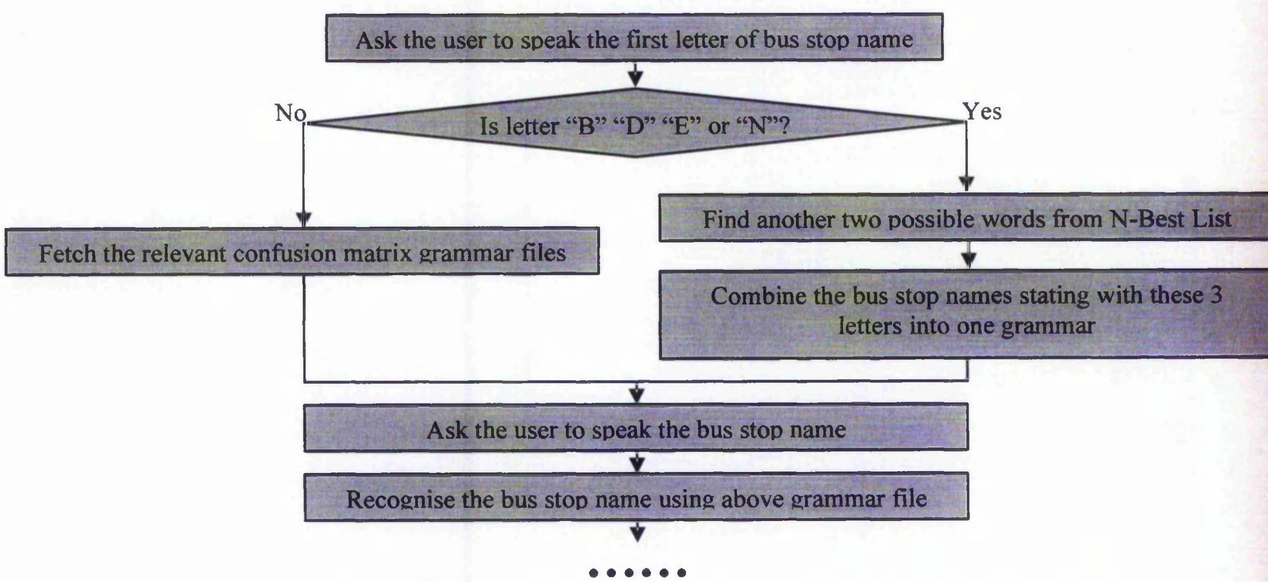


Figure 5.6 - The Design Diagram of Combining Grammars Using the N-Best First Letter List

The N-Best method is another possible way to handle this difficulty. If the system returns "D", "B" "E", or "N" as the top matching letter, the system can get the best 3 recognition results from the recognition list and combine the bus stop name grammars starting with these 3 letters into one grammar file. For example, if the system recognises the input as "D", the system will extract the next two recognition results from the N-Best list. If these are "B" and "E", the system could combine the bus stop names starting with the letters "D", "B" and "E" into one grammar file which it then uses for bus stop name recognition (see Figure 5.6). Dynamically combining the bus stop names starting with the best 3 letters into one grammar file in this manner can reduce the grammar file size. Unfortunately, the problem with this approach is that the system has to spend about 30 seconds combining this grammar file. It is, however, possible to combine all possible 3 letter grammar files offline, so that the system just has to fetch the appropriate grammar files when the system needs them. After changing from dynamic grammar file combination to static offline grammar file combination, the system processing time was found to be significantly improved. The experimental results can be seen in Section 5.2.3.

5.2.3 Comparison between Large and First Letter Grammar Systems Experiment

A system using many small grammar files has been developed in this project. In order to compare its performances against the large grammar file system, an experiment was carried out.

5.2.3.1 Experimental Design and Materials

In this experiment, the system using the large grammar file and another using many small grammar files are used to carry out the same task finding journey information. At the beginning of the experiment the users were informed as to the purpose of the experiment which was testing the interface's speech recognition accuracy. For each system, the users were taught how to interact with the system. In the experiment, the users were required to find one bus information enquiry using both the system with a large grammar file and the system with many small grammar files using the confusion plus n-best static offline grammar file combination approach.

Both of the systems utilise the Nuance V7.0.4 speech recogniser. The confidence level of this speech recogniser was configured to 0.1. This is the minimum value of confidence level which means the system will accept all users' input and rarely give rejection errors. The Timeout property was set to 5 seconds; the Complete Time Out property was set to 1 second.

5.2.3.2 Participant Selection

There are 10 native and 10 non-native United Kingdom English speakers participated in the experiment. The users ranged in age from 18 to 70 years of age.

5.2.3.3 Results and Discussion

System	Large Grammar System	First Letter Grammar System
Processing Time	13 Seconds	1.67-2.17 Seconds
Accuracy Rate	53%±4.9%	53%±4.9%

Table 5.7 - The Experimental Results for the Large Grammar and First Letter Grammar Systems

Table 5.7 shows that the use of small grammar files has significantly reduced the system processing time for a single entry. The processing time is reduced from 13 seconds with the large grammar file to 1.67-2.17 seconds with different small grammar files. The processing times variation for the small grammar file system is due to the different small grammar file sizes. For example, the "Z" grammar only contains 15 names. When the user inputs the bus stop names starting with the letter "Z", the system only takes 1.67 seconds to process the input. Letters "B", "C" and "D", on the other hand, use the combined grammar file containing most names (4220). When the system uses the "BCD" grammar file to recognise the bus stop names, the system takes 2.17 seconds to process it. However, even with the longest delay, the users cannot perceive the latencies associated with these delays.

Our measurement of recognition accuracy depended on whether the users' inputs were exactly recognised; out of grammar errors were counted as an error. For example, in the small grammar file, if the user wants to travel from "Oxford Street", the system will ask the user to speak the first letter of their origin. If the user says "O" and the system returns "A", so the system will fetch the AJK grammar file.

When the user then says “Oxford Street”, the system will definitely return one wrong substitution result because the system has fetched an incorrect grammar file (AJK grammar file does not contain the “Oxford Street” bus stop name). In this case, a substitution error has occurred and is counted as a misrecognition error. Conversely, if the user said “O” and the engine returned “L”, the LHO grammar file would be fetched because letters L, H and O share the same confusing letter matrix. When the user then says “Oxford Street” the desired end recognition result could still be achieved. This would be counted as a correct overall recognition result.

As stated earlier, the mean sample accuracy rate of the large grammar system is 53% (μ=53%±4.9%). This is significantly worse than the Nottingham directed-dialogue interface’s average sample accuracy rate of 88.5% (μ=88.5%±5.8%) which uses a medium size grammar file (1355 bus stop names). Ideally, the London system using several smaller grammar files should be more accurate than the London system using the large grammar file. Unfortunately, the experimental results show that both London systems give the same accuracy rate. The major reason is that many more out-of-grammar errors occurred in the system using the small grammar files. Evaluating the results it can be seen that the system incorrectly recognised the first letter 17 times during the 100 users inputs. This number does not include the incorrectly recognised letters that resulted in fetching the correct small grammar files as a result of using the confusing letter matrix for the reasons stated above. The 17 times the system incorrectly recognised the first letter from the user’s inputs, such that the incorrect small grammar file was fetched, directly reduced the chances of the system correctly recognising the full bus stop names. 3 of these times were caused by the user’s unclear utterance and 2 times were caused by unexpected noisy interruption. Another 12 times were caused by an incomplete confusing letter matrix. For example, when the user said “A”, the engine returned “H” (which is not part of the confusing letter matrix “AJK”). As a result the “AJK” grammar file would not be fetched.

It is possible that the out of grammar file errors could be reduced by producing a more representative confusing matrix. However, although recognising the first letter has significantly changed the system's performance in terms of processing time, it has introduced significant usability issues. In human to human communication, asking people to speak the first letter of a word and then trying to recognise a word is not a natural form of interaction. In addition to this inherent unnatural interaction format, the accuracy rate of the system, as it stands, is still not accurate enough to be used by the public. In order to achieve the goal of naturalness and robustness, the system had to be improved further.

5.3 Recognising the First Phoneme

To allow the user to communicate effectively with the system, it would be better for the user not to be asked any unnecessary questions (i.e. "Please say the first letter of your origin" etc). In the first letter system, the system asks the users to speak the first letter of the bus stop names before asking the bus stop name itself. This is an unnatural communication format from a usability stand point. In actual fact, it should be possible for the system to get this information automatically from the user's input by recording the speech input. If the system were to ask the user to speak the bus stop names once and record the user's input, the system could retrieve the first letter from the recorded audio. This method would help reduce the "unnatural" communication. To illustrate how the system can be given these functions, it is necessary to first explain about Recording Audio in VoiceXML, Audio File Format and Phoneme extraction because of recording the user's utterance will be the first step for speech recognition.

5.3.1 Recording User's Input in VoiceXML

In VoiceXML [W3C 2004], the <record> element is an input item that collects a recording from the user. A reference to the recorded audio is stored in the input item variable, which can be played back (using the expr attribute on <audio>) or submitted to a server, The user is prompted to input a message, and the system then records it. The recording terminates when one of the following conditions is met: the interval of final silence occurs, a DTMF key is pressed, the maximum recording time is exceeded, or the caller hangs up. See Figure 5.7.

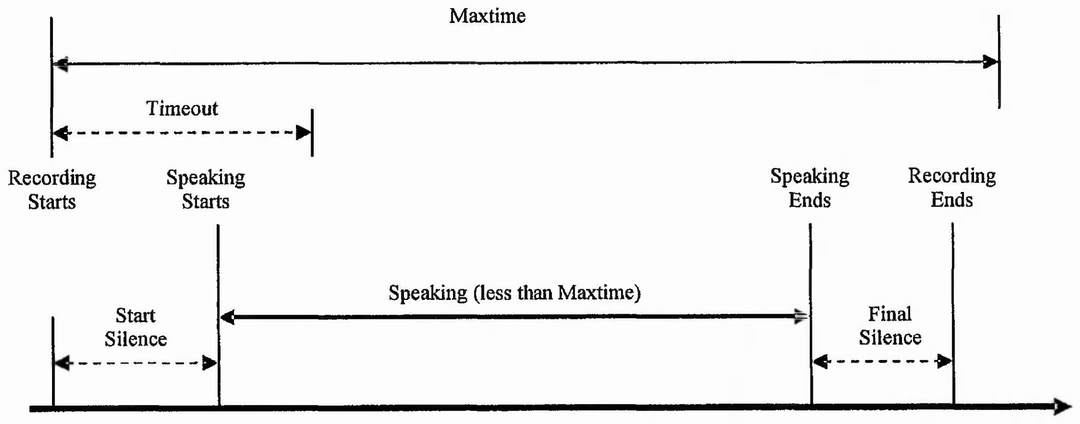


Figure 5.7 - Timing of Audio Recording

A recording normally begins after the playback of any prompts (including the 'beep' tone if defined). As an optimisation, a platform may begin recording when the user starts speaking. A timeout interval is defined to begin immediately after the prompt playback (including the 'beep' tone if defined) and its duration is determined by the 'timeout' property. If the timeout interval is exceeded before recording begins, then a <noinput> event is thrown.

A maxtime interval is defined to begin when recording starts and its duration is determined by a 'maxtime' attribute. If the maxtime interval is exceeded before recording ends, then the recording is terminated and the maxtime shadow variable is set to 'true'.

A recording ends when an event is thrown, DTMF or speech input matches an active grammar, or the maxtime interval is exceeded. As an optimisation, a platform may end recording after a silence interval (set by the 'finalsilence' attribute) indicating the user has stopped speaking. If no audio is collected during execution of <record>, then the record variable remains unfilled. This can occur, for example, when DTMF or speech input is received during prompt playback or the timeout interval expires before an input is received.

VoiceXML support for recognition of speech grammars during recording is optional. If the VoiceXML platform supports simultaneous recognition and recording, then spoken input matching an active grammar can terminate recording. The 'terminating' speech input is accessible via application.lastresult$ and the item's utterance and confidence shadow variables. The audio of the recognised 'terminating' speech input is not available and is not part of the recording. If the termination grammar matched (DTMF or speech) is a local grammar, the recording is placed in the record variable. Otherwise, the record variable is left unfilled and the form interpretation algorithm is invoked. In each case, application.lastresult$ and the item's shadow variables are assigned.

5.3.2 Phonemes

Vowels	Diphthongs	Semivowels	Fricatives	Nasals	Plosives	Affricates
heed	bay	was	sail	am	bat	jaw
hid	by	ran	ship	an	disc	chore
head	bow	lot	funnel	sang	goat	
had	bough	yacht	Thick		pool	
hard	beer		Hull		tap	
hod	doer		zoo		kite	
hoard	boar		azure			
hood	boy		that			
Who'd	bear		valve			
hut						
heard						
the						

Table 5.8 - Phoneme Categories of English and Examples of Words

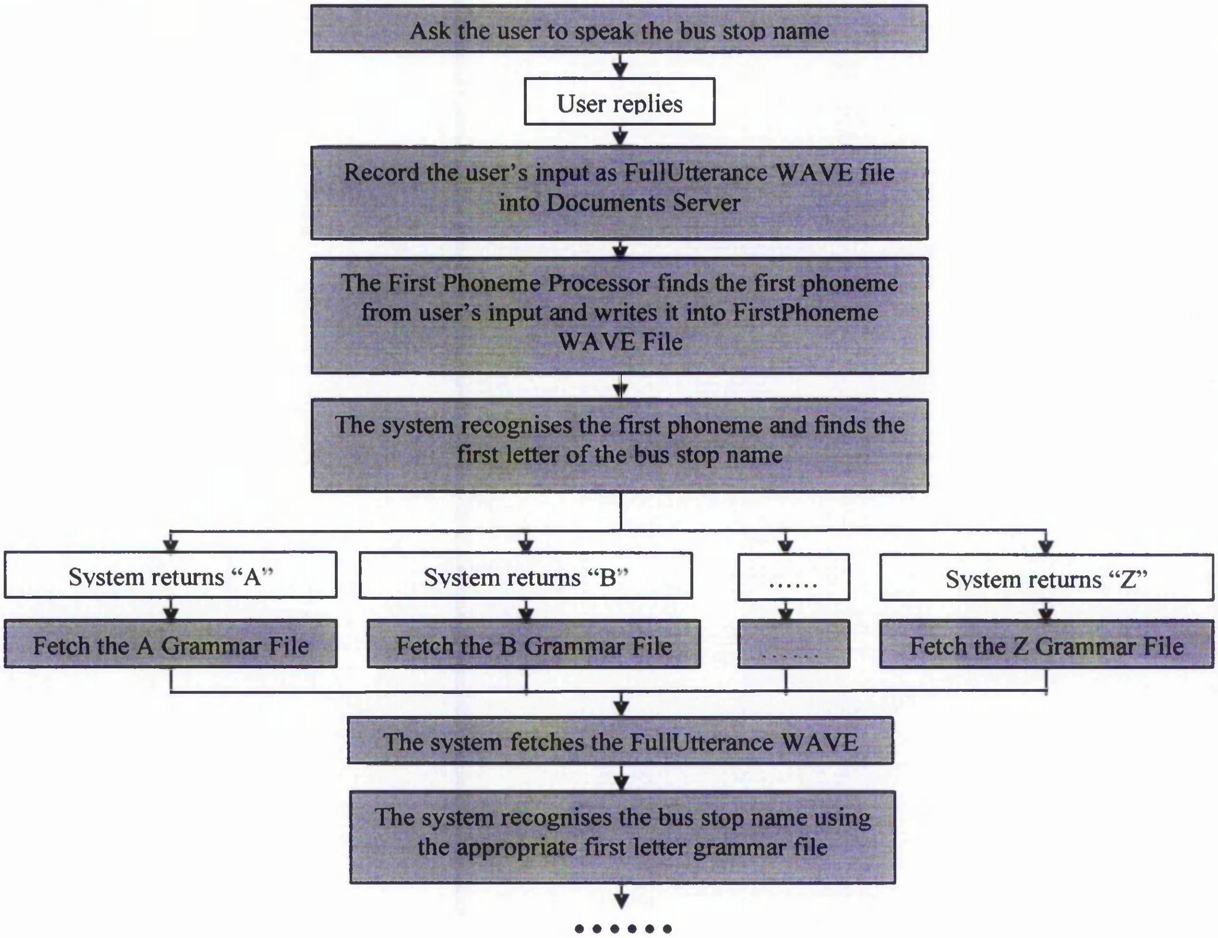
If the system wants to recognise automatically the first letter of a spoken word, the system has to detect the first phoneme of this word. A phoneme is the smallest segment of sound that can distinguish two words. If you change a phoneme within a word, you get a different word. (i.e. /kat/ and /mat/) English has around 45 phonemes which can be categorised [Batt 1999] and Table 5.8 lists all of them.

Phonemes are abstract units - a useful way of describing speech. They provide a categorical description of speech sounds. In real speech, there are no hard boundaries between the categories: researchers use context to disambiguate phonemes. Each phoneme can be realised in different ways depending on the context and speaker: i.e. allophones. The units in this case are known as phones. There are many more phones than phonemes, as some of them are produced in different ways depending on context. For example, the pronunciation of the phoneme /l/ differs slightly when it occurs before consonants and at the end of utterances (as in “people”), as opposed to other positions (e.g. in “politics”). The two phones are called the velarized and the non-velarized “l” respectively. As they are both different forms of the same phoneme, they form a set of allophones.

It is not just the speech organs involved that influence the way an utterance is spoken and subsequently interpreted. The stress is one of its prosodic features. Stress is used at two levels; in sentences it indicates the most important words, while in words it indicates the prominent syllables. For example, the word “object” could be interpreted as either a noun or a verb, depending on whether the stress is placed on the first or second syllable. Rhythm refers to the timing aspect of utterances. English is said to be stress-timed, with approximately equal time intervals between stresses (experiments have shown that, objectively, there is merely a tendency in this direction). The portion of an utterance beginning with one stressed syllable and ending with another is called a foot (by analogy with poetry). So, a four-syllable foot (1 stressed, 3 unstressed) would be longer than a single (stressed) syllable foot, but not four times longer.

5.3.3 Automatically Recognising the First Phoneme

An improved first letter system has been developed that uses on automatic first phoneme recognition method. Figure 5.8 shows the design diagram for a speech interface capable of recognising automatically the first phoneme method.



Firgue 5.8 - The Design Diagram for the Automatic First Letter Recognition Based London System

Figure 5.9 shows the architecture of the system for retrieving automatically the first phoneme from the user's utterance.

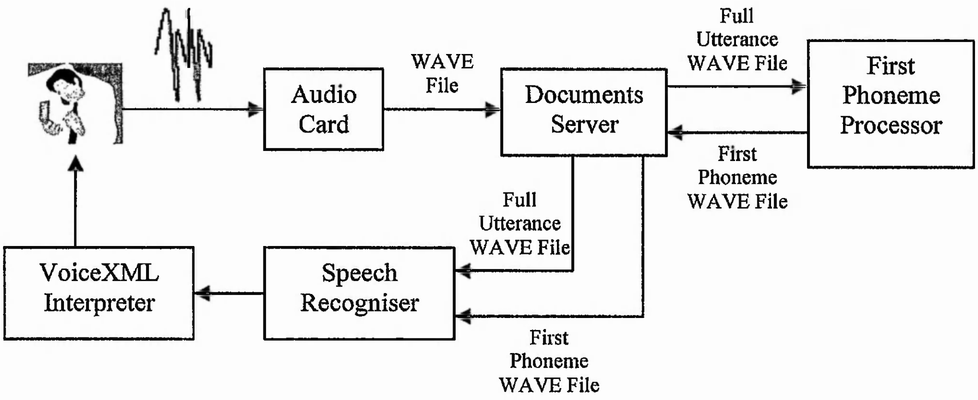


Figure 5.9 - The System Retrieving Automatically First Phoneme Architecture

In this system, the users are just asked to give the bus stop name. The sound card converts the analog signal from the telephone to a digital signal through the A/D converter. The A/D converter records the value of the electrical voltage at specific intervals and converts this into binary data. There are two important factors during this process. First is the "sample rate", or how often to record the voltage values. Second, is the "bits per sample", or how accurately the value is recorded. A third item is the number of channels (mono or stereo), but for most ASR applications mono is sufficient. The analog signal is sampled in 8 KHz 8-bit Mono format and converted into digital form using a technique called Pulse Code Modulation or PCM in this system. The <record> element stores the user's input as a WAVE (RIFF header) format on a document server.

Because speech is relatively low bandwidth (mostly between 100Hz-4kHz), 8000 samples/sec (4kHz) is sufficient for most ASR. 8 bits per sample means that the recorded voltage amplitudes will be given binary values between 0 and 255. The <record> element stores the user's input as WAVE (RIFF Header) format on the Documents Server.

The WAVE file format [Microsoft 2005B] is a subset of Microsoft's RIFF specification for the storage of multimedia files. A RIFF file starts out with a file header followed by a sequence of data chunks. A WAVE file is often just a RIFF file with a single "WAVE" chunk which consists of two sub-chunks -- a "fmt" chunk specifying the data format and a "data" chunk containing the actual sample data. This is called the "Canonical form". 8-bit samples are stored as unsigned bytes, ranging from 0 to 255. See Appendix 2 for the file format.

A First Phoneme Processor (FPP) was developed for extracting automatically the first phoneme from recorded bus stop name inputs. The FPP is a Java component which is based on Kazantsev's theory. [Kazantsev 2004] described a method of phoneme distinction, which is used in continuous speech recognition systems. This method finds phoneme positions in the input sound flow and is based on using correlation functions of the sound power spectrum. Its application reduces the amount of calculations for the recognition unit in a continuous speech recognition system.

The method of phoneme distinction uses the properties of phonemes to find "suspicious" sound fragments that seem like a phoneme in a sound flow. The phonemes of the human voice have certain properties. [Kazantsev 2004] states that a phonemes' length is not less than 0.04 second and not more than 0.35 second. As sound is sampled in 8 KHz 8-bit Mono, in this system, this means that 1 second's worth of sound should occupy 8000 bytes in the Data field of a WAVE file. Therefore a phoneme should take at least 320 (8000*0.04) Bytes and not more than 2800 (8000*0.35) Bytes in the Data field of a Wave file. In addition, by analysing the wave files produced, it was found that silence is represented as values 70-7F and F0-FF.

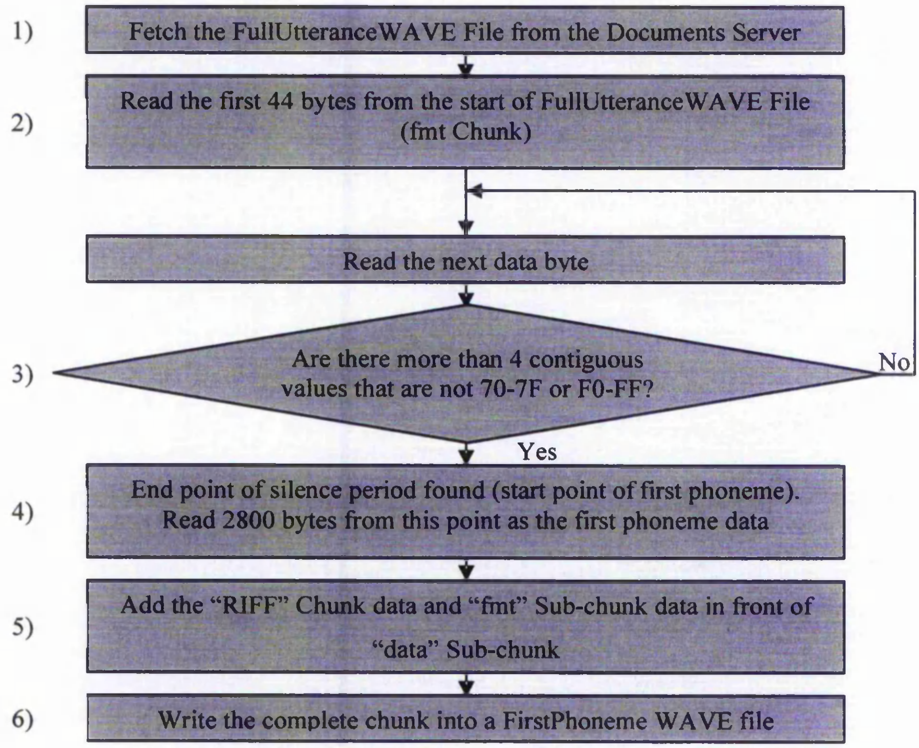


Figure 5.10 - The First Phoneme Process Algorithm

The First Phoneme Process (FPP) works as follows (Figure 5.10 shows the FPP algorithm):

1) The FPP fetches the user's FullUtterance WAVE file from the Documents Server.
2) The FPP finds the end of the "RIFF" Chunk data and "fmt" Sub-chunk data. (The first 44 bytes of FullUtterance WAVE file are "RIFF" Chunk data and "fmt" Sub-chunk data - see Appendix 2)
3) The FPP reads the data from the 45th byte of FullUtterance and compares these data values with silence values 70-7F or F0-FF. After the system has found 4 contiguous values which are not silence value 70-7F or F0-FF, the system has reached the end of the Silence period (the interval between the system starting recording to the user starting to give the response) and found the start of the first phoneme.

4) The FPP counts 2800 Bytes from the end of the Silence period and copies these as the "data" Sub-chunk of a first phoneme RIFF file;
5) The FPP adds the "RIFF" Chunk data and "fmt" Sub-chunk data in front of "data" Sub-chunk
6) The FPP writes the complete chunk into a FirstPhoneme WAVE file.

The FirstPhoneme WAVE file is then passed to the speech recogniser. Initially, the speech recogniser used a grammar file which contained the letters "A" to "Z" in order to recognise the first phonemes. Unfortunately, the results from using this grammar do not give a good recognition performance. Using another grammar file which contained all 45 English phonemes was also tried. Ideally, each incoming frequency band (from the FirstPhoneme WAVE file) would find the right phoneme in the grammar file. However, that also gave poor results.

In order to get the performances of recognizing automatically the first letter, an experiment was carried out.

5.3.3.1 Experimental Design and Materials

In this experiment, the system using the grammar file which contained all 45 English phonemes is used to recognize automatically the first letter of a bus stop name. At the beginning of the experiment the users were informed as to the purpose of the experiment which was testing the interface's speech recognition accuracy of first letter. The users were taught how to interact with the system. In the experiment, the users were required to speak 5 bus stop names to the system.

Both of the systems utilise the Nuance V7.0.4 speech recogniser. The confidence level of this speech recogniser was configured to 0.5. This is the confidence threshold required for the speech-recognition engine to decide whether the first phoneme of input speech matches a phoneme from the grammar. The Timeout property was set to 5 seconds; the Complete Time Out property was set to 1 second.

5.3.3.2 Participant Selection

There are 10 native and 10 non-native United Kingdom English speakers participated in the experiment. The users ranged in age from 18 to 75 years of age.

5.3.3.3 Results and Discussion

Using the automatic FPP to isolate the first letter and a grammar file containing all 45 phonemes, the system incorrectly recognised the first letter up to 74 times during the 100 users' inputs. There are so many variations in sound due to how words are spoken that it's very difficult to reliable match an incoming sound to an entry in the grammar file. Different people pronounce the same phoneme differently. In addition, in this system, the speech recogniser attempts to recognise the first phoneme from the FirstPhoneme WAVE file which uses a fixed 0.35 second sampling period. This duration is long for some phonemes. During the interval in which the speech recogniser is trying to recognise the first phoneme, many phoneme frequency bands could actually be present with the result that the speech recogniser only recognises the most outstanding (stress pronounced) phoneme. To make matters worse, the environment also adds its own share of noise. The above difficulties thus cause the system to give a very poor recognition performance.

To improve the performance of this system, the First Phoneme Processor would have to be able to accurately judge when a phoneme ends and the next one begins. For this, it is possible to use the Hidden Markov Models (HMM) technique [Weber 2003]. This is a mathematical model that uses statistics, to figure out when speech starts and stops. The HMM based speech recogniser uses complex techniques to approximate the incoming sound and figure out which phonemes are being used. Another way of identifying phonemes would be to "train" the speech recognition software. In training, many variations of the same phoneme are given, and the software analyses each of these through statistical methods. Although there has been much success using these methodologies, the approaches do not explicitly incorporate knowledge of important aspects of human speech production. One disadvantage of such methodologies is the large state space and, therefore, large number of parameters that need to be optimzed [Richardson 2000]. In addition, the processing time overhead introduced when using HMM for phoneme segmentation is significant [Weber 2003]. Thus, the obvious strategy of using HMM to improve the FPP accuracy is not practical in any real-time application. However, a novel system using a Last Word Processor (LWP) will be introduced in Section 5.4, which consumes less time and gives a reasonable improvement in the recognition accuracy rate.

5.4 Recognising the Last Word

In Section 5.3, a system that tried to record and extract phonemes from the spoken input was described. Unfortunately, the first phoneme recognition results were not encouraging. There are two probable reasons for this: i) it is inherently harder to disambiguate the small sized phonemes WAVE segments, ii) the first letter grammar file the system is using is not optimal. Whilst analysing this problem, the idea of dividing the large London grammar on the basis of end word (i.e. Road, Street, grove etc.) inspired the development of another system to improve the latency and accuracy of the large grammar system. A system that could recognise automatically the last word would probably work better than one based on first phoneme because the sound segments are longer and the words sounds are more distinct. Development of a last word segmentation process would mean that the system could sub-divide the large London grammar file into many smaller grammar files based on the different word endings. For example, when the system asks the user to say the original/destination bus stop names, it can record the user's input as before. The system then can try to locate and recognise the last word of the bus stop names. Using this last word recognition, the system can attempt to recognise the complete bus stop name using the appropriate small ending grammar file. Based on the first letter experiment result, this method should reduce the speech recogniser's processing time because this system also uses many small grammar files. The design diagram of this system is given in Figure 5.11.

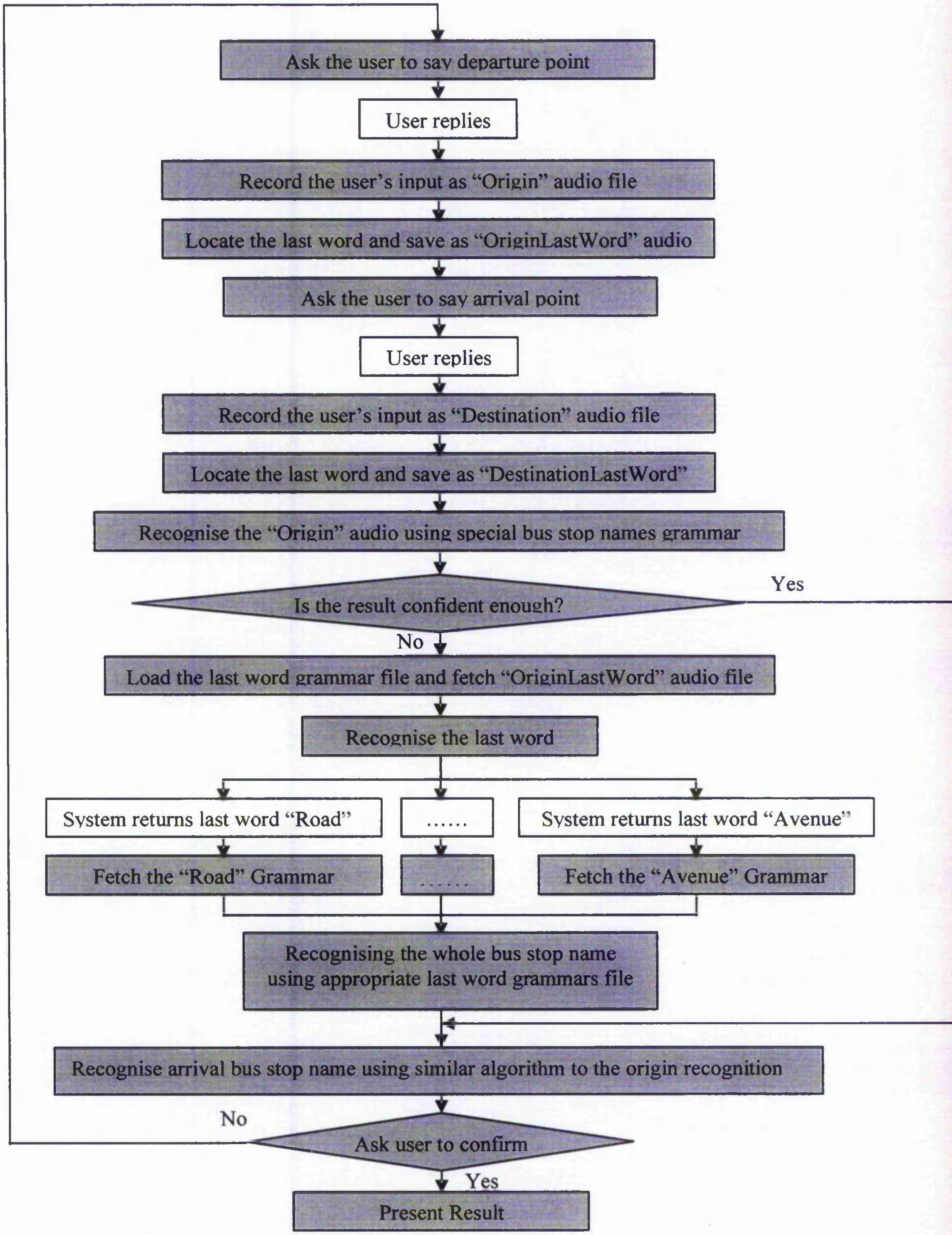


Figure 5.11 - The Design Diagram for the Last Word Based London Interface

From the 27792 bus stop names about 633 common endings can be identified. However most of these endings are only common to one or two bus stop names.

There are 61 different endings that are common to 10 or more bus stop names. The system puts any bus stop names with last words that do not belong to this 61 endings list, into one special grammar file (special.grammar). The large grammar file can thus be divided into 62 smaller grammar files based on the different bus stop endings. See Table 5.9.

road	mews	close	drive	gardens	grove	lane
street	parade	terrace	view	estate	park	place
walk	avenue	crescent	way	villas	buildings	row
square	hill	gate	south	cottages	yard	court
bank	corner	vale	green	chambers	arches	passage
circus	end	rise	croft	market	quay	east
north	west	path	almshouses	mead	village	approach
broadway	wharf	arcade	flats	mount	side	dene
mall	fields	common	bridge	studios	**Special**	

Table 5.9 - Different Bus Stop Endings Grammar Files

5.4.1 The System Design

The architecture and overall operation of this system is similar to the first phoneme recognition system. The difference is that this system locates and recognises the last word of the user's input rather than the first phoneme. Firstly, the users are asked to give the bus stop names. Their responses are passed to the sound card in the system, sampled in 8 KHz 8-bit Mono and converted into digital form using PCM. The <record> element stores the user's input as WAVE (RIFF Header) format in the Documents Server. See figure 5.12.

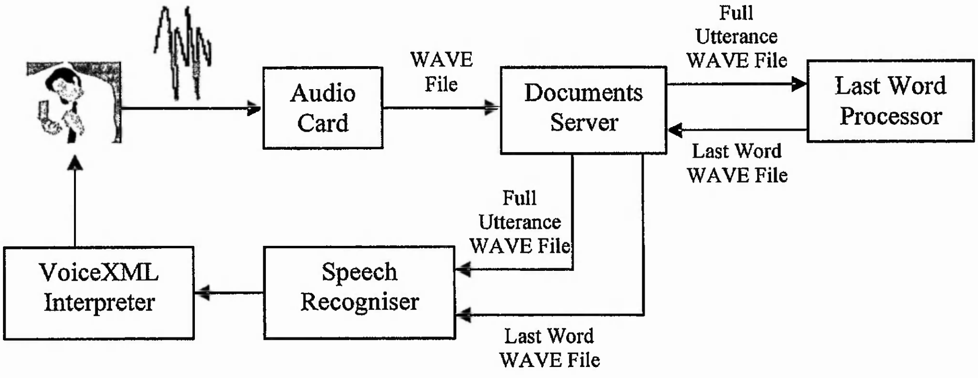


Figure 5.12 - The System Retrieving Automatically Last Word Architecture

The major issue for this system is how to locate the position where one word ends and the next one begins. Natural human speech often contains occasional pauses even in the middle of a word, thereby causes incorrect word recognition. In this system, a Last Word Processor was developed to locate automatically the last word of a user's input. LWP works similar to the First Phoneme Processor. The core issue for the Last Word Processor is how to determine the boundary between words. By analysing an average-speed speaker sound file, it again was found that silence is represented as value 70-7F and F0-FF and that noise is represented as value 60-6F and E0-EF. Thus if there are more than 4 contiguous values of either 60-7F or E0-FF between data samples of other byte values, these values can be assumed to be the boundary between words. Figure 5.13 shows the LWP algorithm.

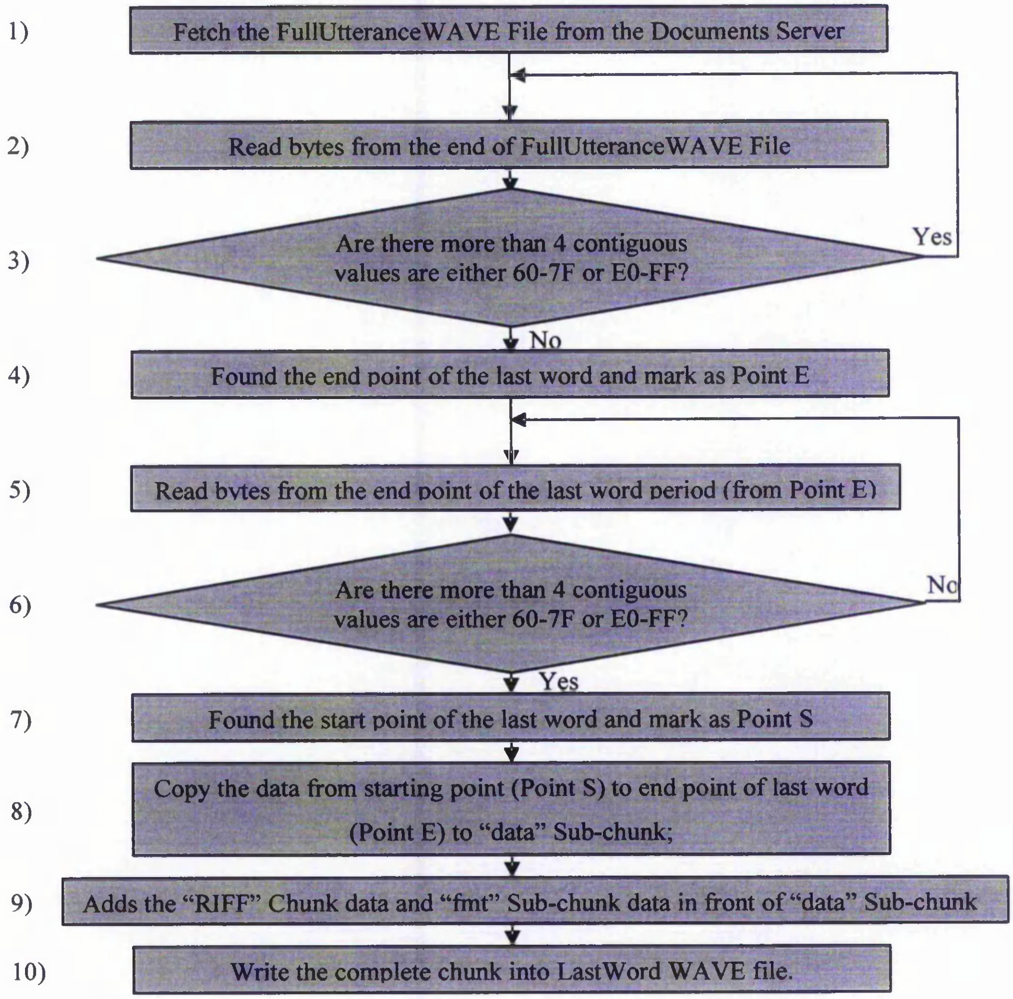


Figure 5.13 - Last Word Location Algorithm

The Last Word Processor (LWP) works as follows:

1) The LWP fetches the user's FullUtterance WAVE file from the Documents Server.
2) The LWP reads the data from the last byte of FullUtterance.
3) The LWP compares these data values against the silence and noise values (60-7F and E0-FF). After the system has found 4 contiguous values which are not silence and noise value, the system have reached the end point of the last word period (Point E).
4) The LWP has found the end point of last word period and remembers this point as Point E.
5) The LWP reads the data from the end point of the last word period (Point E).
6) The LWP compares these data values against the silence and noise values (60-7F and E0-FF). After the system has found 4 contiguous values which are silence and noise value, the system have reached the starting point of the last word period (Point S).
7) The LWP has found the starting point of last word period and remembers this point as Point S.
8) The LWP copies the data from the last word starting point (Point S) to the last word end point (Point E) as a "data" Sub-chunk.
9) The LWP adds the "RIFF" Chunk data and "fmt" Sub-chunk data in front of "data" Sub-chunk.
10) The LWP writes the complete chunk into LastWord WAVE file.

The Last Word Processor can correctly segment most of the last words from the users' inputs. However, some words are still difficult for the LWP to distinguish. For example, the words "Street" and "Approach". The pronunciation of "Street" [stri:t] contains three phonemes: [s], [tri:] and [t]. The data values of the first phoneme [s] are similar to noise data values and lasts for more than four bytes. In this case, the LWP will only write the data values of the phonemes [tri:] [t] into the LastWord WAVE File (sounds like "treet") and regards the phoneme [s] as noise. The speech recogniser will then have difficulty recognising the word using the grammar file because it does not contain the word "treet". The pronunciation of "Approach" [ə`prəutʃ] contains four phonemes: [ə], [p], [rəu] and [tʃ]. When a human speaker pronounces a word like "Approach", they put special emphasis on the second phoneme [p]. Before such an emphasised phoneme, humans always have a period of silence between the initial vowel [ə] and the emphasised phoneme. The length of this silence is often longer than four bytes causing similar problems as the word "Street". The LWP only writes the data of phonemes [p] [rəu] and [tʃ] into LastWord WAVE File (sounds like "Proach") and regards the phoneme [ə] as another word. In order to address these difficulties, the words "Street" and "Approach" are replaced in the grammar file with the words "treat" and "proach" etc.

5.4.2 The Last Word Processor Experiment

A system using small grammar files that are based on the different bus stop name endings has been developed. In order to compare its performance against the large grammar file system and the system using the small grammar files based on first letter, an experiment was carried out.

5.4.2.1 Experimental Design and Materials

In order to make the same comparison environment, this system was used to carry out the same task as the experiment conducted in Section 5.2.3. In this experiment, the system using the small grammar files that are based on the different bus stop names to carry out the same task finding journey information. At the beginning of the experiment the users were informed as to the purpose of the experiment which was testing the interface's speech recognition accuracy. For each system, the users were taught how to interact with the system. In the experiment, the users were required to find one bus information enquiry.

The confidence level of this speech recogniser was configured to 0.1. This is the minimum value of confidence level which means the system will accept all users' input and rarely give rejection errors. The Timeout property was set to 5 seconds; the Complete Time Out property was set to 1 second.

5.4.2.2 Participant Selection

There are 25 native and 25 non-native United Kingdom English speakers participated in the experiment. The users ranged in age from 18 to 70 years of age.

5.4.2.2 Results and Discussion

The experiment results show that the last word based system takes a total of 1.37-1.89 seconds to record, submit and locate the last words in the user's response. The variation in processing time is mainly due to the variation in input time of the different length utterances. The major user perceived latency is now 4.01-5.94 seconds, this includes both origin and destination recognition (See Figure 5.14 and Figure 5.15). These times are dependent on the size of small grammar files used and whether or not the user spoke a bus stop names with one of the special last words. For example, the "Corner" grammar only contains 14 names. When the user inputs bus stop names that end with "Corner", the system takes less time to recognise the bus stop name using the "Corner" grammar file. The "Road" grammar, on the other hand, contains the most names (6487). When a user inputs bus stop names that end with the word "Road", the system takes the most time to recognise it.

In the event the user spoke a bus stop name with one of the special last words, the system does not need to fetch or recognise the last word and instead will recognise the full bus stop name at the special grammar stage. This takes the least amount of time (see Figure 5.9). However, even with the longest delay 5.94 seconds, the user still can accept this latency because they have been told and are prepared for the wait.

***System**: Please say your origin.*

***User:** King's Cross*

*(System takes a total of **1.37-1.89** seconds to record, submit and locate the last words in the user's response. The system does **not** recognise the origin or last word at this stage.)*

***System**: Please say your destination.*

***User:** Bailey Mews*

*(System takes a total of **1.37-1.89** seconds to record, submit and locate the last words in the user's response. The system does **not** recognise the destination or last word at this stage.)*

***System:** please wait for a moment.*

*(The system recognises both the origin and destination using the algorithm shown in figure 5.9. Users will perceive **4.01-5.94** seconds delay.)*

***System:** Do you want to go from King's Cross to Bailey Mews?*

***User**: Yes*

***System**: Ok, you can catch bus......*

Figure 5.14 - The User Perceived Latency in Last Word Recognition System

Because the last word recognition system does not need to ask any additional questions of the user, the users can communicate naturally with the system. The first letter recognition system has to ask users to give the first letter of their origin or destination which is not only unnatural but also wastes time. The experiments show that using the last word recognition system, the average users spent less time (34.7 seconds) accomplishing the task* than using the first letter recognition system (44.8 seconds) or the one large grammar system (77.8 seconds). From a usability point, the users could therefore accomplish the task more naturally and more efficiently using the last word recognition system.

* This task completion time is different to the Average Completion Time discussed in Chapter 3 and 4 because it does not include the time spent retrieving or listening to the DIME enquiry result.

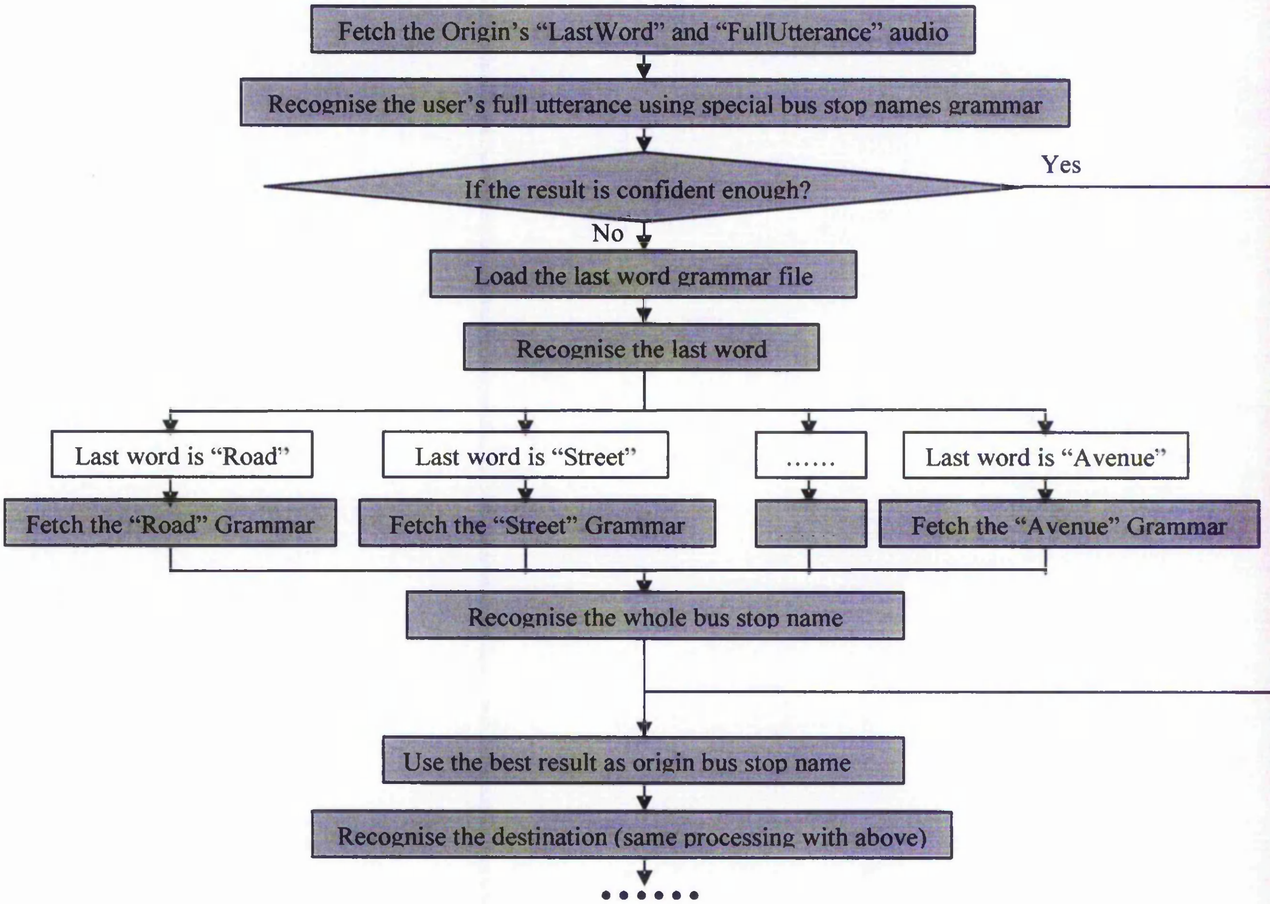


Figure 5.15 - The Design Diagram for the LWP Bus Stop Name Recognition

To evaluate the speech recognition results the standard deviation with student t distribution method was used as described in 3.3.3. Overall there was a mean sample accuracy rate of 61% (μ=61%±4.8%) in the last word recognition system. This is an improvement of 8% on the accuracy rate of both the first letter recognition system (53%+4.9%) and the large grammar system (53%+4.9%). [Peissner 2002] states that a 5% improvement in accuracy is much more effective at a low starting level than at a rate of 90% correct recognition. Thus the last word method is seen to be a significant improvement on the other two techniques.

39 bus stop names were incorrectly recognised in the 100 tests. Among these 39 errors, 29 last word recognition errors were caused by the Last Word Processor segmentation algorithm. Among the LWP's 29 errors, 23 times were the results of

the LWP failing to accurately find the last word in the FullUtterance WAVE file because of the user's very quick speaking rate. When users speak very quickly, the inter-word segments are very small. If the segments values are less than four bytes, the LWP will not separate the words in the user's inputs.

The experiment results also show that 22 of the 39 errors happened in noisy environments. Sudden fluctuations in noise decrease the LWP's performance by causing the algorithm to find false beginnings or ends of the last words. The algorithm in LWP only considers stationary noise. If sudden noise happens whilst the user is speaking, the algorithm will fail to segment the words correctly. The signal amplitude, or more appropriately, the ratio of the signal to noise amplitude, determines the segmentation and therefore recognition accuracy of the system. For any noise level, if the speech signal level is equal to the noise level, the last word segmentation will be relatively poor.

5.5 Conclusion

Based on the successful results of the Nottingham travel information speech-enabled system, a directed-dialogue speech interface has been developed for London to investigate the performance of a speech-enabled interface with very large grammars. A directed-dialogue was chosen rather than a mixed-dialogue because there are 27792 bus stop names in London area. This increase in the number of bus stop names was expected to produce an increase in processing time and a reduction in recognition accuracy. Using a directed-dialogue would help minimise both these critical performance parameters. The initial version of this system used a large grammar file containing all of the bus names. Experimental results show that the system processing time has indeed increased, taking up to 13 seconds to process one bus stop name. During this system processing time, the users do not hear any system prompt. This latency is unacceptable to the users. The experimental results also show that the recognition performance of this interface is significantly lower (sample accuracy rate of 53%) than the equivalent Nottingham system. The system processing time had, therefore, to be reduced and the speech recognition accuracy rate improved for public acceptance. To overcome these critical difficulties, it was necessary to reduce the grammar size.

A second version interface was developed that used many small grammar files that contain bus stop names grouped according to their starting letter. To facilitate this grammar file separation, this system has to ask the user to speak the first letter of their origin or destination before speaking the full bus stop name. After the system has recognised the first letter of the bus stop names, the system attempted to recognise the full bus stop names using the small grammar file that corresponded to the recognised first letter. Experimental results show that recognising the first letter does significantly improve the system's performance in terms of the processing time; which now only takes 1.67-2.17 seconds to process one user's entry. However, this methodology does leave usability issues. In human to human communication, asking a person to speak the first letter of a word and then trying to recognise this word is not a natural form of interaction. The accuracy rate of this system (sample accuracy rate of 53%) is also still not accurate enough to be used by the public. In order to achieve the goal of naturalness and robustness, the system had to be further improved.

To ensure that the user can naturally communicate with the system, the user should not be asked to answer any 'unnecessary' (from the user's perspective) questions. Ideally, the system should get this data from user's input. If the system just asks the user to speak the bus stop names once and records the user's input, the system can retrieve automatically the first letter from the recorded audio. It was hypothesised that this method would help reduce the excrescent communication. Consequently, a First Phoneme Processor was developed which, in theory, should be able to find automatically the first phoneme from a user's input. Unfortunately, the first phoneme recognition results were not encouraging. This is probably due to the complexity of segmenting the first phoneme from the recorded speech image. The system incorrectly recognised the first letter up to 74 times during the 100 user inputs in the experiment. Whilst discussing this problem, the idea of dividing the large London grammar on the basis of end words was proposed. Segmentation of the speech image into separate words should be easier than segmentation based on first phoneme because the sound segments are longer and the words sounds are more distinct. Using this methodology it was possible for the system to sub-divide the large London grammar files into many smaller grammar files based on the different word endings (street, road, avenue etc). The experimental results show that the LWP system takes a total of 1.37-1.89 seconds to record, and segment the user inputs. The major user perceived latency is now 4.01-5.94 seconds. This is when the system is attempting to recognise both the origin and destination bus stop names using the LWP processing algorithm. Because the last word recognition system does not need to ask any unnecessary questions of the user, the users can naturally communicate with the system thereby addressing the first letter version's usability issue. Experimental results also show that the recognition rate of the LWP based speech-enabled interface is improved to a sample accuracy rate of 61%. The LWP based system is thus shown to be the most efficient and effective of all the London grammar systems. In addition of the three London interfaces produced, the LWP based system is seen to be the optimal balance between recognition rate, speed of processing and naturalness of interaction.

CHAPTER 6
CONCLUSIONS AND FURTHER RESEARCH

6.1 Concluding Remarks

The aim of this research was to investigate the use of a robust speech-enabled query interface for the ATTAIN travel information system. This research has successfully produced three speech-enabled interfaces for the ATTAIN system. The original directed-dialogue interface for the Nottingham application gave a baseline performance for a speech-enabled interface with a medium sized grammar (1355 bus stop names). A mixed-initiative and multimodal interface that uses text message to output the result has also been designed and implemented that provides improvement in terms of effectiveness, efficiency and user satisfaction compared to the directed-dialogue interface. A third speech-enabled interface that uses a novel Last Word Processor to recognise automatically the last word of the user input has been developed for a London version of ATTAIN. This investigation contributes a method for implementing real-time speech-enabled interfaces that are capable of dealing with very large grammars (27792 bus stop names). The mean performance parameters for the three speech-enabled interfaces are presented in Table 6.1.

	Grammar Size	Sample Accuracy	Completion Time (Sec)
Directed-dialogue Interface for Nottingham	1355	88.5%	111
Mixed-initiative Interface for Nottingham	1355	74.5%	126
Directed-dialogue Interface with LWP for London	23337	61%	120*

Table 6.1 - The Mean Performance Parameters for the Three Speech-Enabled Interfaces

* This Task Completion Time is different to the Completion Time discussed in Chapter 5 because it includes the time spent retrieving and listening to the DIME enquiry result.

The design of the Nottingham directed-dialogue speech-enabled query interface took into consideration aspects of user interface design and speech dialogue management including error recovery. An initial set of experiments was performed to obtain the underlying baseline accuracy of the speech recognition system as well as the overall usability performance of the interface. The results of these experiments show that the average recognition performance of the interface is very good (sample accuracy rate of 88.5%). However the usability test indicated that the interaction style was neither natural nor fast enough for public acceptance. 70% of the sample thought that the speech-enabled interface was actually slower than the text message interface. The need for the user to perform many unnecessary actions in order to accomplish the task were the major reason given by the users for this false impression. In addition, the average user in the sample could only give 52% of the correct data from their audio memory of the journey enquiry results. This was not enough to help the user complete the journey. To overcome these critical difficulties, it was necessary to improve the dialogue management of the interface.

Based on the feedback from using the directed-dialogue speech-enabled query interface, a second version interface was developed that used a mixed-initiative grammar. This grammar allows the interface to process natural language input, rather than directing the user through a rigid sequence of questions and answers. In order to increase the "natural" aspect of the interaction between the user and the travel information system, the use of a multimodal interface was also investigated. The multimodal version of the speech-enabled system allows the use of speech to input the required journey information but uses text messages to present the results of the search back to the user. This method thus overcomes the human short-term memory problem present in the initial version of the interface. 100% of the users said they could complete their journey following the enquiry result. That said, experimental results show that the recognition rate of the interface is reduced (sample accuracy rate of 74.5%) and the task completion time is increased (average user spent time 126 seconds) compared to the directed-dialogue interface. However, because the mixed-initiative interface provided an effective means of error management, the users still felt that they would accomplish the task more efficiently than with the directed-dialogue interface. The other usability factors of mixed-initiative interface are also seen to be a significant improvement on the directed-dialogue system. I argue that the usability of a system is not mainly determined by the ASR performance and task completion time. A poorer ASR performance and longer completion time does not necessarily produce a low probability for task completion and low user satisfaction levels [Koester 2004]. Based on the results obtained from the Nottingham travel information system speech interface, a directed-dialogue speech interface was developed for London. There are 27792 bus stop names in London area. The initial version of the London system used a large grammar file that contained all of the bus stop names. The experiment results show that the system took up to 13 seconds to process one bus stop name. This latency was obviously unacceptable to the users. In addition, the experiment results show that the recognition performance of this interface is not very good (sample accuracy rate of 53%). The system processing time had to be reduced and the speech recognition accuracy rate had to be improved for public acceptance. To overcome these critical difficulties, it was necessary to reduce the grammar size in some way.

A second version interface was developed that used many small grammar files that are grouped according to the first letter of the bus stop name. This system had to ask the user to speak the first letter of their origin or destination before then asking the user for the full origin/destination bus stop name. After the system had recognised the first letter, the system could then recognise the full bus stop names using the small grammar file that only contained the bus stop names that start with this first letter. Recognising the first letter did significantly change the system's performance in terms of processing time (the system now only takes 1.67-2.17 seconds to process one user's entry), but it leaves usability issues. In human to human communication, asking people to speak the first letter of a word and then trying to recognise this word is not a natural means of interaction. The system also only gave the mean sample accuracy rate 53%. This was not accurate enough to be used by the public. In order to achieve the naturalness and robustness goal of this research, the interface had to be improved.

To make the interface interaction more natural, a third version was developed that just asks the user to speak the bus stop names once and records the user's input. The system then retrieved automatically the first letter from the recorded audio and used this to select the appropriate small grammar file for recognition of the full bus stop name. It was hypothesised that this method could help reduce the excrescent communication present in the first letter system. A First Phoneme Processor has been developed that, theoretically, should find automatically the first phoneme from a user's input. Unfortunately the first phoneme recognition results are not encouraging. To overcome this problem, a final version of the interface was developed that segmented the large London grammar on the basis of bus stop word endings (Last Word Processor). This worked much better than the first phoneme detection system because the sound segments are longer and the words sounds are fairly distinct. The experiment results show that this system takes a total of 1.37-1.89 seconds to record and locate the last word in the users input. The major user perceived latency is now 4.01-5.94 seconds. Because the last word recognition system does not need to ask any unnecessary questions of the user, users can naturally communicate with the system. Experimental results also show that the recognition rate of the interface is improved to a sample accuracy rate of 61%. These experimental results show that the method of segmenting large grammars based on recognising the last word to be more effective (accurate) and efficient (faster) than a similar first phoneme recognition based system. The last word recognition based system is also shown to be more natural and more effective than either the first letter or large grammar systems.

6.2 Further Research

Although three speech-enabled query interfaces for the ATTAIN travel information system have been extensively investigated, there are still many unsolved problems. The further research directions can be outlined as follows.

In the mixed-initiative dialogue Nottingham ATTAIN interface, the mean sample ASR accuracy rate is 74.5%. This has the potential to be improved upon. The current interface only returns a single utterance string as the result of each speech-recognition event. If the user's response is not clear, the result will be the one utterance that the speech-recognition engine judges to be the most likely. However, instead of returning the single most likely utterance, it is possible for the speech-recognition engine to return a list of the most likely utterances. Confidence-scoring post-processing can then be investigated as a way of improving the speech recognition error handling. As described by [Pitrelli 2003], confidence-scoring post-processing uses a recognition verification strategy: the computer calculates a measure of recognition confidence, and if this is above a given threshold value then it can be accepted as a correct recognition result. When the confidence value is below the threshold, the result can be "rejected" automatically, meaning that the recognition result is assumed to be unreliable and the item needs to be re-entered. Confidence scoring may consist of a function of appropriate parameters drawn directly from the recognition process, or it may be considered a learning task in which a classifier is trained to use an array of such parameters to distinguish correct recognition from incorrect recognition. Another avenue of future research would be to use a post-processing approach to confidence scoring to eliminate the need for the confirmation stage. If the system can determine automatically that it is 'confident' in its recognition result, then there is no need for the system to ask the user if the recognition is correct.

Another possible method for improving the speech recognition accuracy would to be use a speaker-adaptive approach. A speaker-adaptive system can offer speaker-independent recognition at first, but then attempt to improve recognition accuracy using speaker-dependent adaptations [Anastasakos 1996]. The current ATTAIN system uses a speaker-independent ASR system which generally has error rates that are two to three times higher than that of speaker-dependent speech recognition systems [Paul 1989 and Kubala 1990]. If the system were to collect speaker data during users interactions with the ATTAIN system, the ASR system parameters could be adapted to the specific speaker to reduce the error rate. By storing these speaker profiles in a mobile phone number history file the accuracy of speech recognition could be improved over time. However, since adaptation for the ATTAIN system would need to be based on only limited system interaction data, any adaptation algorithm would need be consistent with using limited training data. Huang introduced an adaptive model combination method for speaker adaptation which used just such a parameter adaptation technique [Huang 2002]. He claims that the relative error rate reduction of 12.27% is achieved when only 10 utterances are available. This level of interaction approaches that of the ATTAIN system so could be investigated as way of improving the underlying ASR system.

This research has also developed a component, (Last Word Processor LWP), module which can find automatically the last word from a user's utterance. This component can be used in speech-enabled interfaces with a very large grammar to sub-divide the grammar in order to greatly reduce the ASR processing time for one user's input. The accuracy of the LWP component depends on many factors; speaking rates being a major problem. Unfortunately, the current LWP performs significantly worse on fast speech. There are several directions in which this Last Word Processor can be improved for fast speech.

The attempts for improvements on the recognition of fast speech in ASR can be learned. [Mirghafori 1995] has proven that setting high transition probabilities on arcs leaving from one state to a later state in the Hidden Markov Model is useful. Another option is create speaking-rate dependent acoustic models [Martinez 1998] by collecting a corpus of speech with normal, slow, and fast speaking rates. Detecting the speaking rate has also attracted some attention [Morgan 1998] by analysing the spectrum of speech with different speaking rates. [Richardson 1999] proposes cepstrum length normalisation (CLN) which attempts to normalise the phone duration by stretching the length of the utterance in the cepstrum domain so that it matches the acoustic model trained on the regular speech. Using the Hidden Markov Models (HMM) as the acoustic models for distinguishing the last word in LWP would probably be the best way forward for this research. However, as HMMs are complex statistical models, future research have to be aware of the processing time overhead implicit with using HMMs. The main aim of this research was to produce a natural communication interface that would operate in **real-time**. Using advanced techniques to improve performance is only acceptable if the extra processing time overhead do not seriously compromise the real-time interaction objective.

Finally, a different modality of input is also required in situations where the ASR persistently fails to recognise what the user has said because of the user's ambiguous pronunciation. The user will be stuck at this point without access to a different input modality. Keypad input would be the best alternative input method. When the system detects "trouble" conditions, the current system gradually reduces the allowed scope of user input by applying more and more input constraints and specific prompts. After two attempts, the system switches to a directed-dialogue mode where it continues to prompt the user for one piece of information at a time. If the system still cannot recognise what the user has said, the system could ask the user to input the data using the phone keypad. Using such a modality requires that the system must remain open to the possibility that the hypothesised word is misspelt. In future research, an intelligent spell checker could be built. If the spell checker is not robust enough to identify all the words from the misspelling, the system would need to initiate a disambiguation sub-dialogue to resolve the ambiguity.

References:

[Adams 1999] L. J. Adams, R. I. Damper, S. Harnad and W. Hall: A system Design for Human Factors Studies of Speech-Enabled Web Browsing. In: Proceedings of ESCA Workshop on Interactive Dialogue in Multi-Modal Systems Interactive Dialogue in Multi-Modal Systems X, Kloster Irsee, Germany, Page 37-140.

[Ainsworth 1992] W. A. Ainsworth and S. R. Pratt: Feedback Strategies for Error Correction in Speech Recognition Systems. In: International Journal of Man-Machine Studies, Volume 36, Issue 6, Page 833 - 842.

[Allen 1987] J. Allen, S. Hunnicutt and D. Klatt: From Text to Speech: The MITalk System. Cambridge University Press. UK.

[Allen 2001] J. Allen, G. Ferguson and A. Stent: An Architecture for More Realistic Conversational Systems. In: Proceedings of the sixth International Conference on Intelligent User Interfaces, Santa Fe, New Mexico, USA, Page 14 - 17.

[Anastasakos 1996] T. Anastasakos, J. McDonough, R Schwartz, and J. Makhoul: A Compact Model for Speaker-Adaptive Training. In: Proceedings of the Fourth International Conference on Spoken Language Processing (ICSLP 96), Philadelphia, PA, USA, Page 1137-1140.

[Batt 1999] S. Bett: Can We Pin Down the Number of Phonemes in English? In: Simpl Speling, A. Campbell, Eds., Newsletters of the Simplified Spelling, Page 7.

[Bazzi 2000] I. Bazzi and J. Glass: Modelling Out-Of-Vocabulary Words for Robust Speech Recognition. In: Proceedings of International Conference on Spoken Language Processing (ICSLP 2000), Beijing, China, Page 401-404.

[Berglund 2003] A. Berglund and P. Ovarfordt: Error Resolution Strategies for Interactive Television Speech Interfaces. In: Proceedings of the Ninth IFIP TC13

International Conference on Human-Computer Interaction (INTERACT 2003), Zurich, Switzerland, Page 105-112

[Bernhard 1996] S. Bernhard, M. Bradm and W. Alex: Interactive recovery from speech recognition errors in speech user interfaces. In: Proceedings of Fourth International Conference on Spoken Language Processing (ICSLP 96), Philadelphia, PA, USA, Page 865-868.

[Bevan 1991] N. Bevan, J. Kirakowski and J. Maissel: What is Usability? In: Human Aspects in Computing: Design and Use of Interactive Systems with Terminals, H. J. Bullinger, Eds, Elsevier, Amsterdam, Page 651-655.

[Bevan 1994] N. Bevan and M. Macleod: Usability Measurement in Context. In: Journal of Behaviour and Information Technology, Volume 13, Page 132-145.

[Bevan 1995] N. Bevan: Measuring Usability as Quality of Use. In: Journal of Software Quality, Volume 4, Page 115-130.

[Beutnagel 1999] M. Beutnagel, A. Conkie and J. Schroeter, Y. Stylianou, and A. Syrdal: The AT&T Next-Gen TTS System. In: Proceedings of Joint Meeting of ASA, EAA, and DAGA 1999, Berlin, Germany, Page 18-24.

[Bevan 1997] N. Bevan and M. Azuma: Quality In Use: Incorporating Human Factors into the Software Engineering Lifecycle. In: Proceedings of the Third IEEE International Software Engineering Standards Symposium and Forum (ISESS'97), Walnut Creek, California, USA, Page 169-179.

[Bolt 1980] R. Bolt: Put That There: Voice and Gesture at the Graphics Interface. In: Computer Graphics ACM SIGGRAPH80, Volume 14, Issue 3, Page 262-270.

[Bohus 2003] D. Bohus and A. Rudnicky: RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda. In: Proceedings of the European Conference on Speech Communication and Technology (Eurospeech 2003), Geneva, Switzerland, Page 597-600.

[Boyce 1999] S. Boyce: Spoken natural language dialogue systems: User interface issues for the future. In: Book of Human Factors and Voice Interactive Systems, D. Gardner-Bonneau, Eds. Kluwer Academic Publishers, Boston, Page 37-61.

[Bretan 1995] I. Bretan, A. L. Ereback, C. MacDermid and A. Waern: Simulation-Based Dialogue Design for Speech-Controlled Telephone Services. In: Proceedings of Conference on Human Factors in Computing Systems (CHI'95) Denver, Colorado, USA, Page 145-146.

[Burke 2000] M. Burke: Thinking Together: New Forms of Thought System for a Revolution in Military Affairs (DSTO-RR-0173). In: Report of Defence Science and Technology Organisation (DSTO), Department of Defence, Australia. Available at: http://www.dsto.defence.gov.au/publications/2250/DSTO-RR-0173.pdf, accessed on 12/11/2005.

[Chattractichar 2003] J. Chattratichart and J. Brodie: Envisioning a Mobile Phone for 'All' Ages. In: Proceedings of the Ninth IFIP TC13 International Conference on Human-Computer Interaction (Interact 2003), Zurich, Switzerland, Page 725-728.

[CCIR 2005] Centre for Communication Interface Research: Dialogue Design Guide: Priming Metaphors as Aids to Usage. In: Report of SPOTLIGHT Project, Centre for Communication Interface Research, Department of Electronics and Electrical Engineering, University of Edinburgh, Available at: http://spotlight.ccir.ed.ac.uk/public_documents/Dialogue_design_guide/priming_metaphors.htm, accessed on 12/11/2005

[Clark 1986] H. H. Clark and D. Wilkes-Gibbs: Referring as collaborative process. In: Journal of Cognition Science, Volume 22, Page 1-39.

[Clark 1989] H. H. Clark and E. F. Schaefer (1989): Contributing to Discourse. In: Journal of Cognitive Science, Volume 13, Page 259-294.

[Cohen 1995] P. R. Cohen and S. L. Oviatt: The Role of Voice Input for Human-Machine Communication. In: Book of Voice Communication between Humans and Machines, D. B. Roe and J. G. Wilpon, Ed, National Academy Press, Washington, DC, USA, Page 34-75.

[Cox 1998] R. V. Cox, B. G. Haskell, Y. LeCun, B. Shahraray, and L. Rabiner: On the Applications of Multimedia Processing to Communications. In: Proceedings of IEEE, Volume 86, Page 755-824.

[Dahlbäck 1993] N. Dahlbäck, A. Jönsson and L. Ahrenberg: Wizard of Oz Studies - Why and How. In: Proceedings of The ACM International Workshop on Intelligent User Interfaces, Orlando, Florida, USA, Page 193-200.

[Danis 1989] C. M. Danis: Developing Successful Speakers for and Automatic Speech Recognition System. In: Proceedings of the Human Factor Society, 33rd Annual Meeting, Santa Monica, CA, USA, Page 300-304.

[Davis 1952] H. K. Davis, R. Biddulph and S. Balashek: Automatic Recognition of Spoken Digits. In: American Journal of Otolaryngology, Volume 24, Page 637-642.

[Davis 1996] A. W. Davis: Speech Recognition Technology Backgrounder. In: TechOnLine Review, Digital Library Network for Engineering and Technology , Volume 1, Number 2, Available at: http://www.techonline.com/community/ed_resource/feature_article/20044, accessed on 12/11/2005.

[Darves 2002] C. Darves and S. L. Oviatt: Adaptation of Users' Spoken Dialogue Patterns in a Conversational Interface. In: Proceedings of the 7th International Conference on Spoken Language Processing (ICSLP 2002), Denver, Colorado, USA, Volume 1, Page 561-564.

[Deroo 1998] O. Deroo (1998): "Hidden Markov Models and Neural Networks for Speech Recognition" In: PhD Thesis, Faculté Polytechnique de Mons, Available at: http://tcts.fpms.ac.be/publications/phds/deroo/these.zip, accessed on 12/11/2005.

[Deshmukh 2002] O. Deshmukh, C. Y. Espy-Wilson and A. Juneja: Acoustic-Phonetic Speech Parameters for Speaker-Independent Speech Recognition. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, Orlando, Florida, USA, Volume 1, Page 593-596.

[Dutoit 1997] T. Dutoit: An Introduction to Text-To-Speech Synthesis (Text, Speech and Language Technology. Book Published by Kluwer Academic Publishers.

[Dutoit 2003] T. Dutoit and Y. Stylianou: Text-to-Speech Synthesis. In: Handbook of Computational Linguistics, R. Mitkov, ed., Oxford University Press, Page 323-338.

[Edwards 2002] A. Edwards: Multimodal Interaction and People with Disabilities. In: Book of Multimodality in Language and Speech Systems, B. Granstro¨m, D. House and I. Karlsson, Eds., Kluwer Academic Press, Page 73 - 92.

[Faulkner 2000] X. Faulkner: Usability Engineering. Book Published by Palgrave, Page 119.

[Filisko 2004] E. Filisko and S. Seneff: Error Detection and Recovery in Spoken Dialogue Systems. In: Proceedings of Workshop on Spoken Language Understanding for Conversational Systems, Boston, Massachusetts, USA, Page 31 - 38.

[Fontaine 1996] V. Fontaine and H. Bourlard: Speaker-Dependent Speech Recognition Based on Phone-Like Units Models -Application to Voice Dialing. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1997), Munich, Germany, Volume 2, Page 1527-1530.

[Frankish 1992] C. Frankish, D. Jones and K. Hapeshi: Decline in Accuracy of Automatic Speech Recognition as a Function of Time on Task: Fatigue or Voice Drift? In: International Journal of Man-Machine Studies, Volume 36, Page 797 - 816.

[Franz 2002] A. Franz and B. Milch: Searching the Web by Voice. In: Proceedings of 19th International Conference on Computational Linguistics, Taipei, Taiwan, China, Volume2, Page 1213-1217.

[Ferguson 2005] G. Ferguson and J. Allen: Mixed-Initiative Dialogue Systems for Collaborative Prob-lem-Solving. In: Proceedings of the AAAI Fall Symposium on Mixed-Initiative Problem Solving Assistants, Washington DC, USA.

[Furnas 1987] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais: The Vocabulary Problem in Human-System Communication. In: Communications of the ACM, Volume 30, Issue 11, Page 964-971.

[Garian 1993] D. Garlan and M. Shaw: An Introduction to Software Architecture. In: Advances in Software Engineering and Knowledge Engineering, Series on Software Engineering and Knowledge Engineering, V. Ambriola and G. Tortora, Eds., World Scientific Publishing Company, Volume 2, Page 1 - 39.

[Giachin 1996] E. Giachin, S. McGlashan (1996): "Spoken Language Dialogue Systems. In: Corpus-Based Methods in Language and Speech Processing, K. Church, S. Young and G. Bloothooft, Eds., Published by Kluwer, Page 69-117.

[Gollner 1994] W. Gollner and J. Harvey: Time-Compressed Speech: Emerging Ideas for Audio in Computer-Based Learning. In: Proceedings of Australian Association for Research in Education Conference (AARE 1994), Newcastle, Australia, Page 384-387.

[Gorin 1999] A. L. Gorin, E. Ammicht and T. Alonso: Knowledge Collection for Natural Language Spoken Dialog Systems. In: Proceedings of the European

Conference on Speech Communication and Technology (Eurospeech 1999), Budapest, Hungary, Volume 3, Page 1375-1378.

[Gould 1982] J. Gould: Writing and Speaking Letters and Messages. In: International Journal of Man-Machine Studies Volume 16, Page 147-171.

[Grasso 1997] M. A. Grasso: Speech Input in Multimodal Environments: Effects of Perceptual Structure on Speed, Accuracy, and Acceptance. In: PhD Thesis, University of Maryland, Baltimore, USA, Available at: http://ebiquity.umbc.edu/_file_directory_/papers/192.pdf, accessed on 12/11/2005.

[Harte 2002] D. Harte: Non Asymptotic Binomial Confidence Intervals. In: Publication of Statistics Research Associates, Available at: http://www.statsresearch.co.nz/pdf/confint.pdf, accessed on 12/11/2005.

[Hazen 2001] T. J. Hazen and I. Bazzi: A Comparison and Combination of Methods for OOV Word Detection and Word Confidence Scoring. In: Proceedings of the 2001 International Conference on Acoustics, Speech, and Signal Processing, Salt Lake City, USA, Page 397-400.

[Hirsch 2000] H. Hirsch and D. Pearce: The AURORA Experimental Framework for the Performance Evaluations of Speech Recognition Systems under Noisy Conditions. In: Proceedings of International Conference on Spoken Language Processing (ICSLP 2000), Beijing, China, Volume 4, Page 29-32.

[Houser 1999] S. E. Houser, T. F. Sabir, G. R. Thoma: Speech recognition for program control and data entry in a production environment. In: Intelligent Systems in Design and Manufacturing II, B. Gopalakrishnan and S. Murugesan, Eds. Page 24-34.

[Huang 1993] X. Huang, F. Alleva, H. W. Hon, M. Y. Hwang, and R. Rosenfeld: The SPHINX-II speech recognition system: an overview. In: Journal of Computer Speech and Language, Volume 7, No. 2, Page 137 - 148.

[Huang 2000] C. M. Huang, M. Y. Jang and Y. C. Chao: CTW: An Integrated Computer and Telephone-Accessed WWW System. In: Journal of Software - Practice and Experience Volume 30, Issue 13, Page 1485-1507.

[Huang 2002] C. Huang, T. Chen and E. Chang: Adaptive Model Combination for Dynamic Speaker Selection Training. In Proceedings of International Conference on Spoken Language Processing (ICSLP 2002), Denver, Colorado, USA, Volume 1, Page 65-68.

[Hussien 2005] S. Hussien and G, Björn: A Speaker Independent Continuous Speech Recognizer for Amharic. In: Proceedings of 9th European Conference on Speech Communication and Technology (INTERSPEECH 2005), Lisbon, Portugal, Page 4-8

[IBM 2005] IBM: ViaVoice. Available at: http://www.scansoft.com/viavoice/advanced/, accessed on 12/11/2005.

[Inria 2004] Inria: Man-machine oral and multimodal communication. In: Research Project Activity Reports, The French National Institute for Research in Computer Science and Control, Available at: http://www.inria.fr/rapportsactivite/RA2003/cordial2003/cordial.html, accessed on 12/11/2005.

[Intel 2004] Intel: Open Standards-based Voice Services Platform. In: Solution Blueprint, Intel, Available at: http://www.intel.com/business/bss/solutions/blueprints/pdf/sb_voicegenie0247.pdf, accessed on 12/11/2005.

[ISO 1998] ISO: DIS 9241-11, Guidance on Usability. Available at: http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=16883&scopelist=ALL, accessed on 12/11/2005.

[Ito 2001] E. Ito: Multi-modal Interface with Voice and Head Tracking for Multiple Home Appliances. In: Proceedings of Conference on Human-Computer Interaction (INTERACT 2001), Tokyo, Japan, Page 727-728,

[Jones 1990] D. M. Jones, K. Hapeshi and C. Frankish: Design Guidelines for Speech Recognition Interfaces. In: Journal of Applied Ergonomics, Volume 20, Issue 1, Page 40 - 52.

[Jordan 1998] P. Jordan: Human Factors for Pleasure in Product Use. In: Journal of Applied Ergonomics Volume 29, Issue 1, Page 25-33.

[Junqua 1996] J. C. Junqua and J. P. Haton: Robustness in Automatic Speech Recognition: Fundamentals and Applications. Kluwer Academic Publishers, Boston, USA, Page 440.

[Kamm1997] C. Kamm and L. Rabiner: The Role of peech Processing in Human-Computer Intelligent Communication. In: Journal of Speech Communication, Volume 23, Issue 4, Page 263-278.

[Kazantsev 2004] A. V. Kazantsev: Methods of Phoneme Distinction in a Continious Speech Recognition System. In: Proceedings of the European Workshop on the Integration of Knowledge, Semantics and Digital Media Technology (EWIMT 2004), London, UK, Page 127-133.

[Kemble 2001] K. Kemble: An Introduction to Speech Recognition. Available at: http://www.voicexmlreview.org/Mar2001/bios.html#kemble, accessed on 12/11/2005.

[Klatt 1987] D. Klatt: Review of Text-to-Speech Conversion for English. In: Journal of the Acoustical Society of America, JASA Volume 82, Page 737-793.

[Klemmer 2000] S. R. Klemmer, A. K. Sinha, J. Chen, J. A. Landay, N. Abookbaker, and A. Wang: SUEDE: A Wizard of Oz Prototyping Tool for Speech User Interfaces.

In: Proceedings of the13th Annual ACM symposium on User Interface Software and Technology (UIST 2000), Page1-10.

[Koester 2004] H. H. Koester: Usage, Performance, and Satisfaction Outcomes for Experienced Users of Speech Recognition. In: Journal of Rehabilitation Research and Development, Volume 41, Page 739-754.

[Komorita 1963] S. Komorita: Attitude Content, Intensity and the Neutral Point on a Likert Scale. In: Journal of Social Psychology, Volume 61, Page 327-334.

[Kosonen 1999] I. Kosonen and A. Bargiela: A Distributed Traffic Monitoring and Information System. In: Journal of Geographic Information and Decision Analysis, Volume 3, Issue 1, Page 31-40.

[Kubala 1990] F. Kubala and R. Schwartz: A New Paradigm for Speaker-Independent Training and Speaker Adaptation. In: Proceedings of Workshop on Speech and Natural Language, Hidden Valley, Pennsylvania, United States, Pages 306 - 310

[Kun 2004] A. L. Kun, W. T. Miller, A. Pelhe and R. L. Lynch: A Software Architecture Supporting In-Car Speech Interaction. In: Proceedings of IEEE Intelligent Vehicles Symposium, Parma, Italy.

[Landaruer 1993] T. K. Landaruer and J. Nielsen: A Mathematical Model of the Finding of Usability Problems. In: Proceeding of Conference on Human Factors in Computing Systems (INTERCHI'93), Amsterdam, Netherlands, Page 206-213.

[Larsson 2002] S. Larsson: Issue-Based Dialogue Management. In: PhD Thesis, Gőteborg University, Sweden.

[Lee 1990] K. F. Lee, H. W. Hon, and R. Reddy: An Overview of the SPHINX Speech Recognition System. In: IEEE Transactions on Acoustics, Speech and Signal Processing, Volume 38, No. 1, Page 35 - 45.

[Levinson 1993] S. E. Levinson, J. P. Olive and J. S. Tschirgi: Speech Synthesis in Telecommunications. In: IEEE Communications Magazine, Volume 31, Page 46-53.

[Levow 1997] G. Levow: Making Sense of Silence. In: Proceedings of Conference on Human Factors in Computing Systems 1997 Workshop on Speech User Interface Design Challenges, Atlanta, USA.

[Lieberman 2005] H. Lieberman, A. Faaborg, W. Daher and José Espinosa: How to Wreck a Nice Beach You Sing Calm Incense. In: Proceedings of International Conference on Intelligent User Interfaces (IUI 2005), San Diego, USA, Page 278-280.

[Ly 1994] E. Ly, C. Schmandt: A Conversational Learning Speech Interface. In: Proceedings of AAAI'94 Spring Symposium on Multi-Media Multi-Modal Systems, Stanford, CA, USA.

[MacDermid 1996] C. MacDermid and M. Goldstein: The 'Storyboard' Method: Establishing an Unbiased Vocabulary for Keyword and Voice Command Applications. In: Proceedings of HCI'96 Industry Day & Adjunct, British Computer Society Conference on Human Computer Interaction, Page 104-109.

[Malkin 2004] J. Malkin, X. Li and J Bilmes: Custom Arithmetic for High-speed, Low-resource ASR Systems. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2004), Montreal, Canada.

[ManÉ 1996] A. ManÉ, S. Boyce and D. Karis: Designing the User Interface for Speech Recognition Applications. In: Workshop of ACM Special Interest Group on Computer-Human Interaction, Volume 28, Issue4, Page 29-34.

[Mankoff 2000] J. Mankoff, S. E. Hudson and G. D. Abowd: OOPS: A Toolkit Supporting Mediation Techniques for Resolving Ambiguity in Recognition-Based Interfaces. In: Journal of Computers and Graphics Volume 24, Issue6, Page 819 - 834.

[Martin 1980] T. B. Martin and J. R. Welch: Practical Speech Recognisers and Some Performance Effectiveness Parameters. In: Trends in Speech Recognition, W. A. Lea Eds., Prentice Hall Press, Upper Saddle River, NJ, Page 24-38.

[Martin 1999] D. L. Martin, A. J. Cheyer and D. B. Moran: The Open Agent Architecture: A Framework for Building Distributed Software Systems. In: International Journal of Applied Artificial Intelligence, Volume13, Issue 1 - 2, Page 91 - 128.

[Martinez 1998] E. Martinez, D. Tapias and J. Alvarez: Towards Speech Rate Independence in Large Vocabulary Continuous Speech Recognition. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, Seattle, USA, Page 725-728.

[McGee 1998] D. R. McGee, P. R. Cohen and S. Oviatt: Confirmation in Multimodal Systems. In: Proceedings of the International Joint Conference of the Association for Computational Linguistics and the International Committee on Computational Linguistics, Montreal, Quebec, Canada, Page 823 - 829.

[Microsoft 2005A] Microsoft: Speech Application Language Tags (SALT). Available at: http://www.microsoft.com/speech/evaluation/speechtags/, accessed on 12/11/2005.

[Microsoft 2005B] Microsoft: Multiple Channel Audio Data and WAVE Files. Available at: http://www.microsoft.com/whdc/device/audio/multichaud.mspx, accessed on 12/11/2005.

[Miller 2002] M. Miller: VoiceXML: 10 Projects to Voice Enable Your Web Site. Published by John Wiley & Sons, New York.

[Minami 1993] Y. Minami, K. Shikano, T. Yamada and T. Matsuoka: Very-Large-Vocabulary Continuous Speech Recognition Algorithm for Telephone Directory

Assistance. In: Proceedings of the European Conference on Speech Communication and Technology (Eurospeech 1993), Berlin, Germany, Page 2129-2132.

[Mindmaker 2005] Mindmaker: Game Commander: Voice Control for Games & Simulations. Available at: http://www.gamecommander.com/, accessed on 12/11/2005.

[Mirghafori 1995] N. Mirghafori, E. Fosler and N. Morgan: Fast Speakers in Large Vocabulary Continuous Speech Recognition: Analysis & Antidotes. In: Proceedings of the European Conference on Speech Communication and Technology Madrid, Spain, Page 491-494.

[Moraes 2002] I. Moraes: VoiceXML, CCXML, SALT: Architectural Tools for Enabling Speech Applications. In: XML Journal, Volume 3, Issue 9, Page 30-34.

[Morgan 1998] N. Morgan and E. Fosler-Lussier: Combining Multiple Estimators of Speaking Rate. In: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, Seattle, USA, Page 729-732.

[Murray 1993] A. C. Murray, C. R. Frankish and D. M. Jones: Data-Entry by Voice: Facilitating Correction of Misrecognitions In: Interactive Speech Technology: Human Factors Issues in the Application of Speech Input/Output to Computers, C. Baber and J. M. Noyes, Eds., Published by Taylor and Francis, Bristol, PA, USA, Page 137 - 144.

[Mynatt 1994] E. Mynatt and G. Weber: Nonvisual Presentation of Graphical User Interfaces. In: Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI'94), Boston, MA, USA, Page 166 - 172.

[NeuVoice 2005] NeuVoice: Dialler. Available at: http://www.neuvoice.com/products/dialler.php, accessed on 12/11/2005.

[Nielsen 1994] J. Nielsen: Usability Engineering. Published by Academic Press, San Diego, USA.

[Neilson 2002] J. Neilson: Usability for Senior Citizens: 46 Design Guidelines Based on Usability Studies with people Age 65 and Older. Available at: http://www.useit.com/alertbox/20020428.html, accessed on 12/11/2005.

[Neti 2000] C. Neti, G. Potamianos, J. Luettin, I. Matthews, H. Glotin, D. Vergyri, J. Sison, A. Mashari, and J. Zhou: Audio-visual speech recognition. In: Technical Report WS00AVSR, Johns Hopkins University. Available at: http://www.clsp.jhu.edu/ws2000/groups/av_speech/, accessed on 12/11/2005.

[Nielsen 2005] J. Nielsen: Why You Only Need to Test With 5 Users. Available at: http://www.useit.com/alertbox/20000319.html, accessed on 12/11/2005.

[Norman 1983] D. A. Norman: Some Observations on Mental Models. In: Mental Models, D. Gentner and A. Stevens, Eds., Published by Lawrence Erlbaum Associates, Hillsdale, New Jersey, USA, Page 7-14.

[Norman 1991] M. Norman, G. Fraser and N. Gilbert: Simulating Speech Systems. In Journal of Computer Speech and Language, Volume 5, Page 81-99.

[Noyes 2001] J. M. Noyes: Talking and writing-how natural in human-machine interaction? In: International Journal of Human-Computer Studies. Issue 55, Volume 4, Page 503-519.

[NTU 2002] Nottingham Trent University: Advanced Traffic and Travel Information system. Available at: http://www.intelligentmodelling.org.uk/Projects/grr32468/attain.html, accessed on 12/11/2005.

[Oberteuffer 1995] J. Oberteuffer: Commercial Applications of Speech Interface Technology: An Industry at the Threshold. In: Proceedings of the National Academy of Sciences, Volume 92, Page 10007-10010.

[O'Neill] I. O'Neill, P. Hanna, X. Liu and M. McTear: The Queen's Communicator: An Object-Oriented Dialogue Manager. In Proceedings of the European Conference on Speech Communication and Technology (Eurospeech 2003), Geneva, Switzerland, Page 593 - 596.

[Oulasvirta 2004] A. Oulasvirta and P. Sarriluoma: Long-term Working Memory and Interrupting Messages in Human - Computer Interaction. In: Journal of Behaviour & Information Technology. Volume 23, Issue 1, Page 53 - 64.

[Oviatt 1991] S. L. Oviatt and P. R. Cohen: Discourse Structure and Performance Efficiency in Interactive and Noninteractive Spoken Modalities. In: Journal of Computer Speech and Language. Volume 5, Issue 4, Page 297-326.

[Oviatt 1994] S. Oviatt and E. Olsen: Integration Themes in Multimodal Human-Computer Interaction. In: Proceedings of the International Conference on Spoken Language Processing, Yokohama, Japan, Page 551 - 554.

[Oviatt 1999A] S. Oviatt: Mutual Disambiguation of Recognition Errors in a Multimodal Architecture. In: Proceedings of the SIGCHI conference on Human Factors in Computing Systems: the CHI is the limit, Pittsburgh, Pennsylvania, USA, Page 576-583.

[Oviatt 1999B] S. Oviatt: Ten Myths of Multimodal Interaction. In: Communications of the ACM, Volume 42, Issue 11, Page 74 - 81.

[Oviatt 2000A] S. Oviatt: Designing the User Interface for Multimodal Speech and Gesture Applications: State-of-the-art Systems and Research Directions. In: Journal of Human Computer Interaction, Volume 15, Issue 4, Page 263 - 322.

[Oviatt 2000B] S. Oviatt: Multimodal Interface Research: A Science without Borders. In Proceedings of the sixth International Conference on Spoken Language Processing (ICSLP 2000), Beijing, China, Page 1-4.

[Pakucs 2001] B. Pakucs: The Speech-Enhanced World. In: A Contest for Young Researchers at Eurospeech2001 (Imagination2001), Scandinavia, Aalborg, Denmark.

[Pakucs 2003] B. Pakucs: Towards Dynamic Multi-Domain Dialogue Processing. In Proceedings of the European Conference on Speech Communication and Technology (Eurospeech 2003), Geneva, Switzerland, Page 741 - 744.

[Peacocke 1990] R. D. Peacocke and D. H. Graf: An Introduction to Speech and Speaker Recognition. In: Journal of IEEE Computer, Volume 23, Issue 8, Page 26-33.

[Paul 1989] D. Paul: The Lincoln Robust Continuous Speech Recogniser. In: Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1989), Glasgow, UK, Page 449- 452.

[Peissner 2001] M. F. Heidmann and J. Ziegler: Simulating Recognition Errors in Speech User Interface Prototyping. In: Usability Evaluation and Interface Design , M. J. Smith, G. Salvendy, D. Harris, and R .J. Koubek, Eds, Lawrence Erlbaum Associates, Inc., Mahwah, NJ, USA, Page 233-237

[Peissner 2002] M. Peissner: What the Relationship between Correct Recognition Rates and Usability Measures Can Tell Us about the Quality of a Speech Application. In Proceedings of 6th International Scientific Conference on Work With Display Units (WWDU 2002), Berchtesgaden, Germany, Page 296-298.

[Philips 2005] Philips (2005): Philips Dictation Systems. Available at: http://www.dictation.philips.com/index.php?id=start, accessed on 12/11/2005.

[Pitrelli 1995] J. F. Pitrelli, C. Fong and H. C. Leung: PhoneBook: A Phonetically-Rich Isolated-Word Telephone-Speech Database. In: Proceedings of International Conference on Acoustics, Speech, and Signal Processing (ICASSP 1995), Detroit, Michigan, USA, Page 101-104.

[Pitrelli 2003] J. F. Pitrelli and M. P. Perrone (2003): Confidence-Scoring Post-Processing for Off-Line Handwritten-Character Recognition Verification. In:

Proceedings the Seventh International Conference on Document Analysis and Recognition, Edinburgh, Scotland, Page 278-282.

[Placeway 1996] P. Placeway, S. Chen, M. Eskenazi, U. Jain, V. Parikh, B. Raj, M. Ravishankar, R. Rosenfeld, K. Seymore, M. Siegler, R. Stern, and E. Thayer: The 1996 HUB-4 Sphinx-3 system. In: Proceedings of the DARPA Speech Recognition Workshop. Chantilly, VA: DARPA. Available at: http://www.nist.gov/speech/publications/darpa97/pdf/placewa1.pdf, accessed on: 12/11/2005.

[Prodanov 2003] P. J. Prodanov, A. Drygajlo, G. Ramel, M. Meisserl and R. Siegwart: Voice Enabled Interface for Interactive Tour-Guide Robots. In: Journal of Advanced Robotics, Volume 17, Issue7, Page 599-616.

[Rabiner 1993] L. Rabiner and B. Juang: Fundamentals of Speech Recognition. Prentice-Hall Signal Processing Series, Upper Saddle River, NJ, USA.

[Richardson 1999] M. Richardson, M. Hwang, A. Acero and X. Huang: Improvements on Speech Recognition for Fast Talkers. In: Proceedings of the European Conference on Speech Communication and Technology (Eurospeech 1999), Budapest, Hungary, Volume 1, Page 411-414.

[Richardson 2000] M. Richardson, J. Bilmes and C. Diorio: Hidden-Articulator Markov Models for Speech Recognition. In Proceedings of Conference on Automatic Speech Recognition (ASR2000), Paris, France, Page 133-139.

[Rosenfeld 2001] R, Rosenfeld, D. R. Olsen and A. I. Rudnicky: Universal Speech Interfaces. In: Journal of Interactions Volume 8, Issue 6, Page 34-44.

[Raux 2005] A. Raux, B. Langner, D. Bohus, A. W. Black and M. Eskenazi: Let's go public! taking a spoken dialog system to the real world. In: Proceedings of Interspeech 2005 - Eurospeech, Lisbon, Portugal, Page 885-888.

[Rudnicky 1999] A. Rudnicky and W. Xu: An Agenda-based Dialog Management Architecture for Spoken Language Systems. In Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop, Keystone, Colorado, USA, Page 337 - 340.

[Rybczynski 1995] W. Rybczynski: Controlling Technology Means Controlling Ourselves. In: Science and Technology Today, N. R. MacKenzie Eds., St. Martins Press, New York, USA, Page 103-109.

[Sauro 2005A] J. Sauro: Restoring Confidence in Usability Results. Available at: http://www.measuringusability.com/conf_intervals.htm, accessed on 12/11/2005.

[Sauro 2005B] J. Sauro: You Don't Need a Large Sample of Users to Obtain Meaningful Data: Continuous Data (e.g. Task Time). Available at: http://www.measuringusability.com/sample_continuous.htm, accessed on 12/11/2005.

[Sawhney 1998] N. Sawhney and C. Schmandt: Speaking and Listening on the Run: Design for Wearable Audio Computing" Proceedings of International Symposium on Wearable Computing (ISWC 1998), Pittsburgh, Pennsylvania, USA, Page 108-115.

[ScanSoft 2004] ScanSoft: ScanSoft Speech Technology Drives CULLMANN's Universal Hands-free Kit for Mobile Phones. Available at: http://www.scansoft.co.uk/news/20050110_cullmans.asp, accessed on 12/11/2005.

[Schalkwyk 2003] J. Schalkwyk, L. Hetherington and E. Story: Speech Recognition with Dynamic Grammars Using Finite-State Transducers. In: Proceedings of the European Conference on Speech Communication and Technology (Eurospeech 2003), Geneva, Switzerland, Page 1969-1972.

[Schmandt 1993] C. Schmandt: The Telephone as a Computer Terminal. In: Proceedings the First ACM International Conference on Multimedia, Anaheim, CA, USA, Page 373-382.

[Schegloff 1977] E. A. Schegloff, G. Jefferson and H. Sacks: The Preference for Self-Correction in the Organisation of Repair in conversation. In: Journal of Language, Volume 53, Page 361-382.

[Schiaffino 2004] S. Schiaffino, A. Amandia: User - Interface Agent Interaction: Personalization Issues. In: International Journal of Human-Computer Studies, Volume 60, Page 129-148.

[Schroeter 2001] J. Schroeter: The Fundamentals of Text to Speech Synthesis. In: VoiceXML Review, March 2001. Available at: http://www.voicexmlreview.org/Mar2001/features/tts.html, accessed on 12/11/2005.

[Seltzer 2005] M. Seltzer and A. Acero: Training Wideband Acoustic Models Using Mixed-Bandwidth Training Data via Feature Bandwidth Extension. In Proceedings of Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2005), Philadelphia, PA, USA, Page 921-924.

[Seneff 1998] S. Seneff, E. Hurley, R. Lau, C. Pao, P. Schmid and V. Zue: Galaxy-II: A Reference Architecture for Conversational System Development. In: Proceedings of the Fifth International Conference on Spoken Language Processing (ICSLP9 1998), Sydney, Australia, Page 931 - 934.

[Seneff 2000] S. Seneff and J. Polifroni: Dialogue Management in the MERCURY Flight Reservation System. In: Proceedings of ANLP Conversational Systems Workshop.Seattle, USA, Page 1-6.

[Seneff 2002] S. Seneff : Response Planning and Generation in the MERCURY Flight Reservation System. In: Journal of Computer Speech and Language, Volume 16, Page 283-312.

[Shackel 1986] B. Shackel: Ergonomics in Design for Usability. In: People and Computers: Designing for Usability, M. D. Harrison, A.F. Monk, Eds., Cambridge University Press, Cambridge, UK, Page 44 - 64.

[Skantze 2003] G. Skantze: The Use of Speech Recognition Confidence Scores in Dialogue Systems. In: Course Term Paper, Goteborg University, Graduate School of Language Technology, Sweden, Available at: http://www.speech.kth.se/~rolf/gslt_papers/GabrielSkantze.pdf, accessed on 12/11/2005.

[Stern 1996] R. M. Stern: Robust Speech Recognition. In: Survey of the State of the Art in Human Language Technology, R. A. Cole, Eds, Cambridge University Press.

[Suhm 1996] B. Suhm, B. Myers and A. Wailel: Interactive Recovery from Speech Recognition Errors in Speech User Interfaces. In: Proceedings of International Conference on Spoken Language Processing (ICSLP 1996), Philadelphia, PA, USA, Page 861-864.

[Suhm 1999] B. Suhm, B. Myers and A. Waibel: Model-Based and Empirical Evaluation of Multimodal Interactive Error Correction. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: the CHI is the limit, Pittsburgh, Pennsylvania, USA, Page 584 - 591.

[Suhm 2001] B. Suhm, B. Myers and A. Waibel: Multimodal Error Correction for Speech User Interfaces. In: ACM Transactions on Computer-Human Interaction, Volume 8, Issue 1, Page 60 - 98.

[Svendsen 1995] T. Svendsen, F. K. Soong and H. Purnhagen: Optimising Baseforms for HMM-Based Speech Recognition. In: Proceedings of the European Conference on Speech Communication and Technology (Eurospeech 1995), Madrid, Spain, Page 783-786.

[Tan 2003] Y. K. Tan, N. Sherkat and T. Allen: Is Speech Enough? We can use eye too! In: Proceedings of the Applied Voice Input/Output Society (AVIOS 2003), San Jose, California, USA.

[Tang 2005] M. Tang: Large Vocabulary Continuous Speech Recognition Using Linguistic Features and Constraints. PhD Thesis, MIT, USA.

[Torre 2002] A. Torre, A. Peinado, A. Rubio, J. Segura and C. Benitez: Discriminative Feature Weighting for HMM-Based Continuous Speech Recognisers. In: Journal of Speech Communication, Volume 38, Issue 3-4, Page 267-286.

[Trauboth 1996] H. Trauboth: Software Quality, Published by Oldenbourg, Germany.

[Turunen 2001A] M. Turunen and J. Hakulinen: Agent-Based Adaptive Interaction and Dialogue Management Architecture for Speech Applications. In: Proceedings of the 4th International Conference on Text, Speech and Dialogue (TSD 2001), Zelezna Ruda, Czech Republic, Page 357-364.

[Turunen 2001B] M. Turunen and J. Hakulinen: Agent-Based Error Handling in Spoken Dialogue Systems, In: Proceedings of the European Conference on Speech Communication and Technology (Eurospeech 2001), Aalborg, Denmark, Page 2189-2192.

[Turunen 2005] M. Turunen, J. Hakulinen, K. J. Raiha, E. P. Salonen, A. Kainulainen and P. Prusi: An Architecture and Applications for Speech-Based Accessibility Systems. In: IBM Systems Journal, Volume 44, No. 3, Page 485-504.

[Umeda 1975] N. Umeda and R. Teranishi: The Parsing Program for Automatic Text-to-Speech Synthesis Developed at the Electrotechnical Laboratory in 1968. In: IEEE Transactions on Acoustics, Speech, and Signal Processing, Page183-188.

[Venkatagiri 2002] H. S. Venkatagiri: Speech recognition technology applications in communication disorders. In: American Journal of Speech-Language Pathology, Volume 11, Page 323-332.

[VoconGames 2005] VoconGames: Speech-enabled Games. Available at: http://www.scansoft.com/speechworks/vocon/games/gallery/, accessed on 12/11/2005.

[VoiceXML Forum 2005] VoiceXML Forum: Current Members. Available at: http://www.voicexml.org/member_companies.asp, accessed on 12/11/2005.

[W3C 2000A] W3C: Introduction and Overview of W3C Speech Interface Framework. Available at: http://www.w3.org/TR/voice-intro/, accessed on 12/11/2005.

[W3C 2000B] W3C: Voice extensible Markup Language (VoiceXML) version 1.0. Available at: http://www.w3.org/TR/2000/NOTE-voicexml-20000505/, accessed on 12/11/2005.

[W3C 2004] W3C: Voice extensible Markup Language (VoiceXML) version 2.0. Available at: http://www.w3.org/TR/2004/REC-voicexml20-20040316/, accessed on 12/11/2005.

[Wachowic 2003] M. Wachowicz and G. Hunter: Spatial Data Usability. In: Data Science Journal (Spatial Data Usability Special Section), Volume 2, Page 75-78.

[Walker 1998] M. A. Walker, J. Fromer, G. D. Fabbrizio, C. Mestel and D, Hindle: What can I say?: Evaluation a Spoken Language Interface to Email. In: Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI 1998), Los Angeles, California, USA, Page 182-589.

[Walker 2000] M. A. Walker, I. Langkilde, J. Wright, A. Gorin and D. Litman: Automatic Prediction of Problematic Human-Computer Dialogues in How May I Help You?" In: Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU 1999), Colorado, USA, Page 210-217.

[Walker 2002] M. A. Walker, I. Langkilde-Geary, H. W. Hastie, J. Wright, and A. Gorin: Automatically Training a Problematic Dialogue Predictor for a Spoken Dialogue System. In: Journal of Artificial Intelligence, Volume 16, Page 293-319.

[Walker 2004] W. Walker, P. Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf, and J. Woelfel (2004): Sphinx-4: A Flexible Open Source Framework for Speech Recognition. In: Technical report, Sun Microsystems Inc, Report Number: TR-2004-139.

[Wang 1995] J. Wang: "Integration of Eye-gaze, Voice and Manual Response in Multimodal User Interface. In: Proceeding of IEEE International Conference on Systems, Man and Cybernetics (ICSMC 1995), Vancouver, British Columbia, Canada, Page 3938-3942.

[Weber 2003] K. Weber: HMM Mixtures (HMM2) for Robust Speech Recognition. In: PhD thesis, Swiss Federal Institute of Technology Lausann, Switzerland.

[Weiser 1996] M. Weiser and J. Brown: Designing Calm Technology. In: Journal of PowerGrid, Volume 1, No. 1, Available at: http://www.ubiq.com/hypertext/weiser/calmtech/calmtech.htm, accessed on 12/11/2005.

[White 2004] K. White, H. Ruback and R. Sicconi: Is There a Future for Speech in Vehicles? In: Speech Technology Magazine (November / December 2004), Available at: http://www.speechtechmag.com/issues/9_6/cover/11181-1.html, accessed on 12/11/2005.

[Whiteside 1998] J. Whiteside, J. Bennett and K. Holzblatt: Usability Engineering: Our Experience and Evolution. In: Handbook of Human-Computer Interaction, M. Helander, Eds., Elsevier Science Publishers, Amsterdam, Page 791-817.

[Wikipedia 2005] Wikipedia: Speech Recognition. Available at: http://en.wikipedia.org/wiki/Speech_recognition, accessed on 12/11/2005.

[Wilpon 1996] J. G. Wilpon and C. N. Jacobsen: A Study of Speech Recognition for Children and Elderly. In: Proceedings of International Conference on Acoustics, Speech and Signal Processing (ICASSP 1996), Atlanta, USA, Page 349-352.

[Yankelovich 1998] N.Yankelovich: Using Natural Dialogues as the Basis for Speech Interface Design. In: Automated Spoken Dialog Systems, Published by MIT Press, USA.

[Young 1997] S. Young, M. Adda-Dekker, X, Aubet, C. Dugast, J. L. Gauvain, D. J, Kershaw, L, Lamel, D. A. Leeuwen, D. Pye, A. J. Robinson, H. J. M. Steeneken and P. C. Woodland: Multilingual Large Vocabulary Speech Recognition: the European SQALE project. In: Journal of Computer Speech and Language, Volume 11, Page 73-89.

[Yun 1999] S. J. Yun and Y. H. Oh: Stochastic Lexicon Modeling for Speech Recognition. In: IEEE Signal Processing Letters, Volume 6, No. 2, Page 28-30.

[Yuen 2002] P. Yuen, Y. Tang and P. Wang: Multimodal Interface for Human-Machine Communication. Published by World Scientific Publishing Co., Inc., River Edge, NJ, USA.

[Zhao 2004A] B. Zhao, T. Allen and A. Bargiela: Usability Evaluation of a Directed-Dialogue Speech-Enabled Query Interface for the ATTAIN Travel Information System. In: Proceedings of 5th International Conference on Recent Advances in Soft Computing, Nottingham, UK, Page 554-559.

[Zhao 2004B] B. Zhao, T. Allen and A. Bargiela: Evaluation of a Mixed-Initiative Dialogue Multimodal Interface. In: Applications and Innovations in Intelligent System XII, A. Macintosh, R. Ellis and T. Allen Tony, Eds., Springer, UK, Page 265-278.

[Zhou 2004] B Zhou, D. Dechelotte and Yuqing Gao: Two-way Speech-to-Speech Translation on Handheld Devices. In: Proceedings of International Conference of Spoken Language Processing (ICSLP 2004), Korea, Page 1637-1640.

[Zipf 1949] G. K. Zipf: Human Behaviour and the Principle of Least Effort. In: An Introduction to Human Ecology, Published by Addison-Wesley Press Inc., Cambridge Massachusetts, USA.

Appendix 1: The Questionnaire of Usability Test

(Text Message interface)
Please circle one of the numbers for each question.

Question 1
This system is

1 Very boring	2	3 Boring	4	5 Plain	6	7 Interesting	8	9 Very interesting

Question 2
The system is easy to learn

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 3
The system is easy to use

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 4
I am able to use the system for a second time without referring to the instruction sheet.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 5
This system is very accurate.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 6
I often made mistakes when using this system.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 7
This system is very natural.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 8
I enjoyed using the system.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

(Speech interface)

Please circle one of the numbers for each question.

Question 1
This system is

1 Very boring	2	3 Boring	4	5 Plain	6	7 Interesting	8	9 Very interesting

Question 2
The system is easy to learn

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 3
The system is easy to use

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 4
I am able to use the system for a second time without referring to the instruction sheet.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 5
This system is very accurate.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 6
I often made mistakes when using this system.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 7
This system is very natural.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

Question 8
I enjoyed using the system.

1 Strongly disagree	2	3 Disagree	4	5 Neither agree or disagree	6	7 Agree	8	9 Strongly agree

(Preference)

Please circle one of the numbers for each question.

Question 1
Which system would you prefer to use to get a journey plan?

Text Message	Speech

You like the above system because:
(Please circle YES or NO)

Question 1.1
the system is innovative and interesting? YES NO

Question 1.2
the system allows you get result quickly? YES NO

Question 1.3
the system is very accurate? YES NO

Question 1.4
you feel natural using it compared to using the others system? YES NO

Question 1.5
Other Reasons?

If you did not like using the voice interface is it because:
(Please circle YES or NO)

Question 1 the system too slow?	YES	NO
Question 2 the system is not very accurate?	YES	NO
Question 3 you are not used to using such system to get information?	YES	NO
Question 4 the system is boring?	YES	NO
Question 5 the dialogue is inflexible?	YES	NO
Question 6 the system output is unclear?	YES	NO
Question 7 you can not remember the enquiry result?	YES	NO
Question 8 the system is not natural?	YES	NO

Question 9
Other Reasons?

Appendix 2: The Canonical WAVE File Format

File Offset	Field Name	Field Size (Bytes)	Chunk Descriptor
4	ChunkID	4	The format of concern here is "WAVE", which requires tow sub-chunks: "fmt" and "data".
8	ChunkSize	4	
12	Format	4	
16	Subchunk1 ID	4	This chunk describes the format of the sound information in the data sub-chunk.
20	Subchunk1 Size	4	
22	AudioFormat	2	
24	NumChannels	2	
28	SampleRate	4	
32	ByteRate	4	
34	BlockAlign	2	
36	BitsPerSample	2	
40	Subchunk2 ID	4	This Chunk indicates the size of the sound information and contains the raw sound data.
44	Subchunk2 Size	4	
...	Data	Subchunk2 Size	

Offset	Size	Name	Description
0	4	**ChunkID**	Contains the letters "RIFF" in ASCII form (0x52494646 big-endian form).
4	4	**ChunkSize**	36 + SubChunk2Size, or more precisely: 4 + (8 + SubChunk1Size) + (8 + SubChunk2Size) This is the size of the rest of the chunk following this number. This is the size of entire file in bytes minus 8 bytes for the two fields not included in this count: ChunkID and ChunkSize.
8	4	**Format**	Contains the letters "WAVE" (0x57415645 big-endian form).
12	4	**Subchunk1ID**	Contains the letters "fmt " (0x666d7420 big-endian form).
16	4	**Subchunk1Size**	16 for PCM. This is the size of the rest of the Subchunk which follows this number.
20	2	**AudioFormat**	PCM = 1 (i.e. Linear quantization) Values other than 1 indicate some form of compression.

22	2	**NumChannels**	Mono = 1, Stereo = 2, etc.
24	4	**SampleRate**	8000, 44100, etc.
28	4	**ByteRate**	SampleRate * NumChannels * BitsPerSample/8
32	2	**BlockAlign**	NumChannels * BitsPerSample/8 The number of bytes for one sample including all channels.
34	2	**BitsPerSample**	8 bits = 8, 16 bits = 16, etc.
36	4	**Subchunk2ID**	Contains the letters "data" (0x64617461 big-endian form).
40	4	**Subchunk2Size**	NumSamples * NumChannels * BitsPerSample/8 This is the number of bytes in the data. You can also think of this as the size of the read of the subchunk following this number.
44	*	**Data**	The actual sound data.

Glossary:

ANN	Artificial Neural Networks
ASP	Active Server Page
ATTAIN	Advance Traffic and Travel Information System
CD	Compact Disc
CGI	Common Gateway Interface
CTW	Computer and Telephone-Accessed WWW System
DIME	Distributed Memory Environment
DTI	Dialogic Telephony Interface
DTMF	Dual Tone Multi-Frequency
GPS	Global Positioning Systems
GSL	Grammar Specification Language
HCI	Human Computer Interactions
HPML	Hyper Phone Markup Language
HMM	Hidden Markov Models
IBM	International Business Machines
IM	Instant Messaging
ISDN	Integrated Services Digital Networks
JSP	JavaServer Page
MIT	Massachusetts Institute of Technology
MMS	Multimedia Messaging Service
OOV	Out Of Vocabulary
PDA	Personal Digital Assistant
PRI	Primary Rate Interface
SDK	Software Development Kit
SMS	Short Messaging Service
VoiceXML	Voice eXtensible Markup Language
W3C	The World Wide Web Consortium
WWW	World Wide Web