Nelson Mandela Metropolitan University

for tomorrow

Department of Computing Sciences

A Model for Mobile, Context-Aware In-Car Communication Systems to Reduce Driver Distraction

Patrick Tchankue-Sielinou

Submitted in fulfilment of the requirements for the degree of Philosophiae Doctor in the Faculty of Science at the Nelson Mandela Metropolitan University

April 2015

Promoter: Prof. Janet Wesson

Co-Promoter: Dr Dieter Vogts

Declaration

I, Patrick Tchankue-Sielinou (Student Number 208022767), hereby declare that, in accordance with Rule G4.6.3 of the Nelson Mandela Metropolitan University, this thesis is my own work and that it has not previously been submitted for assessment to another University or for another qualification.

Signature: __________________________

This thesis is dedicated in loving memory to:

My father Andre Sielinou

Acknowledgments

Firstly, I would like to thank my promoters, Prof Janet Wesson and Dr Dieter Vogts, for their support, direction and encouragement during the course of this research. Their input, advice and ideas have made an invaluable contribution to this thesis.

I would like to thank the Nelson Mandela Metropolitan University (NMMU)/Telkom Centre of Excellence for the funding and equipment, which have made this research possible. NMMU also provided valuable funding, which supported this research.

I would also like to thank the staff and postgraduate students of the Department of Computing Sciences for creating a pleasant and enjoyable atmosphere in which to do this research.

Finally, I would like to thank my family for their staunch support throughout the course of this research.

Summary

Driver distraction remains a matter of concern throughout the world as the number of car accidents caused by distracted driving is still unacceptably high. Industry and academia are working intensively to design new techniques that will address all types of driver distraction including visual, manual, auditory and cognitive distraction. This research focuses on an existing technology, namely in-car communication systems (ICCS). ICCS allow drivers to interact with their mobile phones without touching or looking at them. Previous research suggests that ICCS have reduced visual and manual distraction. Two problems were identified in this research: existing ICCS are still expensive and only available in limited models of car. As a result of that, only a small number of drivers can obtain a car equipped with an ICCS, especially in developing countries. The second problem is that existing ICCS are not aware of the driving context, which plays a role in distracting drivers.

This research project was based on the following thesis statement: *A mobile, context-aware model can be designed to reduce driver distraction caused by the use of ICCS*. A mobile ICCS is portable and can be used in any car, addressing the first problem. Context-awareness will be used to detect possible situations that contribute to distracting drivers and the interaction with the mobile ICCS will be adapted so as to avert calls and text messages. This will address the second problem. As the driving context is dynamic, drivers may have to deal with critical safety-related tasks while they are using an existing ICCS.

The following steps were taken in order to validate the thesis statement. An investigation was conducted into the causes and consequences of driver distraction. A review of literature was conducted on context-aware techniques that could potentially be used. The design of a model was proposed, called the Multimodal Interface for Mobile Info-communication with Context (MIMIC) and a preliminary usability evaluation was conducted in order to assess the feasibility of a speech-based, mobile ICCS. Despite some problems with the speech recognition, the results were satisfying and showed that the proposed model for mobile ICCS was feasible. Experiments were conducted in order to collect data to perform supervised learning to determine the driving context. The aim was to select the most effective machine learning techniques to determine the driving context. Decision tree and instance-based algorithms were found to be the best performing algorithms. Variables such as speed, acceleration and linear acceleration were found to be the most important variables according to an analysis of the decision tree.

The initial MIMIC model was updated to include several adaptation effects and the resulting model was implemented as a prototype mobile application, called MIMIC-Prototype.

A field study was conducted in order to determine if the thesis statement could be proved. An adaptive version of MIMIC-Prototype was compared to a non-adaptive version of MIMIC-Prototype. The results were encouraging and showed that the adaptive version was significantly less distractive than the non-adaptive version. The comparison was done using objective metrics as well as subjective metrics. The usability of both versions of MIMIC-Prototype was found to be satisfactory.

Keywords

Distraction Level, Driver Distraction, Driving Context, Field Study, In-Car Communication System, Machine Learning, Usability Evaluation.

Table of Contents

List of Figures

List of Tables

Abbreviations

Acronyms	Definitions
AAA	American Automotive Association
ACC	Adaptive Cruise Control
AD	Adaptivity
ADAS	Advanced Driver-Assistance System
AIDE	Adaptive Integrated Driver-Vehicle Interface
AMOLED	Active Matrix Organic Light Emitting Diode
ANN	Artificial Neural Networks
API	Application Programming Interface
ARFF	Attribute-Relation File Format
ASR	Automatic Speech Recognition
BN	Bayesian Networks
CD	Compact Disk
CDS	Crashworthiness Data System
CML	Context Modelling Language
CoBrA	Context Broker Architecture
CPT	Conditional Probability Table
CPU	Central Processing Unit
CSV	Comma Separated Values
DAML+OIL	DARPA Agent Mark-up Language + Ontology Interchange Language
dB	Decibel
DC	Driving Context
DC	Distributed Contribution

DE	Driving Event
DL	Distraction Level
DM	Dialogue Manager
DNN	Deep Neural Networks
DOM	Document Object Model
ER	Entity Relationship
FM	Frequency Modulation
FOR	Formality
GMM	Gaussian Mixture Models
GPS	Global Positioning System
GUI	Graphic User Interface
HCI	Human-Computer Interaction
HD	High Definition
HMM	Hidden Markov Model
hPa	hecto Pascal
HTML	Hyper Text Markup Language
IB	Instant Based
ICCS	In-Car Communication Systems
INC	Incompleteness
ITS	Intelligent Transportation System
IVIS	In-Vehicle Information System
Km	Kilometre
Km/h	Kilometres per hours
LCT	Lane Change Task
LMT	Logic Model Tree

MIMI	Multimodal Interface of Info-communication
MIMIC	Multimodal Interface for Mobile Info-communication with Context
MMI	Multimodal Interface
MMS	Multimedia Messaging Service
MRT	Multiple Resource Theory
NFC	Near Field Communication
NHTSA	National Highway Traffic Safety Administration
NLG	Natural Language Generation
NLU	Natural Language Understanding
NMMU	Nelson Mandela Metropolitan University
NN	Neural Networks
OS	Operating System
OWL	Web Ontology Language
PCA	Principal Component Analysis
PTT	Push To Talk
PV	Partial Validation
QUA	Quality
RBFNetwork	Radial Basis Function Network
RDF	Resource Description Framework
REC-H	Research Ethics Committee – Human
RGB	Red Green Blue
SAICSIT	South African Institute of Computer Scientists and Information Technologists
SAX	Simple API for XML
SDK	Software Development Kit

SMO	Sequential Minimal Optimisation
SMS	Short Message Service
STT	Speech-To-Text
SUI	Speech User Interface
SUS	System Usability Scale
SVM	Support Vector Machine
TLX	Task Load Index
TTS	Text-To-Speech
UML	Unified Modelling Language
US	United Stated
USA	United States of America
UV	Ultraviolet
VICS	Vehicle Information and Communication System
W3C	World Wide Web Consortium
WEKA	Waikato Environment for Knowledge Extraction
WSDL	Web Service Description Language
XML	Extensible Mark-up Language

Chapter 1: Introduction

1.1. Background

People spend a lot of time in their cars. Studies have shown that people use their mobile phones while driving, which can be useful because they can make use of this time to communicate with friends, family and colleagues. However, this behaviour can cause driver distraction. Driving is a complex task, because the driver has to deal with several internal and external factors simultaneously. Studies classify the driving task into three levels: strategic, tactical and operational (Michon, 1985). On the strategic level, the driver merely plans the trip, the tactical level is influenced by traffic rules and other users of the road, while the operational level deals with the action control of the car. Drivers switch between these levels in order to operate their vehicles. A secondary task may therefore divert the driver's attention from the primary task, which is driving.

Driver distraction occurs when the driver is no longer able to control the car safely due to an external event. This phenomenon is unfortunately the cause of many car accidents worldwide (Hoff, Grell, Lohrman *et al.*, 2013), some of which are fatal to drivers and passengers. This external event can be the use of a mobile phone or mobile device. The interaction between drivers and an in-car application can be facilitated by interfaces using non-traditional modalities. Multimodal interfaces make use of more than one modality (speech, gesture) in order to interact with users. Multimodal interfaces have been extensively used to design in-car interfaces as studies show that they can mitigate the effects of driver distraction (Becker, Blaylock, Gerstenberger *et al.*, 2006, Tchankue, Vogts & Wesson, 2010a).

Several car manufacturers have introduced in-car communication systems (ICCS) as part of infotainment systems, including navigation and entertainment systems. These ICCS aim to reduce driver distraction caused by the use of mobile phones whilst driving. ICCS are synchronised with the mobile phone by using Bluetooth and can facilitate hands-free and eyes-free communication. Some of the functions of these systems include the following: making and receiving calls, sending and reading text messages. Some examples of recently introduced ICCS include Honda Connect (NVIDIA, 2014), IQon from SAAB (SAAB, 2011) and UVO from KIA Motors (KIA, 2010, KIA, 2014). Some older ICCS such as the Ford SYNC (Ford, 2008), BMW ConnectedDrive (BMW, 2013) and the BMW iDrive (BMW, 2009) are constantly being updated to meet the desired requirements.

Talking on a mobile phone while driving is more distracting than having a conversation with a passenger (Jenness, Lattanzio, O'Toole *et al.*, 2002). This is due to the fact that the passenger is aware of the current driving context and can adapt the conversation according to the difficulty of the driving task. This is one of the reasons that led to the introduction of adaptive interfaces for automotive applications. The interface adapts the interaction between the driver and the application according to the current driving context. Adaptive user interfaces depend on adaptation factors, adaptation mechanisms and adaptation effects (Lavie & Meyer, 2010, Tchankue *et al.*, 2010a). Adaptation factors are the actual events that should be followed by an adaptation, adaptation mechanisms are the processes and techniques (e.g. automatic or manual) used for adaptation and the adaptation effects are the actions taken by the system. These elements provide a basis for determining the risks in the current driving situation. Driving is dynamic; drivers go through several driving contexts which require their full attention. In-car applications that are context-aware could help in preventing driver distraction.

ICCS can help in reducing driver distraction but some issues still have to be investigated. The following paragraph discusses the problems that will be solved in this research project.

1.2. Problem Statement

Existing ICCS only partially address the critical issue of driver distraction (Tchankue, Wesson & Vogts, 2011). Most ICCS are multimodal and use speech as the primary input modality. Modes such as manual (steering wheel buttons), haptic, gesture, monitor and touch screen are also used. Multimodality addresses manual distraction as the driver does not need to manipulate the mobile handset manually. However, most ICCS are embedded into the car and cannot be moved from one car to another. This reduces the availability of hands-free and eyes-free communication in cars. The lack of intelligence or adaptation to the current driving context is also identified as a major shortcoming of several ICCS (Tashev, Seltzer, Ju *et al.*, 2009).

Although the use of mobile phones whilst driving is illegal in most countries, people are still tempted to use their mobile devices in driving situations. This dangerous behaviour can lead to car accidents. However, in-car communication using eyes-free and hands-free devices is allowed whilst driving in some countries.

The problem statement of this research is the following: *Existing ICCS do not support the context of the driver. In addition, most ICCS are not available in all types of cars.*

The following paragraph discusses the thesis statement that will be used to guide this research project.

1.3. Thesis Statement

Interaction between a driver and an ICCS can cause driver distraction. Several studies show that this distraction can be mitigated by using adaptive user interfaces (Jenness *et al.*, 2002). ICCS can provide a partial solution to driver distraction and the number of cars shipped with such technology is increasing. However, most drivers do not use ICCS yet, especially in developing countries. This is due to the fact that most ICCS come as an optional extra, which is often not affordable to drivers, especially younger drivers and those in developing countries. Meanwhile several drivers including young drivers own a mobile phone from which they can download applications. Mobile phones are equipped with several sensors that may help in determining the driving context. The determination of the driving context should be as simple as possible so that the use of a mobile phone will be sufficient for this research.

This research is based on the following thesis: *A mobile, context-aware model can be designed to reduce driver distraction caused by the use of ICCS*

The following paragraph discusses the research questions to be answered in order to prove the thesis statement.

1.4. Research Questions

Research questions are used to state what the study will attempt to find out (Hofstee, 2006). These questions are derived from the thesis statement. The objective is to split the study into manageable parts that will be addressed in specific sections.

Following from the thesis statement, the primary research question to be addressed by this research is: *How should a model for a mobile, context-aware ICCS be designed to reduce driver distraction?*

#	Research Questions	Method (s)	Chapter (s)
RQ1	What are the causes and effects of driver distraction?	Literature review	2
RQ2	What are the existing models for context-aware applications?	Literature review	3
RQ3	What should a model for a speech-based, mobile ICCS comprise?	Iterative design	4
RQ4	Is a speech-based, mobile ICCS feasible?	Usability study	4
RQ5	How can an inference engine be designed for a mobile ICCS?	Iterative design	5
RQ6	What are the most efficient techniques to determine the driving context?	Experiment	5
RQ7	How can an algorithm be developed to determine if the driving situation is safe?	Iterative design	6
RQ8	How can adaptation effects be implemented to reduce driver distraction?	Iterative design	6
RQ9	To what extent does a mobile, context-aware ICCS reduce driver distraction?	Field study	7

Table 1.1: Research Questions to be addressed in this Thesis

Table 1.1 summarises the secondary research questions that will help in answering the main research question. Research questions 1 and 2 aim to review the literature in order to find what other scholars have found about driver distraction and context-aware applications. An iterative design method will be used to answer research question 3. When using an iterative design method, several steps (analysing, prototyping and testing) are repeated until a prototype that meets the requirements is obtained. The answer to research question 4 will determine whether a speech-based ICCS is feasible, in terms of usability. Research question 5 addresses the proposal of a model that will be used to infer the driving context. Research question 6 will be answered by analysing the results of several experiments. Research question 7 will be addressed by developing an algorithm to determine if a driving situation is safe. Research question 8 will be answered by developing an algorithm that can be used to adapt the communication between the ICCS and the driver. Finally, research question 9 will be answered by determining the benefits that can be gained from an integrated mobile, context-aware ICCS using a field study.

1.5. Research Objectives

The primary aim of this research is to design a model that will reduce driver distraction by using a mobile, context-aware ICCS. In order to achieve this aim, the following objectives were identified:

- To identify the causes and effects of driver distraction (Chapter 2),
- To define and review existing models for context-aware applications (Chapter 3),
- To design a model for a speech-based, mobile ICCS (Chapter 4),
- To investigate the feasibility of a speech-based, mobile ICCS (Chapter 4),
- To design an inference engine that can be used to determine the driving context (Chapter 5),
- To select a machine learning technique that can determine the driving context (Chapter 5),
- To design a context adapter that can be used to reduce driver distraction (Chapter 6),
- To identify and implement adaptation effects that can help in reducing driver distraction (Chapter 6),
- To measure the extent to which the proposed model for a mobile, context-aware ICCS can help in reducing driver distraction (Chapter 7).

These objectives are aligned with the research questions proposed in Table 1.1. Each research objective will be used to provide an answer to its related research question.

1.6. Scope

This research will be limited to the domain of mobile, context-aware ICCS. The interaction between the driver and the system will be made possible with a speech-based user interface to prevent drivers from using their hands and eyes. This will help in preventing manual and visual distraction. Some studies (Healey & Picard, 2004, Nasoz, Lisetti & Vasilakos, 2010) aiming to determine the driving context make use of physiological sensors in order to measure variables such as the perspiration and the heart rate of the driver. This methodology will not be used in this research because this project aims to propose a simple, portable and inexpensive model to address the problem. Other studies (You, Lane, Chen *et al.*, 2013) have made use of the front and rear cameras of the mobile phone to monitor the driver's behaviour. This thesis will only use the mobile phone sensors and web services in order to monitor the driver's context.

Reducing driver distraction using techniques, such as sending calls to voice mail when the car is moving, also affects the mobile phones of passengers. Some research projects have been conducted in order to distinguish between the mobile phone of the driver and the passenger (Yang, Sidhom, Chandrasekaran *et al.*, 2012, Bo, Jian, Li *et al.*, 2013, Wang, Yang & Liu, 2013, Chu, Raman, Shen *et al.*, 2014). In this research, there is no attempt to distinguish the driver from a passenger who owns a phone running an implementation of the proposed model. The assumption is made that the owner of the phone, which runs the context-aware ICCS, is the driver. Some work can be added in order to make sure that the owner of the phone is the driver, before activating the context-aware ICCS.

A set of methods is needed to conduct this research. The following sections discuss various methods that will be used throughout the course of this research project.

1.7. Research Method

The research method provides a general plan of how a researcher can attempt to answer research questions (Saunders, Lewis & Thornhill, 2009). Figure 1.1 illustrates an overview of research philosophies, approaches and strategies used in scientific research. This is called the research "onion", and serves to highlight the relationships between different research philosophies, approaches and strategies. The following paragraphs discuss the philosophies to be followed during this research and the research approach and techniques.

1.7.1 Research Philosophy

The first concept highlighted in the "onion" is research philosophy. Research philosophy refers to the development of knowledge and the nature of the knowledge developed (Saunders *et al.*, 2009). This study will mainly follow the positivist research philosophy. Positivist research aims to provide evidence from formal propositions, quantifiable measures of variables, hypothesis testing and finally, drawing of inferences from a representative sample of a population (Klein & Myers, 1999).

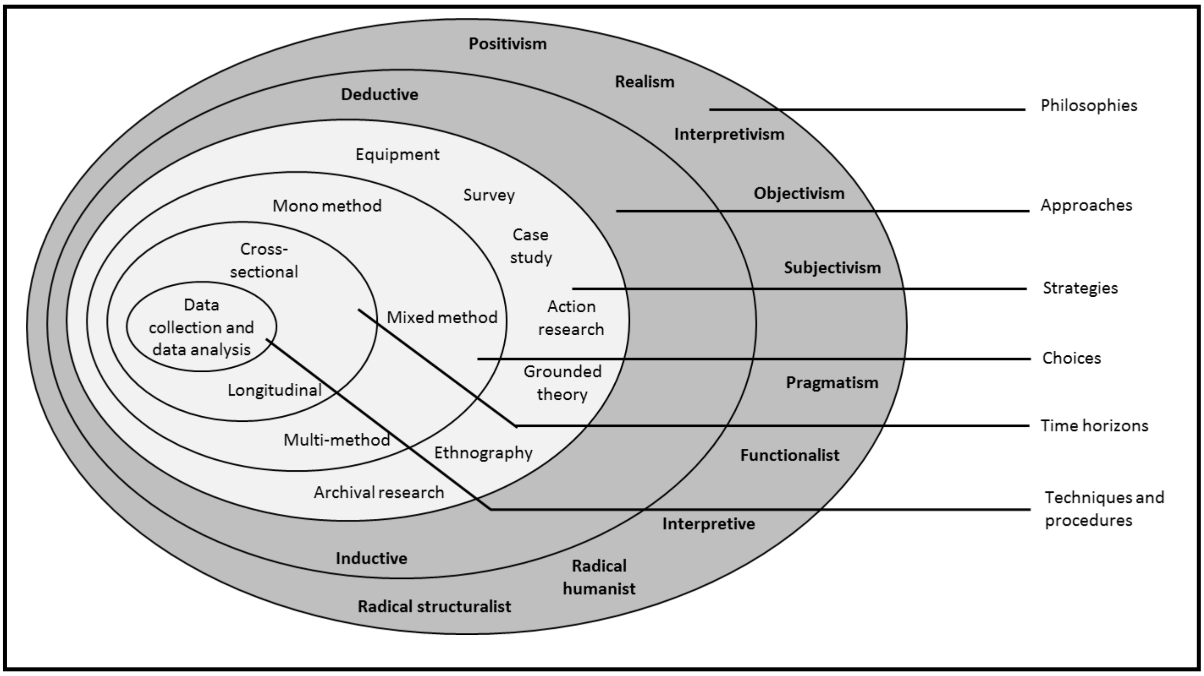


Figure 1.1: Research Onion (Saunders, Lewis & Thornhill, 2003)

The test of the hypothesis will require the design and implementation of a proof of concept (prototype). The interpretivist philosophy will be used in order to interpret the meaning of the behaviour of the users whilst using the prototype, and to analyse comments made by participants after the experiments.

This study can be classified in the field of human-computer interaction (HCI), which is a cross-disciplinary research field. It involves several other disciplines, namely, human factors, ergonomics, computer science, psychometrics, behavioural psychology and cognitive psychology (Hartson, 1998). The accuracy of the context-aware technique used will be measured using a positivist paradigm while the effects of the user interface on the users will be assessed using the interpretivist approach.

1.7.2 Research Approach

Inductive reasoning applies to situations where specific observations or measurements are made towards developing broader conclusions, generalizations and theories. While deductive approaches are based on scientific principles, data are collected with tools based on theory and there is a need to generalise the conclusions of the study. In other words "*induction*" is the creation of a theory, whereas deduction is the process of testing a theory (Saunders *et al.*, 2009).

Deduction involves the development of a theory that is subjected to a test. Deduction is the prevailing research approach in the natural sciences where laws present the basis of explanation, allow the anticipation of phenomena, predict their occurrence and therefore permit them to be controlled (Collis & Hussey, 2009). It is often advantageous to combine both of these methods (Saunders *et al.*, 2009). Although this study will be mostly deductive, there will be some aspects that are inductive.

This research will use both the inductive and the deductive research approaches. The deductive approach will be mainly used because the goal of the study is to show whether the thesis statement is true or false. Facts provided by the user study, experiments and the field study will be used to prove the thesis statement. Prototypes built from the proposed model will be implemented and evaluated in order to show whether the thesis statement can be accepted. The inductive approach will be used when analysing the comments of the participants in order to infer some useful knowledge from these comments.

1.7.3 Research Approach

Research strategy is defined as a general plan that is used to answer the research questions. The choice of a good strategy is driven by a set of objectives addressing the research questions. Possible data collection sources and research constraints are explored in the strategy.

The thesis statement of this research contains the main hypothesis, which is that driver distraction can be reduced by using a mobile, context-aware ICCS. The research will consist of an exploration of the available techniques and literature on driver distraction. Following the positivist research paradigm, a model will be designed using the most suitable techniques identified. A prototype mobile application will be developed as a proof-of-concept. Several user studies and experiments will be conducted in order to test the thesis statement.

The next two chapters review the literature in order to have a better understanding of the problems (Chapter 2) and to explore possible solutions (Chapter 3). A laboratory study will be conducted in Chapter 4 in order to find out if a speech-based, mobile ICCS is feasible. Usability and workload metrics will be used to assess the feasibility of using a speech-based, mobile ICCS. Some experiments not involving the driver will be conducted in order to collect driving data that will be used to identify the most effective machine learning techniques to determine the driving context (Chapter 5). Data collected will be split into two sets: the training set and the testing set to evaluate the effectiveness of each technique. The effectiveness of each technique will be measured using the percentage of correctly classified instances from the testing set.

In Chapter 6, some features that will assist with the adaptation of the mobile, context-aware ICCS will be added to the initial model. This helped in building the final version of the prototype that includes the automatic detection of the driving context. In Chapter 7, a field study involving drivers was conducted. The data captured facilitates measuring the usability, the user experience and the accuracy of the proposed model. This field study compared adaptive and a non-adaptive versions of the prototype. This was done to find out whether applying adaptation to the driving context can reduce the level of driver distraction. User comments and suggestions were collected in order to obtain a better understanding of the field study (interpretivist research paradigm). The level of driver distraction was measured by using the following metrics:

- *Mental workload*: The mental demand of the tasks,
- *Physical workload*: The physical demand of the tasks,
- *Temporal workload*: The pace of the tasks,
- *Performance*: The success in accomplishing the tasks,
- *Effort:* The work put into the accomplishment of the level of performance,
- *Frustration:* The insecurity, annoyance, and irritation experienced whilst completing the tasks,
- *Ease of use of the prototype:* The extent to which users can complete the tasks easily,
- *Efficiency of the prototype*: The ability to perform tasks with speed and precision,
- *Effectiveness*: The ability to successfully achieve every task,
- *Distraction level*: the distraction level predicted by the system when completing tasks,

- *Effectiveness in determining safe situations*: the ability to determine safe situations accurately,
- *Effectiveness in implementing adaptation effects:* the ability to apply the appropriate adaptation effects.

Performance metrics are not always sufficient to evaluate the usability of a system. The NASA task load index (TLX) questionnaire (Hart & Staveland, 1988) was used as a tool to measure the mental workload of each participant. The System Usability scale (SUS) post-test satisfaction questionnaire was used to capture subjective usability ratings from users. A logging system implemented in the prototype was used to collect performance data.

1.7.4 Ethical Considerations and Budget

Each experiment performed throughout this study was subject to ethics approval from the Nelson Mandela Metropolitan University (NMMU) Research Ethics Committee - Human (REC-H) committee (see Appendix E). The ethics application included a detailed description of the objectives of the experiment, the methodology, the questionnaires to be completed by participants and the analysis of the data. A consent form was signed by each participant prior to the experiment. This ensured that each person participated willingly in the study and that the experiment did not affect the participants negatively (emotionally or physically).

This research project was carried out with the support of the NMMU/Telkom Centre of Excellence, a research centre at the NMMU. All costs related to this project were covered by this research unit. The cost of this research project included the following:

- Mobile phones equipped with the sensors needed to measure all variables used to assess the driving context accurately; and
- Resources to be used to set up the test environment (petrol costs and data bundles).

A large 42” touch screen available in the Centre of Excellence laboratory was also used for the usability evaluation of the speech-based ICCS.

The following paragraph identifies some envisaged practical and theoretical contributions.

1.8. Envisaged Research Contribution

This research envisaged making the following contributions to the study of mobile, context-aware ICCS:

- A prototype speech-based, mobile ICCS,
- Identification of the usability issues of speech-based, mobile ICCS,
- Experimental results regarding effective machine learning techniques to determine the driving context using mobile phone data,
- A prototype mobile ICCS incorporating machine learning techniques to predict the driving context as well as the adaptation effects to reduce driver distraction,
- Experimental results regarding the evaluation of the prototype mobile, context-aware ICCS,
- Design recommendations regarding the use of the mobile ICCS to reduce driver distraction,
- A model for the design of a mobile, context-aware ICCS to reduce driver distraction.

A model for a speech-based, mobile ICCS will be iteratively developed to evaluate different aspects of the proposed model. The first implementation was used to assess the feasibility of a speech-based, mobile ICCS. The final version of the prototype was used to conduct a comparative field study between the adaptive and the non-adaptive versions of the prototype. The results of this field study were used to identify the benefits of the proposed model in terms of reducing driver distraction. The shortcomings identified were used to propose design recommendations for a mobile, context-aware ICCS.

1.9. Thesis Structure

As depicted by Figure 1.3, this thesis is divided into eight chapters. Each of these chapters answers a specific research question.

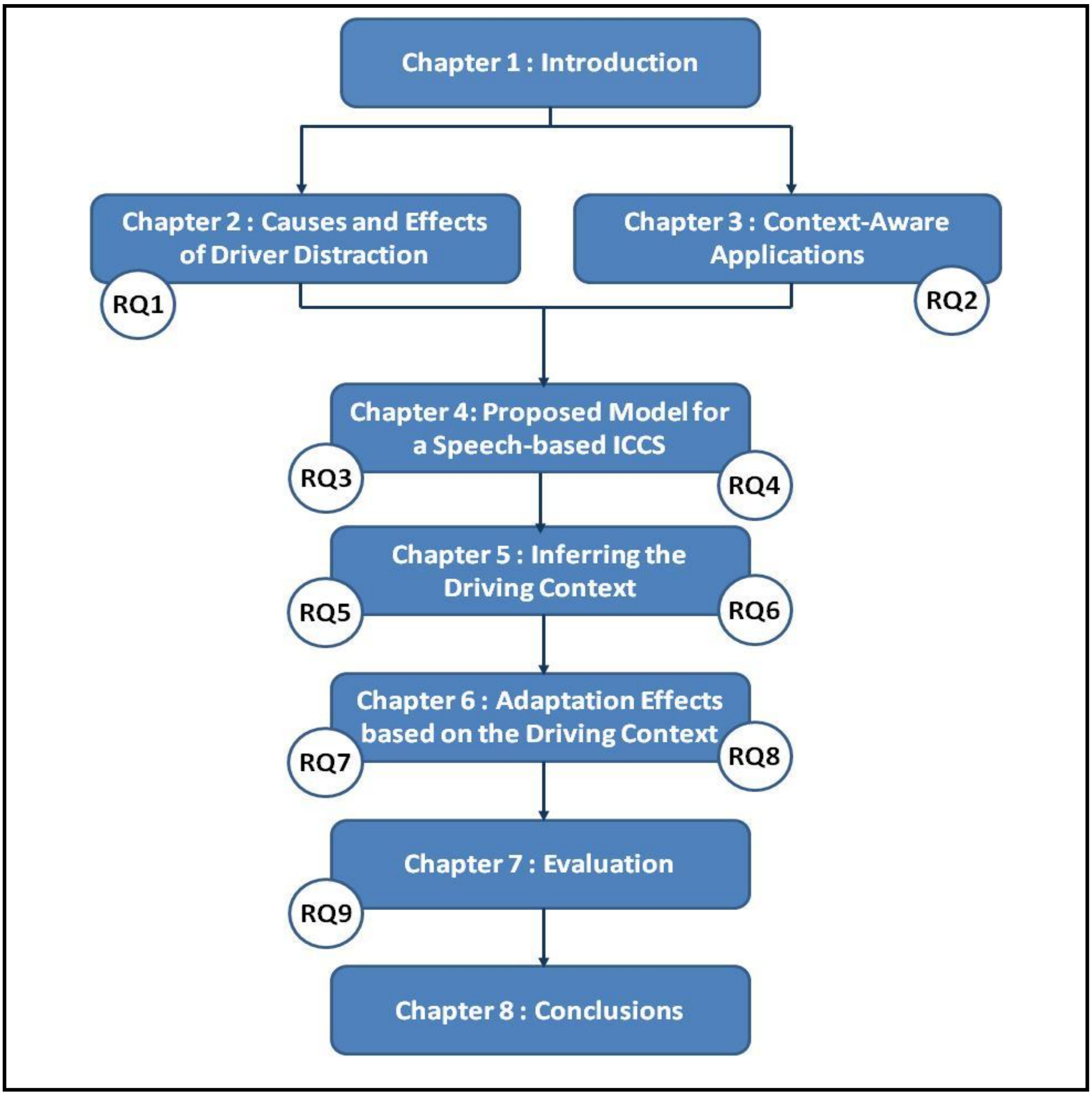


Figure 1.2: Outline of Thesis

Chapter 1 (Introduction) provided the motivation for this research, including a background to the problem that needs to be solved. The research objectives and research methodology were described, which detailed the research questions to be answered.

Chapter 2 (Causes and Effects of Driver Distraction) reviews the literature on ICCS and driver distraction. The causes and effects of driver distraction are highlighted in order to obtain better insight into how to address this issue.

Chapter 3 (Context-Aware Applications) discusses the techniques used to design context-aware applications. These techniques include nearest neighbour, decision tress, artificial neural networks, Bayesian networks, fuzzy logic and queuing networks.

Chapter 4 (Proposed Model of a Speech-based ICCS) discusses the design of a model for a speech-based, mobile ICCS and the evaluation of a prototype application in a laboratory environment. This was done in order to assess the feasibility of speech-based technology.

Chapter 5 (Inferring the Driving Context) discusses the extension of the initial model for a mobile, context-aware ICCS. This was done using an iterative design approach based on the initial model for a speech-based, mobile ICCS proposed in Chapter 4. Experiments were conducted to select effective machine learning techniques for inferring the driving context. The selected techniques were used to determine the driving context, which is comprised of driving events and the perceived level of driving distraction.

Chapter 6 (Adaptation Effects based on the Driving Context) discusses the design and the integration of the complete model for a mobile, context-aware ICCS. An iterative design approach was used to update the model proposed in Chapter 5. The updated model will take into account the adaptation effect and the context sharing.

Chapter 7 (Evaluation) discusses the evaluation of the prototype ICCS in a field study. This prototype was based on the model proposed in Chapter 6. Results are discussed in terms of safety, adaptivity and usability.

Chapter 8 (Conclusions) summarises the work done in this thesis. It highlights the major scientific contributions and the importance of the findings made in this thesis. This is followed by recommendations for future research in the design of models for mobile, context-aware ICCS.

Chapter 2: Causes and Effects of Driver Distraction

2.1 Introduction

The goal of this chapter is to answer the research question *RQ1: What are the causes and effects of driver distraction?* A review of the existing literature will be used as a methodology to identify major usability problems when operating in-car communication system (ICCS) that can cause distraction while driving.

Section 2.2 contains a discussion on ICCS as a technology that enables drivers to keep in touch with the external world whilst they are driving. This is followed by a description of the current driving model (Section 2.3) that aims to understand the complexity of driving a vehicle. The complexity can be increased when the driver engages in a secondary or tertiary activity whilst driving. Section 2.4 provides more details on driver distraction while using mobile phones and the different types of driver distraction that can occur. Section 2.5 investigates the psychological aspects that occur whilst driving and performing multiple tasks, which helps understand the distraction caused by secondary tasks. Section 2.6 discusses several causes of car accidents. Some solutions that have been used to address driver distraction so far are reviewed in Section 2.7, while Section 2.8 focuses on attempts to reduce driver distraction using mobile devices.

2.2 In-Car Communication Systems

Cars have traditionally been a way to transport people safely and comfortably from point A to point B. Over the past decade this picture has changed. The car may also be a place to do business, find friends, access real-time road information, watch movies, download videos, send voice-controlled instant messages or listen to e-mail. The digital lifestyle of this century is being extended to the car by consumers.

Consumer expectations began changing when computing and media went mobile. The iPod provided the '*ultimate'* in mobile media. Suddenly a person could carry over one hundred hours of video and over twenty thousand songs in his/her pocket. This kind of media mobility has created an end-user expectation of a digital lifestyle that extends just about anywhere. Consumers started to express the need for these capabilities in their cars, integrated in a safer and more user-friendly environment.

In-car communication systems (ICCS) are computer systems that can be embedded in the car to achieve many purposes. ICCS are used to manage the communication between the driver and an external contact. By using text-to-speech (TTS) synthesis and speech-to-text (STT) it is now possible to command a system verbally to listen and to respond to voicemail and e-mail messages. Vehicles can even communicate with other vehicles, sharing information on highway alerts and emergency systems, to provide a safer journey. State-of-the-art navigation systems integrate real-time traffic data, personal points of interest and the locations of friends into a new experience; even extending the car into the Web 2.0 era of social networking (e.g. Facebook and Twitter).

Infotainment systems are very useful as they allow a driver to be connected to the external world. The disadvantage of the integration of these systems into cars is the increased amount of information the driver must manage, which can be a cause of driver distraction. Existing ICCSs often include the following sub-systems (Tashev *et al.*, 2009): navigation, car information, safety, entertainment and communication systems.

In-car communication systems (ICCSs) can be designed to address specific problems in the context of a car. The following paragraphs discuss specific type of ICCS.

2.2.1 Navigation Systems

Navigation systems provide directions and points of interest on an electronic map. Free-standing navigation systems use separate microphones and speakers that can interfere with the car's sound system or vice versa.

Figure 2.1 shows the classic system configuration of a car navigation system. The architecture of these systems is similar to the architecture of a typical personal computer. A processor computes data from the volatile memory and displays the corresponding map saved on the hard drive.

Absolute position data from the global positioning system (GPS) receiver is combined with speed signals integrated with time and direction information from a gyroscope to match the current vehicle position on a digital roadmap by using data from pre-recorded media. The system can plan routes to the destination of the driver and can display the routes on the map. Newer systems usually include a receiver for vehicle information and communication system (VICS) broadcasts, thus adding the status of congested roads to the map display. The map scrolls as the car moves and a synthesised voice directs the driver to turn right or left as needed to reach the destination.

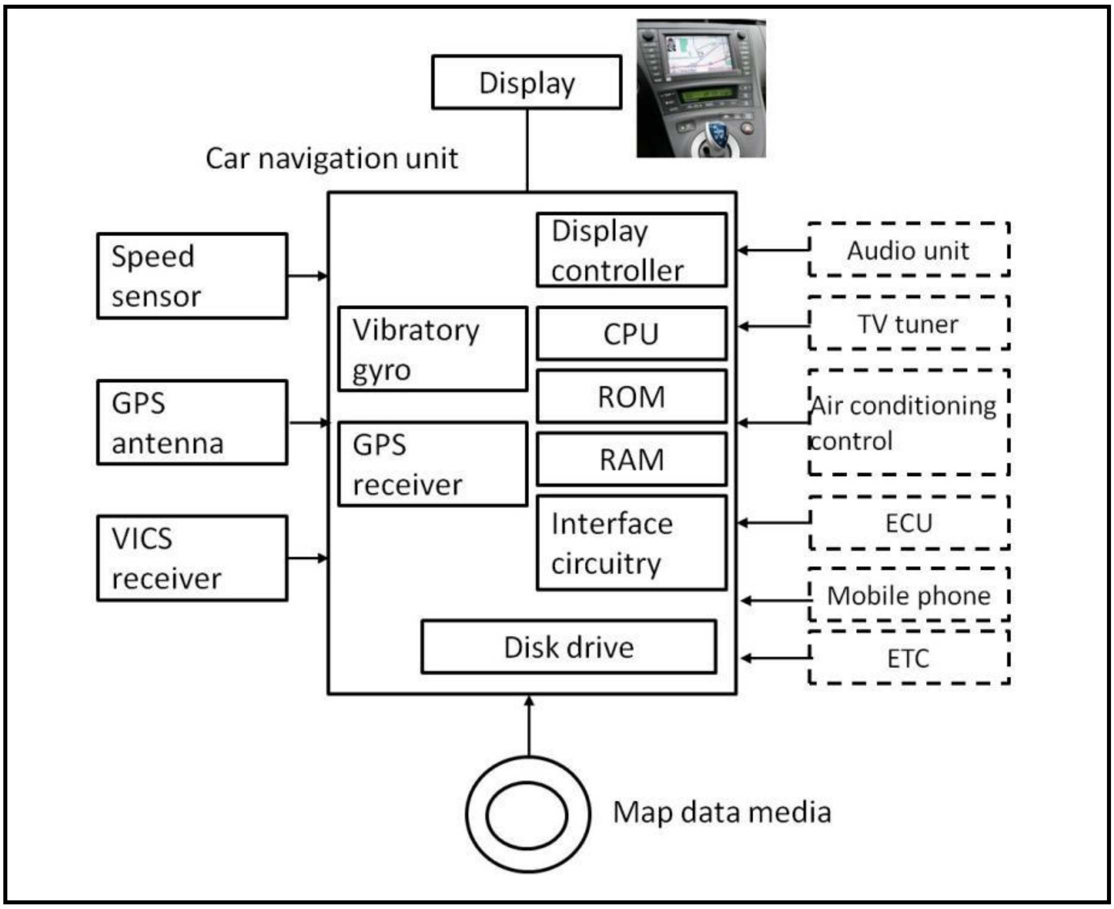


Figure 2.1: Components of a Car Navigation System (Yokouchi, Ideno & Ota, 2000)

Navigation systems use data from the speech sensor, gyroscope and the GPS in order to display the most accurate map to the driver. These data sources could also be used when designing a model for mobile, context-aware ICCS.

2.2.2 Car Information Systems

Car information systems provide a variety of data through various sensors. Modern vehicles are provided with a wide range of built-in sensors that are used to collect data. Information such as weather conditions and traffic conditions can be obtained from a web service (Theodoridis & Koutroumbas, 2009).

A car information system can provide the following: petrol level, climate temperature, traffic conditions, weather conditions, stock values and petrol prices. Most of this information is standard in existing IVIS.

2.2.3 Safety Systems

Today cars are able to send emergency messages to emergency services in the case of an accident. Safety systems use built-in sensors to detect accidents and to evaluate the severity of the accident. Depending on the severity, appropriate services (emergency service or ambulance) are called for help. The OnStar system from General Motors (General Motors, 2003) was one of the first IVIS to include a safety system. Several existing IVIS have included this critical feature in order to enhance the safety of drivers.

2.2.4 Entertainment Systems

Entertainment systems often include music players and radio receivers. Music players read media information on tracks such as title, artist, album, genre and track. This information can be retrieved from a compact disk (CD) or a music device connected to the car such as an iPod or Zune. The radio can be tuned by selecting a radio name or a frequency (BMW, 2009). Toyota Entune is an example of IVIS, which integrates several web services to enable the access to frequency modulation (FM) and satellite radios (Toyota, 2012).

Entertainment systems are often combined with other in-car systems. Infotainment systems combine in-car entertainment and information systems.

2.2.5 In-Car Communication Systems

ICCS enable the use of mobile phones. Information from the address book is retrieved and used to build a proper grammar for the speech recognition engine. This allows the driver to make calls and send text messages efficiently whilst driving using voice commands (Ford, 2008).

Several car manufacturers are introducing ICCS as part of their infotainment systems. These ICCS aim to improve the driver experience and reduce driver distraction caused by the use of mobile phones whilst driving. ICCS are synchronised with the mobile phone using Bluetooth and facilitate hands-free and eyes-free communication.

2.2.6 Example of Existing ICCS

Some examples (Table 2.1) of recently introduced ICCS include IQon from SAAB (SAAB, 2011) and Entune from Toyota. Several user studies have shown that using an ICCS can improve the safety of the driver (Shutko, Mayer, Laansoo *et al.*, 2009). This is one of the reasons that the adoption of ICCS is increasing. For example, Ford Sync has been installed in more than 3 million vehicles since 2007. Sync has evolved to meet the needs of consumers; AppLink and myFord Touch are recent versions of Sync. MyFord Touch uses a touch screen to address the issue of space available on the dashboard for buttons.

IVIS	Manufacturer	Year
COMAND	Daimler Chrysler	1999
iDrive	BMW	2001
Blue&Me	Fiat	2004
SYNC	Ford	2007
UVO	KIA Motors	2011
IQon	SAAB	2011
Entune	Toyota	2012
Modular Infotainment System	Volkswagen	2013

Table 2.1: Examples of ICCS

In addition, KIA Motors introduced a new open source operating system, called UVO. Users can interact with it using an 8" touch screen and it is compatible with Android's mobile phones as well as iPhones. Apple announced the introduction of CarPlay in 2014 (Apple Inc, 2014). CarPlay is an operating system similar to iOS 7.1 that can be synchronised with any iPhone; automotive manufacturers can install it in their car. Similarly, Google started the Open Automotive Alliance, similar to the Open Handset Alliance, which led to the introduction of Android.

These systems are often limited to some models of vehicles and are therefore accessible to a small number of people. Although these systems are mostly speech-based and therefore reduce visual and manual distraction, they do little to address cognitive distraction. The following paragraphs discuss processes that drivers go through when navigating their car.

2.3 The Driving Model

Driving a vehicle is a difficult task. Several researchers have been studying models for a better understanding of the driving task. The most popular is the hierarchical approach proposed by Michon (1985). The hierarchical model (Figure 2.2) divides driving into three levels: the strategic, tactical and operational levels.

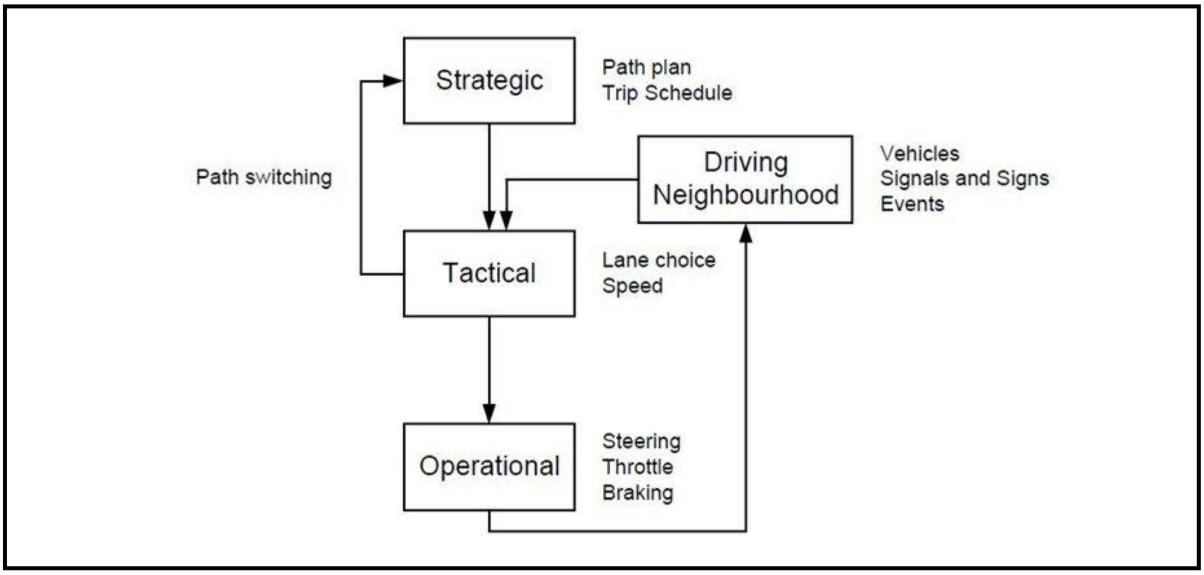


Figure 2.2: Interaction between Driving Task Layers (Toledo, Koutsopoulos & Ben-Akiva, 2003)

2.3.1 Strategic Level

The strategic level is also called the navigating or planning level. This involves route choice and trip schedule decisions that drivers make before or while they are driving. These decisions depend on their familiarity with the transportation network and traffic conditions and also on real-time information, which they may obtain whilst driving.

2.3.2 Tactical Level

The tactical level is responsible for the guidance or maneuvring of the vehicle. It determines the two-dimensional movement of the vehicle in traffic. The lateral dimension changes when drivers negotiate curves, change lanes or overtake other vehicles; the longitudinal dimension is related to the velocity of the car. The tactical level can be affected by both internal and external events. Internal events include the status of the vehicle and the traffic rules that the driver should follow. External events include the different traffic situations (e.g. pedestrians, open road and slippery road), which the vehicle experiences.

2.3.3 Operational Level

The operational level can also be called the stabilisation level. It includes activities performed by drivers to control and direct their cars (e.g. steering, braking and throttling). These activities are based on skills acquired by drivers through experience. Most of these activities are done automatically with little conscious effort. These skills can be reduced significantly by driver distraction.

This model can be used to identify driving situations where drivers need to be focussed on the driving task. This can also guide the selection of participants or the choice of a route in a field study. Participants not familiar with the route could experience driver distraction as a result of not coping on the strategic level (Section 2.3.1). Inexperienced drivers could have some problems on the operational level (Section 2.3.3).

From the previous paragraphs, it can be understood that it is difficult to perform the primary task, which is driving. The following paragraphs discuss different types of distraction that may occur.

2.4 Driver Distraction

Some academic literature uses the terms inattention and distraction interchangeably. Wang, Stutts, Klauer However, driver inattention is a broad concept that includes driver distraction. There is no triggering event when driver inattention occurs; the attentional shift is generated in the mind of the driver. Drowsiness and daydreaming are examples of driver inattention.

Driver distraction occurs when the attention of the driver is, voluntarily or involuntarily, diverted from the driving task to the extent that the driver is no longer able to drive adequately or safely (Young, Regan & Hammer, 2007). Driver distraction can also be defined as a situation in which a driver has chosen to engage in a secondary task that is not necessary to perform the primary driving task (Klauer, Dingus, Neale *et al.*, 2006).

Driver distraction is an intricate phenomenon. Visual, manual, auditory and cognitive distractions are identified as the four types of driver distraction (Ranney, 2008). The types of driver distraction are not mutually exclusive. It is suggested that future studies should focus on measuring combined driver distraction (Basacik & Stevens, 2008). The following subsections will discuss how visual, manual and cognitive distractions occur.

2.4.1 Visual Distraction

Visual distraction is defined as not looking at what is classified as being relevant for driving. It occurs when the driver neglects to look at the road and instead focuses on another visual target, such as an in-car route navigation system or advertisement, for an extended period of time. For example, some drivers can look down at their phone for 4.6 to 6 seconds while typing a text message. If driving at 88 km/h, the car can travel a distance corresponding to the length of a rugby field during this time period (Olson, Hanowski, Hickman *et al.*, 2009).

Three different types of visual distraction exist (Young *et al.*, 2007):

- *Obstructed visual field*: This occurs when the visual field of the driver is blocked by objects, such as stickers on the windscreen or windows of the car or by dark window tints, that prevent him/her from recognising objects or hazards on the road (Ito, Uno, Atsumi *et al.*, 2001),
- *Driver negligence*: This occurs when the driver neglects to look at the road and instead focuses on another visual target, such as an in-car navigation system or advertisement, for an extended period of time,
- *Loss of attention*: This involves a loss of visual "*attentiveness*", often referred to as "*looked, but did not see*" and interferes with the ability of the driver to recognise hazards in the road environment (Ito *et al.*, 2001).

A study focusing on analysing real life data provided by 100 cars showed that it is rare that a crash occurs while the eyes of the driver are on the roadway, regardless of any cognitive task that he/she might be engaged in (Dingus & Klauer, 2008).

2.4.2 Manual Distraction

Manual distraction occurs when a driver removes one or both hands from the steering wheel to manipulate an object instead of focusing on the tasks required to drive safely, such as steering in the appropriate direction or changing gears (Haigney, 1997). This involves some tasks like text messaging. Sending text messages using a hand-held phone whilst driving can cause manual distraction (Hosking, Young & Regan, 2009).

Speech-based ICCSs are designed to solve such problem. The driver is provided with a natural medium for making calls and sending text messages.

2.4.3 Cognitive Distraction

Cognitive distraction occurs when the attention of the driver is absorbed by thoughts to the point where safe navigation through the road network is difficult (Direct Line Motor Insurance, 2002). In this case the loss of attention is triggered by an external event such as a conversation with a passenger. Talking on a mobile phone while driving is well recognised as a form of cognitive distraction; nevertheless cognitive distraction can also take place when talking to a passenger or trying to operate in-vehicle devices such as a mobile phone.

2.4.4 Auditory Distraction

Few papers mention auditory distraction when classifying driver distraction. Nonetheless this type of distraction can be as dangerous as the three previous types. Auditory distraction occurs when sounds prevent drivers from making the best use of their hearing, because their attention has been drawn to whatever caused the sound (Young *et al.*, 2007).

2.4.5 Impact of Driver Distraction

Driver distraction has an impact on driver performance as well as safety. Research has shown that four main factors impact on driver performance and safety (Lee, Young & Regan, 2008b). These factors are driver characteristics, driving task demand, competing task demand and the ability of the driver to self-regulate in response to a competing activity.

- *Driver characteristics*: These comprise age, gender, driving experience, driver state (drowsy, drunk, angry or upset), familiarity with and amount of practice on the competing task, personality (e.g. risk taking and succumbing to peer pressure) and the vulnerability of the driver to distraction (Lee, Young & Regan, 2008a),
- *Driving task demand*: This depends on traffic conditions, road conditions, weather conditions, the number and type of occupants in the vehicle, the ergonomic quality of the cockpit design and vehicle speed (Lee *et al.*, 2008a). The fewer the demands on the driver; the greater will be the residual attention available to attend to competing activities,
- *Demand of the competing task*: This is influenced by factors such as similarity of the task to driving subtasks, its complexity, whether it can be ignored, how predictable it

is, how easily it can be adjusted, how easily its performance can be interrupted and resumed, and how long it takes to perform the task,

- *Ability of the driver to self-regulate (compensation)*: Self-regulation at the strategic, tactical and operational levels of driving can be exercised by drivers to manage pressure from competing activities, to regulate the timing of the engagement and to control resource investment (Lee *et al.*, 2008a). It was found that drivers take the decision to receive an incoming call or initiate a call if they believe that their conversation is important (Nelson, Atchley & Little, 2009), even if they believe that there may be an associated risk with conversing on a mobile phone while driving.

The consequences of driver distraction can be dramatic. According to the Automotive Association in South Africa, an average of forty people die and twenty-five are permanently disabled on South African roads daily (The Automotive Association of South Africa, 2011). This has a negative socio-economic impact on the country. In the United States of America (USA), the use of mobile phones while driving has been indicated as a factor in crashes that have led to 995 deaths in 867 crashes, which represents 18% of all fatal distracted driving crashes nationally (National Highway Traffic Safety Administration, 2010). This figure is similar and sometimes worse in other parts of the world.

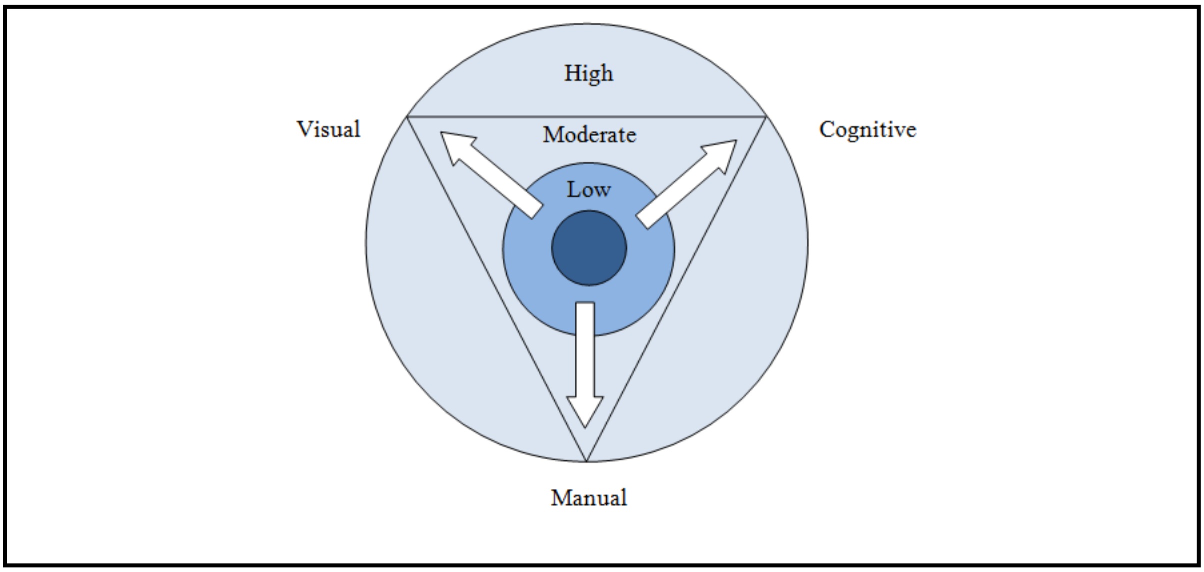


Figure 2.3: Framework of Sources of Driver Distraction (Strayer, Watson & Drews, 2011)

Figure 2.3 shows a framework for understanding the sources of driver distraction. Driving can be affected by the competition involving visual processing, manual interference and cognitive distraction. These sources of distraction can occur independently or simultaneously; that is interacting with different devices can result in competition from one, two, or all three sources.

	Visual distraction: "eye-off-road"	**Cognitive distraction: "mind-off-road"**
Visual behaviour	Frequent and long off-road glances	Visual attention allocated to the middle of the road plane
Vehicle control	Sudden steering movements with large (2 - 6 degrees) amplitude, large steering entropy	Corrective movements with small (less than 1 degree) amplitude, small steering entropy
Vehicle state	Large and frequent lane deviations, speed decrease and headway increase	Unchanged or small lane variation speed does not change significantly

Table 2.2: Summary of Distraction Assessment through Visual and Driver Performance Metrics (Yekhshatyan, 2010)

The different types of driver distraction have an impact on the way the driver behaves and on the way the car moves. A study on visual and cognitive distraction (Yekhshatyan, 2010) revealed the impact of these distractions on visual behaviour, vehicle control and the vehicle state. While visual distraction causes large lane deviation, cognitive distraction affects the state of the vehicle less (Table 2.2).

Driver distraction has cognitive implications. The following paragraphs discuss some psychological aspects related to driver distraction.

2.5 Psychological Aspects of Driver Distraction

Methods to reduce driver distraction are often studied in human factors, which include two key aspects: psychology and engineering. The main goal of the human factor discipline is to study the design of products in a way that users can use them effectively. Cognitive load theory is a theory that describes learning structures of information processing involving long-term memory, which stores knowledge and skills on a permanent basis; and working memory, which performs tasks associated with consciousness (Cooper, 1998). Cognition theory involves attention, perception memory and learning skills. The following sections discuss the attention and the multi-resource theories, as well as the mental workload involved when performing several tasks simultaneously.

2.5.1 Processing of Multiple Tasks

The multiple resource theory (MRT) studies the processing of multiple tasks simultaneously. The theory introduces a four-dimensional model, which includes *modalities*, *codes*, *stages* and *responses* (Figure 2.4). MRT is used to improve designs, which present the possibility of multi-task resource overloading.

The model of MRT suggests that tasks that share the same pool of resources interfere with each other. For example, driving requires a visual modality (resource) that can interfere with an ICCS provided with a GUI. This can be addressed by introducing an ICCS without visual feedback.

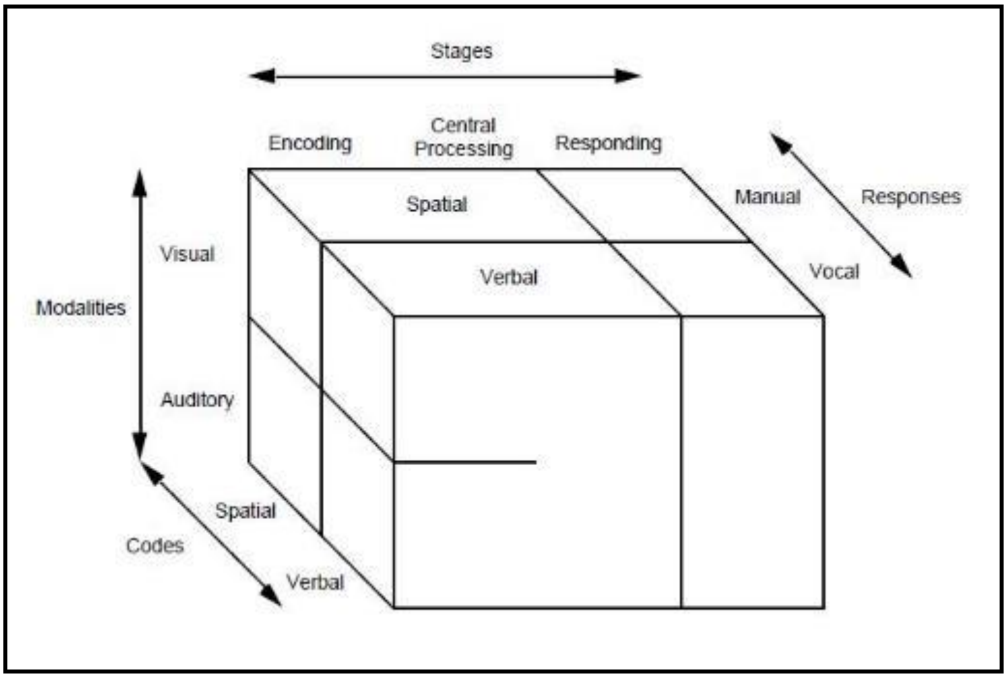


Figure 2.4: Model of Multiple Resource Theory (Wickens, 1984)

According to Wickens (1992, 2008), tasks that share the same pool of resources may interfere with each other. This theory also supports the statement that two concurrent tasks with different modalities will have less interference than two tasks with the same modality (Gellatly, 1997). This means that combining manual-visual tasks and a speech task would result in less cognitive load than two concurrent visual tasks.

Other studies of dual-task interference maintain that the resource model of Wickens is too simple to address dual-task interference issues accurately. Regardless of the resource pool involvement, increased task similarity may be associated with increased interference (Bailey, 1994).

In the mobile phone context, this interference can be described in terms of four elements (Haigney & Westerman, 2001): visual processing, aural processing, a generalised cognitive function and kinaesthetic processing. This description is similar to the distinction that Wickens makes between the processing stage and processing modalities. In the driving context of a car, it is very important to distinguish between visual and auditory modalities. Mobile phone use and driving performance are complicated tasks because they are both multifaceted, dynamic tasks with physiological and cognitive processing requirements that may vary depending on internal and external parameters. Driving along a winding road requires more response processing than driving along a motorway. Similarly, dialling a number may require a combination of perceptual/visual and response processing, whereas conducting a conversation may require a combination of auditory, verbal and central processing. Sometimes, a conversation could require a high degree of spatial processing (e.g. if the conversation is about the route that the driver should follow).

2.5.2 Workload

The workload of a driver depends on several components, however, being aware constantly of driving situations minimises the mental workload. This means that an adaptive ICCS may help reduce this workload (Tashev *et al.*, 2009).

An investigation into the divided attention of drivers (Iqbal, Ju & Horvitz, 2010) showed that the intensity of the cognitive interference of a phone conversation with driving depends on the type of conversation. The different types of conversations are assimilation (listening), retrieval (respond: memory) and generation (respond: information). Driving problems may arise when cognitive resource demands exceed resource availability, such as when drivers are engaged in conversations involving retrieving information from memory while dealing with a complex situation on the road. These findings underscore the complexity of interactions between different kinds of conversation-centric tasks and driving.

Studies have distinguished between different types of workload, namely, driving-induced and dialogue-induced cognitive types (Villing, 2009). For a driving-induced cognitive load, the system should be able to suspend the dialogue in order to allow the driver to concentrate on the driving task; in the case of dialogue-induced cognitive load, the question should be reformulated.

Driver distraction can interfere with the driving task, by sharing the resources required for driving activities, such as visual, auditory, motor and cognitive resources (Ranney, 2008). The conflict between ICCS and driving task demands is a source of driver distraction (Verwey, 2000) and needs to be mitigated.

The competition between the primary and secondary tasks has the potential to impair the driver. This impairment, in the form of loss of attention and driver errors, can have a negative impact on driver performance, which can result in an increase of accident risk.

2.6 Causes of Car Accidents

Literature mentions two types of causes of driver distraction: internal and external. Internal causes of driver distraction refer to anything that happens in the vehicle and that has a direct or indirect impact on the driver attention. According to an extensive survey in the USA (Stutts, Reinfurt, Staplin *et al.*, 2001), some internal causes of distraction include: talking on a mobile phone, answering a mobile phone, dialling on a mobile phone, eating, drinking, preparing to eat or drink, manipulating music/audio controls, smoking, reading or writing, conversing, baby distraction, child distraction, adult distraction, etc.

External causes of driver distraction are anything from outside the vehicle that has an impact on the driver’s ability to safely drive the car. Several studies have focussed on external factors causing driver distraction (Milloy & Caird, 2011, Yannis, Papadimitriou, Papantoniou *et al.*, 2013). Such external causes include: other vehicles, pedestrians, accident or incident outside the vehicle, landscape or buildings, animal, advertising sign, road signs, sun or vehicle lights, etc.

Driving performance deficits can occur as a result of driver distraction. This includes degraded lane keeping, degraded speed control, increased reaction time, missed traffic signals, a shorter or longer distance between cars, unsafe gap acceptance, reduced situation awareness, poorer visual scanning, reduced horizontal field of view and missed mirror checks (Bayly, Young & Regan, 2009).

Driver distraction is a contributing factor in 10 - 12% of crashes. Approximately one-fifth (20%) of these crashes involved the driver interacting with technology (Gordon, 2008). Several factors, besides driver distraction, are often reported as the main causes of car accidents. These include driver impairment (alcohol, drugs, old age, physical impairment, or a combination of these factors), road design and vehicle design (seatbelts, maintenance, centre of gravity).

According to the federal highway administration in the USA (Stuster, Coffman & Warren, 1998), the following factors are contributing causes of car accidents:

- The risk of having a crash is increased both for vehicles travelling slower than the average speed and for those travelling above the average speed,
- The risk of being injured increases exponentially with speeds much faster than the median speed,
- Most crashes related to speed involve the speed being too fast for the conditions,
- Effectiveness of traffic calming (speed bumps).

Distraction has been reported to be the main cause in rear-end crashes, same direction crashes, single vehicle crashes and crashes occurring at night. The risk of being involved in a car accident increases by a factor of 4.1 when the driver is distracted (McEvoy, Stevenson & Woodward, 2006).

A study was conducted (Stutts *et al.*, 2001) for the american automobile association (AAA) Foundation in which they examined detailed crash records from the Crashworthiness Data System (CDS) collected between 1995 and 1999. This study found that of the crashes examined, 8.3% were the result of the driver being distracted by some event, object or activity inside or outside the vehicle.

Some measures have been researched in order to reduce driver distraction. The following paragraphs discuss some solutions that have been applied so far.

2.7 Solutions Used to Reduce Driver Distraction

Over the years industry and academia have worked towards reducing fatalities resulting from driver distraction. Reactive and proactive measures have been investigated. Features, such as seatbelts and airbags, have been used to reduce the severity of a crash. Several technologies have been introduced as proactive measures. ICCS are amongst the most common technologies, but when ill-designed, they can cause distraction. The following sub-sections will discuss the design of several user interfaces used to improve ICCS. Intelligent Transportation Systems will be also discussed as a technique often used to improve driver safety.

2.7.1 Speech User Interfaces

Speech user interfaces (SUI), also called voice user interfaces, are what a user interacts with when communicating with a speech-based application (Cohen, Giangola & Balogh, 2004). SUI have been used in a wide range of applications both in desktop and mobile applications. The use of the speech channel is encouraged strongly in situations where the hands of the user are used for other purposes, such as steering a car.

The design of a SUI is subject to several challenges related to the application domain. These challenges become more critical when designing for a domain such as the automotive domain, because the ambient noise could negatively affect the speech recognition accuracy. When designing a general SUI, the following requirements are often taken into account (Suhm, 2003): speech recognition accuracy, dialogue flow, reliability (ambient noise), human cognition (limited working memory), user (native speaker) and hardware (microphone). Some user studies have highlighted the importance of the first language of the user and of background noise for the success of speech recognition on a mobile platform (Khalil, Khalifeh & Darabkh, 2012).

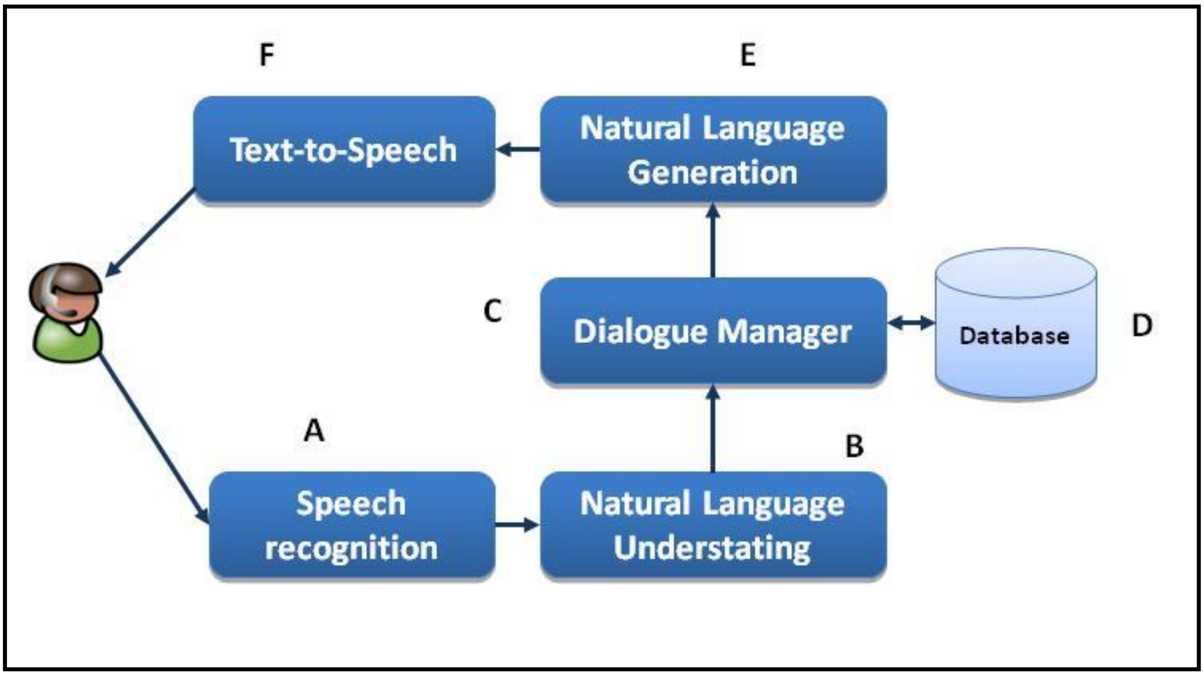


Figure 2.5: Typical Architecture for Speech-based Application – Adapted from Becker et al (2006)

Figure 2.5 depicts the typical components of a speech-based application (Becker *et al.*, 2006). These include: a speech recogniser (A) that processes the user's voice and converts it into text, a Dialogue Manager (C) to control the interaction with the user and a mechanism for conveying information to the user. A database (D) that provides data is needed by the DM, for example contact numbers or map information can be stored in the database. More advanced systems incorporate modules for Natural Language Understanding (B) and Natural Language Generation (E) to manage inputs from the user and to prepare outputs from the system.

Finally, the text-to-speech (F) converts the text into speech and plays it through the speaker. This is the way the driver receives feedback from the system.

2.7.1.1 Automatic Speech Recogniser

A speech recogniser can also be called a speech-to-text (STT) engine. The main function of a speech recogniser is to convert the voice of the user, captured by a microphone, into text. The spoken input is made of linguistic units called phonemes. The speech recognition engine uses techniques such as hidden markov models (HMM) to analyse phonemes and return a list of hypotheses. Results are always given in the form of a list of hypotheses together with their confidence levels. The best speech recognisers are able to recognise a large set of words independently of the speaker and in a continuous manner. The efficiency of speech recognisers is measured by the word error rate.

Speech recognition engines require two types of files to recognise speech: an acoustic model and a language model. The acoustic model is created with a set of pairs comprised of audio recordings of speech and their transcriptions. The training phase compiles each pair into a statistical representation of the sounds that make up each word. The language model is a file containing the probabilities of sequences of words. A grammar is a much smaller file containing sets of predefined combinations of words. Language models are used for dictation applications, whereas grammars are used in desktop command and control applications.

The word recognition accuracy was discovered to be inversely proportional to the level of noise. Native English speakers managed to perform better than non-native English speakers under all noise conditions, especially with loud background noise.

A major limitation of speech recognition on mobile devices is that the recognition is performed on a remote server. The speech recognition can be intermittent or slowed down by the quality of the mobile network connections (Lei, Senior, Gruenstein *et al.*, 2013).

The training of acoustic models is done using techniques such as HMM or gaussian mixture models (GMM) or neural networks. The use of more efficient algorithms for training, such as deep neural networks (DNN), can facilitate an offline implementation of a mobile speech recognition engine.

2.7.1.2 Natural Language Understanding

Natural language understanding (NLU) is responsible for the assignment of meaning to any text coming from a speech recogniser. This is often performed through three steps: lexical, syntactic and semantic analysis. The lexical analyser parses the input text and converts a sequence of characters into sequences of tokens. Tokens represent parts of speech (verb, noun and object) that are assigned to each word. The syntactic analyser uses the tokenised text to determine its grammatical structure. Finally the semantic analyser relates syntactic structures from the levels of phrases and sentences to their language-independent meanings.

Here again, several hypotheses may be forwarded to the dialogue manager (DM) module; each result is attached with a confidence score (from 0 to 1). The accuracy of the NLU may sometimes be of poor quality.

2.7.1.3 Dialogue Manager

The DM is the main component of a SUI. It controls the dialogue between the system and the user. When a command is understood successfully, the action requested is executed. The DM can seek clarification from the user if the command is ambiguous. Mixed-initiative dialogue managers are the most flexible as the system or the user can initiate a dialogue. Previous research (Seneff, Lau & Polifroni, 1999) has shown that the ability of the user to interrupt the system (barge in) can improve the usability of SUI.

Several techniques are used to implement dialogue managers. These include finite-state based, frame-based and agent-based dialogue management (Bui, 2006). With finite-state based dialogues, the dialogue is designed as a set of states that can change depending on the situation. This results in unnatural dialogues as the user has to provide information to the system only when the system needs it. The advantage of such techniques is the ease of implementation.

Frame-based dialogues contain frames which, in turn, are made of slots. A frame is a data structure that manages commands given by users. Parameters needed for the commands are saved in slots. For example, a frame representing a text message will be made of a slot holding the contact to which the message is addressed and a second slot holding the actual message to be sent. Several slots can be filled at once, leading to mixed-initiative dialogues where either the user or the system can initiate the conversation.

Agent-based dialogues are more advanced, using artificial intelligence agents to control the flow of the dialogue. However, the implementation of such techniques can be challenging and time-consuming. A mobile phone has several resource constraints. The implementation of an agent-based dialogue strategy might drain the battery.

2.7.1.4 Natural Language Generation

The NLG module selects the most appropriate answer depending on the context and the modalities available for data presentation. In the scenario of an application that has to be used in a car, the only safe modality is speech. Other modalities can prevent the driver from looking at the road while driving. The NLG module can also decide on the volume of the speech and the choice of the voice (e.g. male or female voice).

2.7.1.5 Speech Synthesiser

Speech synthesisers are also called TTS engines. TTS converts text to speech. Speech synthesisers are widely available on a number of devices and are included in most operating systems (Android, Linux, Mac Operating System and Windows 8).

The successful design of an SUI depends on several factors (Cohen *et al.*, 2004). These include the design of prompts, the grammar used and the dialogue logic. Prompts have two main purposes. They cause the user to speak and they convey to the user what may be spoken. Synthesised speech is difficult to understand in noisy environments. This is why users will benefit from commands such as “REPEAT” that allow the user to have a sentence repeated by the system.

2.7.1.6 Guidelines and Requirements to Design Speech User Interfaces

Speech recognition can be used in situations where the other input channels of the user are not available. Important factors are the surroundings of the user (i.e. background noise) and that the user should be alone and stationary. Several guidelines designed by industry have been proposed to help the designer of a SUI. The European Statement of Principles (European Commission, 2000) provides a set of guidelines in three categories, namely input, output and hardware requirements. These guidelines are the following:

Input Guidelines:

- Keep the number of words small in the speech recognition vocabulary,
- Keep each speech input short,
- Use speech inputs that sound distinctly different from each other,
- Provide immediate feedback for every speech input,
- Make error correction intuitive,
- Do not use speech to position objects,
- Use a command-based user interface,
- Allow users to turn the Speech Recogniser on and off quickly and easily.

Output Guidelines:

- Messages that require an urgent action should be a single word or a short sentence with the fewest number of syllables possible. Drivers should be able to understand the message immediately,
- Messages that are not urgent or for which a response may be delayed can be a maximum of seven units of information in the fewest number of words possible. If the information cannot be presented in a short sentence, the most important information should be presented at the beginning and/or the end of the message;
- Offer a possibility to repeat output,
- Offer visual support for auditory output.

Hardware Requirements:

- Use a highly directional, noise cancelling microphone,
- Consider using headphones or an earphone (versus a speaker) for auditory feedback,
- Use full duplex audio (combine input and output channels),
- Consider providing a backup input technique to speech.

Talking on the phone, regardless of phone type, has a negative impact on driving performance especially in detecting and identifying events (Ishigami & Klein, 2009). Use of hands-free phones can be as dangerous as the use of handheld phones while driving; possibly even more dangerous because of the underestimation of danger.

The National Highway Traffic Safety Administration (NHTSA) in the United States published Driver Distraction Guidelines; these guidelines are meant to promote safety by discouraging the introduction of excessively distracting devices in vehicles. These guidelines were released in two phases. NHTSA issued the first phase of these guidelines (National Highway Traffic Safety Administration, 2012a). The Phase 1 Guidelines cover embedded in-vehicle electronic devices that are operated by the driver through visual-manual means. Phase 1 Guidelines are as follows:

- The driver's eyes should usually be looking at the road ahead,
- The driver should be able to keep at least one hand on the steering wheel,
- Any interactive task performed by a driver should be interruptible at any time,
- The driver should control the human-machine interface and not vice versa,

- Displays should be easy for the driver to see.

The Phase 2 Guidelines apply to mobile devices that are operated through visual-manual means and are based on the same general principles as the Phase 1 guidelines. Three areas are addressed by these guidelines, namely: pairing, driver mode and advanced technologies. Processes are suggested in order to implement tasks such as pairing mobile phones to the car. Some recommendations are made to select safe tasks when in driving mode. Some recommendations are also made to guide the design of advanced technologies that reduce driver distraction. This research project addresses the advanced technologies category.

Benefit	**Description**
Flexibility	Users can switch modalities depending on the environment and their abilities.
Efficiency	Improved speed and accuracy.
Task effectiveness	More tasks are achieved because the user does not give up owing to frustration.
Error-handling or cross-modality synergy	Weaknesses of one modality can be minimised by another modality.
Learnability	Easy to learn because it is more natural. Minimise cognitive load owing to the interface.
User experience enhancement	More engaging than a unimodal interface.

Table 2.3: Benefits of Multimodal Interfaces

2.7.2 Multimodal Interfaces

Multimodal interfaces (MMI) process two or more combined user inputs in a coordinated manner with multimedia system outputs (Maybury, 2002) and aim to recognise naturally occurring forms of human language and behaviour by incorporating one or more recognition-based technologies (e.g. speech, pen and vision).

Table 2.3 gives an overview of the most relevant benefits of MMI for ICCS. Flexibility refers to the choice of input and output modalities that users have. These choices might be necessary, as in the case of a user who is unable to communicate using a particular modality such as speech, vision, or gesture because of deafness, blindness or paraplegia.

The KIA UVO (KIA, 2010) **The BMW iDrive (BMW, 2009)**

Figure 2.6: Example of ICCS Operated with MMIs

MMI allows users with different skills, ages, native languages and physical abilities to work more effectively with a computer system. For example, a non-native speaker might prefer to use manual input more often than speech input (compared with a native speaker).

The success of an MMI depends on the way inputs are combined (fusion) or split (fission). Figure 2.6 illustrates two examples of ICCS utilising MMIs. They both make use of speech, steering wheel and dashboard buttons as input. Speech and GUIs are used as output modalities only. Most existing ICCS provide a visual display. The presentation of information on a screen can divert the attention of the driver from the road. Studies have shown that different push-to-talk (PTT) solutions can improve user satisfaction (Lin, Hwang & Green, 2009). This is because users are less frustrated by recognition errors as the PTT button helps to capture the correct speech.

A wide range of modalities are now used in the context of the car. A recent study investigated how ICCSs can take advantage of multimodality by using gesture input and speech (Muller & Weinberg, 2011). The benefits offered by multimodal systems are important in minimising the driver workload. These benefits can be envisaged for a mobile, context-aware ICCS. These benefits can therefore be used as metrics in a field study when investigating the positive impact of such systems.

2.7.3 Intelligent Transportation Systems

Intelligent transportation systems (ITS) are state-of-the-art in-car systems based on information, communication and satellite technologies. ITS are used to reduce traffic congestion, enhance safety and improve the quality of the environment (Shah, Lee, Mahalik *et al.*, 2006). Information used by ITS include communications, sensor and control technologies. Many such technologies have been developed to enhance vehicle safety, prevent crashes, reduce trauma during a crash or reduce trauma following a crash.

Examples of ITS technologies for ICCS include advanced driver-assistance systems (ADAS), intelligent-speed adaptations, driver-monitoring systems, collision-warning and avoidance systems, lane keeping and lane-change warning systems, visibility-enhancing systems and seat-belt reminder systems.

ADAS can be grouped into five categories: lateral control systems (lane keeping and warning systems), longitudinal control systems (adaptive cruise control or distance keeping systems), reversing or parking aids, vision enhancement systems and intelligent speed adaptation. Despite the benefits gained from using innovative technologies, there is great potential for distraction from in-vehicle technologies and ADAS (Brooks & Rakotonirainy, 2005). A study showed that adaptive cruise controls (ACC) improve the performance and the subjective acceptance of the users (Jameson, 2003). The ACC in this study was developed using a biologically inspired technique, namely a genetic algorithm.

ADAS can also be used to provide active assistance, such as emergency braking. Projects such as COMUNICAR (Andreone, Amditis, Deregibus *et al.*, 2005), investigated opportunities and challenges related to ADAS. The Adaptive Integrated Driver-Vehicle Interface (AIDE) system (Amditis, Andreone, Pagle *et al.*, 2010) was designed to monitor the driving state and to perform some adaptation when necessary. AIDE is an adaptive system that uses the sensors of the car to determine the environmental conditions.

Sensing the context in a car can be an intricate task. Some research has used intrusive technologies such as a mounted video camera that monitors the physical state of the driver (fatigue, drowsiness) (Andreone *et al.*, 2005). Mobile phone sensing has recently been used to predict events related to the transportation system. Road conditions, including potholes and bumps can be accurately detected in real-time (Eriksson, Girod, Hull *et al.*, 2008, Mednis, Strazdins, Zviedris *et al.*, 2011). Research has shown that the means of transportation can also be predicted by using sensors (Bedogni, Di Felice & Bononi, 2012). Research conducted by Bedogni *et al.* (2012) to detect whether the user is in a car or a train, showed that using the multiple sensor inputs of a smartphone (i.e. the accelerometer and the gyroscope) can improve the accuracy of the classifiers significantly. Machine learning has also been used to implement nonintrusive and real-time detection of visual distraction, using vehicle dynamics data and without using eye-tracker data as inputs to classifiers (Tango & Botta, 2013).

Google released an Application Programming Interface (API) (Android Developers, 2013) capable of predicting whether the phone holder is walking, driving, cycling or not moving. Experiments conducted with the activity recognition API revealed some shortcomings in terms of accuracy and responsiveness. More recently, a study was conducted focusing on determining whether the phone belonged to the driver (Wang *et al.*, 2013). This study was designed using the mobile phone sensor information.

A context-sensitive mediation system is believed to provide substantial benefits in reducing driver inattention. A study on the proactive mediation of phone conversation recommends the use of such mediation (Iqbal, Horvitz, Ju *et al.*, 2011). Mediation in this context can be defined as the intervention in the system; this intervention can be achieved by using an adaptive system. Context-awareness has also been used in efforts to mitigate driver distraction. Context-awareness was used to implement some adaptations such as burden-shifting, time-shifting and activity-based sharing (Lindqvist & Hong, 2011).

2.7.4 Multimodal Interface for In-car Info-Communication

This research finds its roots in a previous study on developing a safer ICCS. Based on the fact that the interaction with several existing ICCS was not adaptive, an adaptive ICCS was designed and compared with a non-adaptive ICCS.

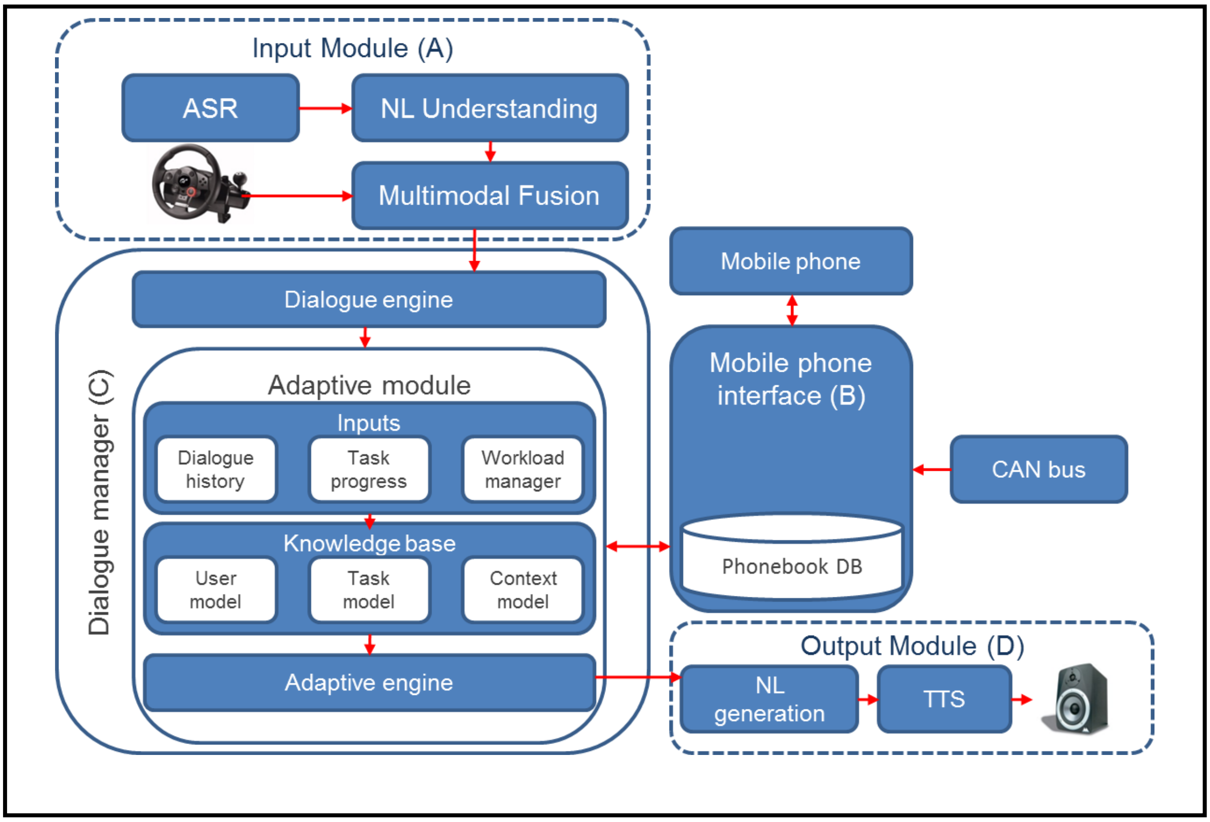


Figure 2.7: Multimodal Architecture of MIMI (Tchankue *et al.*, 2010a)

A prototype called Multimodal Interface for Mobile Info-communication (MIMI), was developed and evaluated (Tchankue, Wesson & Vogts, 2010b, Tchankue, 2011, Tchankue *et al.*, 2011). MIMI includes four main modules: Input, Dialogue Manager, Mobile phone interface and Output modules (Figure 2.7). The Adaptive Module comprises Inputs, a Knowledge base and the Adaptive engine. The Adaptive engine was implemented using a neural network based on the speed of the car (using a driving simulator) and the steering wheel angle. The system postpones incoming calls and text messages when the distraction level is high.

Thirty (30) participants, equally distributed between genders, were recruited and used the adaptive and the non-adaptive versions of MIMI. These two versions were evaluated using a counterbalancing approach. The analysis of the results showed that the observed difference between males and females was not significant. The results also showed that most participants achieved better performance with the adaptive interface and preferred the adaptive version of the prototype.

Despite several advantages of the architecture of MIMI, some shortcomings still exist. These include the fact that MIMI was designed to be built into a car and therefore cannot be implemented on a mobile device. The Adaptive Module uses a distraction level, which is generated by using only two variables (speed and steering wheel angle). There is a need to use a wider range of variables in order to improve the accuracy of the determination of the distraction level.

2.8 Existing Mobile Applications

Since the introduction of the iPhone (iOS) in 2007, Apple has been the leader in innovations of micro-technologies on smartphones. Other mobile phone operating systems, such as Android and Microsoft Windows Phone have since enabled mobile phones using their systems to take advantage of several embedded sensors. Accelerometers can detect changes in the orientation, vibration, rotation or fall of a phone by detecting linear acceleration on the three axes: x, y and z.

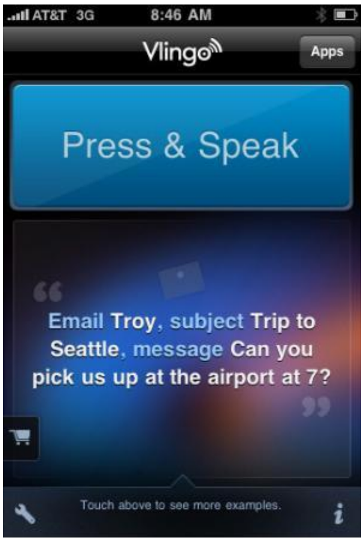


Figure 2.8: Vlingo, an Example of a Mobile ICCS (Vlingo Corporation, 2010)

A gyroscope can detect the angular acceleration for the x, y and z axes. This enables more precise virtualising of pitch, yaw and roll.

Some examples of mobile ICCS that are currently available on the various mobile application stores are listed below:

- *Handcent*: a free Android application used to read aloud incoming text messages and Multimedia Messaging Service (MMS) whilst driving. It can also help to dictate a text message,
- *StartTalking*: a free Android application that can be used to dictate and read text messages, to update a twitter or Facebook status, to ask the time, weather and any other information (AdelaVoice, 2010),
- *Vlingo*: an application freely downloadable and available for several platforms (Android, iPhone and Blackberry). Figure 2.8 depicts a screenshot of Vlingo sending an email on an iPhone,
- *ZoomSafer*: an Android application that blocks any incoming call or text messages when a driver is driving a car (ZoomSafer, 2011),
- *Drivesafe.ly*: an application that reads incoming text messages when the car is moving. The Blackberry and iPhone versions are available. Android and Windows versions will soon be available,
- *Dragon Mobile Assistant:* it is a speech-based application that can assist drivers in dictating emails and text messages, checking the traffic and weather (Nuance Inc, 2013).

Some freely available mobile applications can read incoming text messages when the car is moving. Unfortunately, this feature has to be set by the driver before the start of the journey. This setting can be easily forgotten and can cause the risk of unsafe behaviour whilst driving. It is interesting to note that none of these applications support a context-aware adaptation. The driver may still be at risk even though one of the previous applications is used to minimise driver distraction.

2.9 Conclusion

The aim of this chapter was to answer the research question *RQ1: What are the causes and effects of driver distraction?* A discussion on ICCS including their main functionalities was given as well as a list of examples. The driving model involves several complex activities, which make the driving task very difficult. The complex nature of driver distraction was discussed together with its impact on driver performance and safety. Several solutions investigated so far, such as SUI, MMI and ITS, were discussed. Finally, mobile devices are becoming a platform for ICCS as they can host hands-free applications.

Causes of driver distraction can be internal and external. Some internal causes include conversing with passengers, dialling or answering a mobile phone. Some external causes include interaction with other road users, accident scenes, landscape and sunlight. The result of driver distraction can be translated into decreased driver attention and dangerous driving behaviour (speed, following distance). This ultimately increases the risk of a car accident.

The next chapter will discuss different approaches that can be used to design context-aware ICCS. The most relevant approaches will be used to design a model for a mobile, context-aware ICCS.

Chapter 3: Context-Aware Applications

3.1 Introduction

The goal of this chapter is to answer the research question *RQ2: What are the existing models for context-aware applications?* The answer to this question will provide insight into the benefits of context-aware applications. This will guide the design of a model to reduce driver distraction by using contextual information.

As discussed in the previous chapter, distraction can be caused by several factors. An interesting fact is that a driver conversing with passengers whilst driving is less distracted than a driver involved in other secondary tasks. This can be explained by the fact that passengers are aware of the current driving context; they can see when the driving situation needs more attention and can then initiate a mitigation mechanism, which often consists of pausing or stopping the conversation. This highlights the importance of context-awareness for interactions in ubiquitous scenarios where users are often multitasking.

Considerable effort is being put into designing context-aware applications to solve problems in various areas. Although these applications provide several benefits to users, their development process can be intricate and is domain-dependent. This chapter investigates the models and techniques that are used to design context-aware applications. Particular attention will be placed on in-car communication systems (ICCS). Context-aware and adaptation techniques will be reviewed so as to identify requirements for context-aware ICCS.

Section 3.2 defines and classifies context-awareness as many definitions can be found depending on the purpose of the application. Section 3.3 discusses models of context information with the aim of selecting the most appropriate model for a mobile context-aware ICCS. Section 3.4 discusses different approaches used to design context-aware applications. Section 3.5 focuses on sources of context information, especially sources that are available on mobile devices. Section 3.6 discusses several machine learning techniques that are used to determine the context. Section 3.7 discusses several context adaptations that can be designed for a mobile, context-aware ICCS. Finally Section 3.8 discusses general requirements that should be taken into consideration when designing context-aware applications.

3.2 Context-Awareness

Pervasive and ubiquitous computing are terms that are becoming commonly used in the field of computer science. They imply that computing no longer only takes place on desktop computers, but has become ubiquitous through the improved performance and smaller size of mobile devices. Today, applications are designed to be used in cars, homes, portable devices and appliances. This led to the development of context-aware applications in the early nineties, to address the lack of ubiquity of traditional applications.

A better understanding of context-aware applications depends on the definition of the term "*context*". The dictionary defines context as "*the circumstances that form the setting of an event, a statement or an idea*" (Oxford Dictionaries, 2011). These circumstances will help the event, statement or idea to be fully understood and assessed. In the case of driving, the context will only refer to circumstances that form the setting of an event.

Another definition refers to context as "*information that captures the characteristics of a ubiquitous computing environment*" (Zhang, Cao, Zhou *et al.*, 2009). The characteristics of the computing environment may include available resources (Central Processing Unit (CPU), disk space or memory) as well as other external events that may influence the computing environment.

According to Dey (2001), one of the pioneers in the field of context-awareness, "*context refers to a set of states and settings that determine the behaviour of an application"*. Context is restricted only to what interests or influences the user. This definition introduces a very important concept: the change in the behaviour of the application, which is merely an adaptation of the application to fit the current context.

The literature in the field of context-awareness often refers to these terms as: situation-aware, activity-aware and context-aware. Activities are high-level situations such as "*walking*", "*driving*" or "*running*". Context can be considered to be a set of parameters that influence the situation of a user such as location, time and temperature. All these factors have a given value at a certain state (Weißenberg, Gartmann & Voisard, 2006).

When driving, a driver can be in several situations depending on the complexity of the driving situation, which can be either difficult or easy. Under easy situations, the level of distraction is believed to be fair as the driver requires less attention to operate the car safely. Driving on a straight, dry road can be considered to be an easy driving context because the architecture of the road and the road conditions are believed not to be challenging. Conversely, difficult situations require a lot more attention from the driver and therefore secondary tasks would be prohibited. Such difficult situations may include negotiating a curvy section on a wet road, which can be quite complicated.

All definitions highlight concepts such as circumstances, characteristics of the environment and the state of these events. This suggests that the context is the description of all events that can influence an application at any given time.

3.2.1 Definition of Context-Awareness

As proposed by Dey (2001), a context-aware application is software using contextual information in order to adapt its behaviour. The adaptation may be in terms of user interfaces (information presentation), algorithms or other parameters that have an effect on the task being performed. Earlier definitions denoted context-aware software as an application that adapts according to the location of use, the collection of nearby people, hosts and accessible devices, as well as adaptation to changes to such things with time (Schilit, Adams & Want, 1994).

Although several definitions of context-aware applications exist, all agree on the fact that the application has to adapt itself. A well-known example of a context-aware desktop application is the contextual menu (right click) used by Microsoft Office applications to provide a popup menu depending on the task of the user. Today, owing to the increased capabilities of smartphones, most ubiquitous applications are location-aware. These applications make use of embedded GPS chips to provide services and information depending on the location of the user.

3.2.2 Classification of Context-Aware Applications

One of the first attempts in classifying context (Schmidt, Beigl & Gellersen, 1999) suggests organising context into two general categories: *human factors* and *physical environment*. Each category can be divided into three subcategories.

This scheme also suggests defining *history* as an additional dimension of context information. The history of some contextual data can be used to generate new knowledge. For example, the list of Global Positioning System (GPS) locations recorded gives information about the route followed by the user.

3.2.2.1 Human Factors

Context-aware applications based on human factors obtain their inputs from human factors-related data. The human factor category is organised into three subcategories, which include users, social environment and tasks:

- *Information on the users*: This information depends on the user and may change with time for a specific user. It can include the following: habits, mental state, expertise, or physiological characteristics, etc,
- *Information on their social environment*: This information is based on the social environment of each user. It can include the following: proximity of other users, their social relationship, collaborative tasks, etc.,
- *Information on their tasks*: This information is based on the tasks performed by the user. It can include the following: goal-directed activities, higher-level abstraction about general goals of the users, etc.

The design of in-car applications can be more challenging because of possible safety issues, especially when the attention of the driver is divided between the primary and a secondary task. Therefore the human factors to be considered for in-car context-aware applications are information related to the mental state, expertise and type of task.

3.2.2.2 Physical Environment

Context-aware applications based on physical environment are influenced by environment-related data. The physical environment category also has three subcategories. These include:

- *Location information*: the location can be either absolute or relative. The context-aware application uses the physical location of a user to change its behaviour. Absolute location can include GPS-coordinates and relative location, such as the distance from a specific object or the position of the user inside a car (driver, passenger in front seat and passenger in back seat),

- *Infrastructure information*: this includes all information about surrounding communication equipment (e.g. networking) and that may help in having a better context (e.g. bandwidth), and
- *Physical conditions information*: in this case the meteorological and acoustic information are used. It can include the level of noise, the brightness, vibration, the outside temperature, and the room lighting.

This type of data would be relevant for in-car applications as the car is subjected to all kinds of environmental changes. The location changes all the time as well as the external physical conditions.

Context-aware applications can also be classified in terms of their adaptation mechanisms. Chen and Kotz (2000) suggest two types of context-aware applications: *active* and *passive*.

Active adaptation occurs when applications adapt automatically to discovered context, by changing the behaviour of the application. Passive adaptation occurs when applications present the new or updated context to an interested user or make the context persistent for the user to retrieve later. Active context is more interesting as it does not require any actions from the user. However, its success depends on the accuracy of the context determination. An inaccurate action can be dangerous or frustrating for the user.

3.3 Models for Context Information

Models are used in several fields to simplify and explain problems or situations. In computer systems, these allow an abstraction of the system to be designed. A number of models were introduced in order to develop context-aware applications. These include key-value models, mark-up scheme models, graphical models, logic-based models, ontology-based models and object-oriented models (Strang & Linnhoff-Popien, 2004).

Research in context information modelling (Bettini, Brdiczka, Henricksen *et al.*, 2010) has proposed a set of requirements that context-aware models need to meet. A model for context-aware applications should:

- Handle a variety of information types as well as their relationships,
- Model high-level context abstraction describing real world situations,
- Model histories of context information,

- Include the uncertainty of context information in the model.

It appears that the handling of uncertainty is very important as most data are provided by error-prone sensors. The concept of time is also highlighted by the history that has to be kept. A detailed description of each model is given in the following subsections.

3.3.1 Key-Value Models

Key-value models describe contextual information by providing a list of attributes together with their corresponding values. This model is simple, but powerful enough to allow pattern-matching queries.

Shortcomings of this approach include the difficulty of querying and reasoning or interpreting the information captured. This model is only efficient in solving problems that require exact matching. Sophisticated systems would be difficult to model using this technique.

Keys	Values
Date time	Between 10 to 12 AM
Location	In room 1045
Co-location	With [User Adams]

Table 3.1: Example of a key-value Model

A key-value model was used to design the Active Badge application (Schilit *et al.*, 1994), which determines the location of the wearer of a badge by periodically broadcasting the unique identifier of each wearer. Keys included [Date time], [Location] and [Co-location] (Table 3.1). The following paragraph discusses a more complex information model used in designing context-aware applications.

3.3.2 Mark-up Scheme Models

Mark-up scheme models represent contextual information using mark-up languages. These languages are text files that store the description of the information as well as its structure by using tags. The most popular of these is the hypertext markup language (HTML) language used to design web pages and extensible markup language (XML) used to exchange data across different applications and platforms. Mark-up languages form the basic components of mark-up scheme models, which are characterised by a hierarchical data structure using a combination of tags with attributes and content. Attributes are the properties of the context. This model can be seen as a structured key-value model. This helps with complex information retrieval as the parsing can be achieved quite easily by using technology such as DOM or SAX (Lam, Ding & Liu, 2008) and querying using XPath.

Mark-up models have some shortcomings, which include the difficulty of modelling general-purpose context information (Strang & Linnhoff-Popien, 2004). Context-aware applications modelled by this technique cannot easily be reused in different domains.

3.3.3 Graphical Models

Graphics are often used to model computer systems in order to better understand a concept or a problem as well as to describe the solution. Two types of graphics are popular in computer science: the Unified Modelling Language (UML) and Entity Relationship (ER) diagrams.

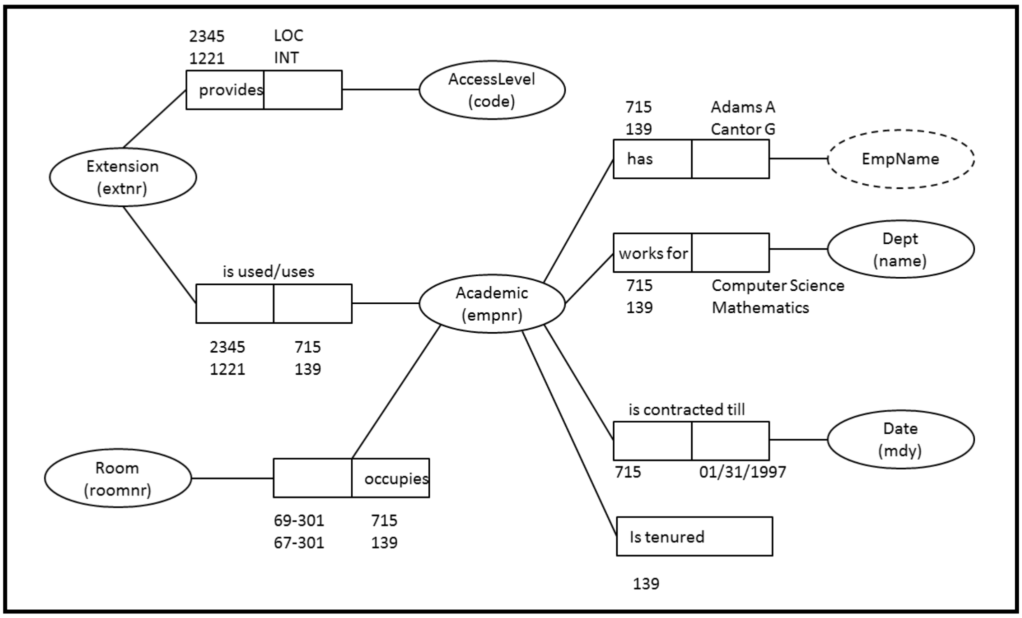


Figure 3.1: Example of object-role Modelling (Halpin, 1998)

An example of the use of a graphic model to design a context-aware application is shown in Figure 3.1. The context-aware application manipulates information about academics in a university. This information includes: their biographical details, the department for which they work, their contract details, the office that they occupy and the access levels. A specific syntax is used in order to draw the model. Entities (objects) are represented with an ellipse symbol and relationships (roles) are represented by rectangles. There exist several variants of this approach such as context modelling language (CML).

CML (Henricksen & Indulska, 2006) provides a graphical notation suitable to support analysis and to formally specify the context requirements of a context-aware application. This is built to capture different types of context information (static, sensed, derived and profiled). Information used to derive the context is provided directly by the user. CML can also handle imperfect and conflicting context information. Dependencies between the types of context facts can be captured, as well as the history and constraints about these context facts.

This approach is criticised for not supporting interoperability, that is, the ability to share information on context with other context-aware applications. All context types are represented as atomic facts (Bettini *et al.*, 2010).

3.3.4 Object-oriented Models

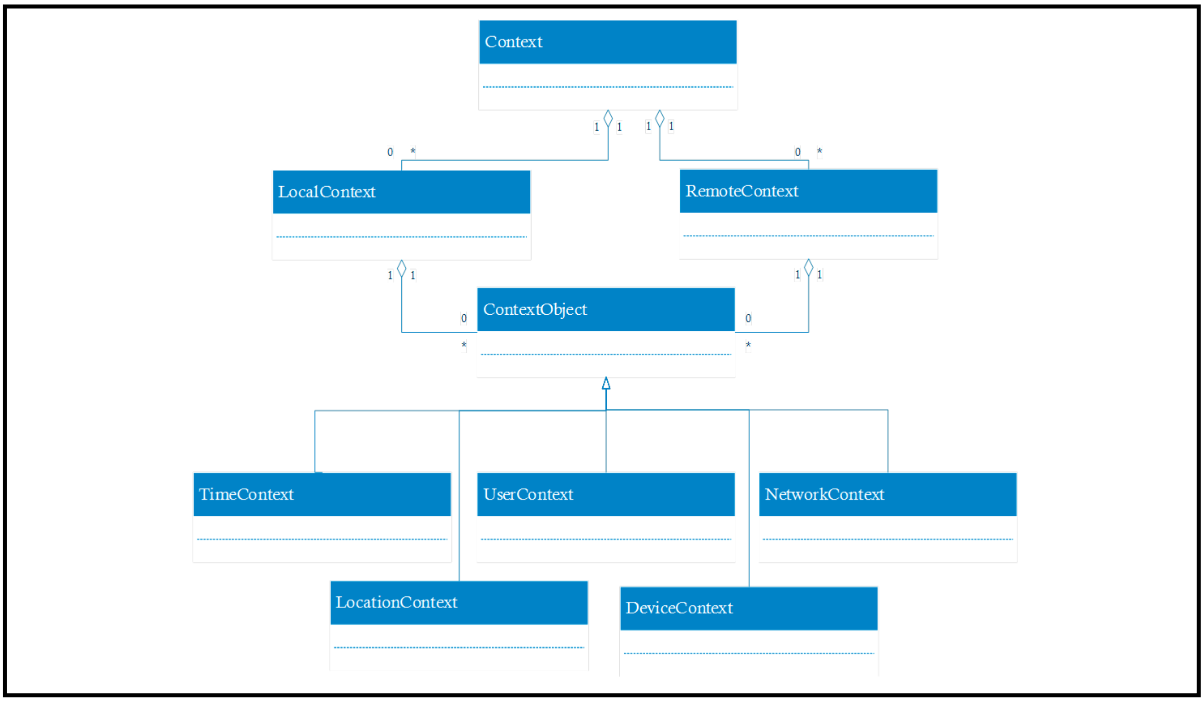


Figure 3.2: Hydrogen an object-oriented Model - Adapted from Hofer *et al.* (2003)

Object-orientation is a design paradigm that is characterised by encapsulation, inheritance and polymorphism. Object-oriented models for context-aware applications aim to use the benefits of object-orientation. Context information, such as location and ambient noise, are represented by objects. Details of the context and processing are encapsulated in the object. This allows the developer to reuse the code.

Examples of such models are *Hydrogen* (Hofer *et al.*, 2003), *Active Object Model* (Cheverst, Mitchell & Davies, 1999) and *Nexus Augmented World* Model (designed for context integration and schema evolution) (Niklas & Mitschang, 2004).

Figure 3.2 depicts the architecture of *Hydrogen*. The Unified Modelling Language (UML), a graphical language, is used to visualise the model. The context class is either local or remote; *RemoteContext* and *LocalContext* are linked to the main context class by an aggregation association. The class *ContextObject* is an abstract class that any specific context class (*TimeContext*, *UserContext*, *NetworkContext* etc.) can instantiate.

3.3.5 Logic-Based Models

Logic-based models are characterised by their high degree of formality. These models introduce the following concepts: *facts*, *expressions* and *rules*.

A fact is information used to determine the context, while a rule describes the relationship between several facts. An expression is a formula that is used to derive the context from available facts and rules.

A logic-based system is used to manage the above concepts by using several operations on facts (addition, update and deletion). The inference process can be used to derive new facts based on existing rules in the systems. The contextual information needs to be represented in a formal way as facts (Chen & Kotz, 2000).

3.3.6 Ontology-based Models

The word *ontology* comes from the Greek term *ontologyo*, which refers to the subject of existence. A short definition of ontology would be a specification of a conceptualisation. In knowledge management, ontology is referred to as the shared understanding of some domains, which are often conceived as a set of entities, relations, functions, axioms and instances (Wang, Gu, Zhang *et al.*, 2004). Ontology models combine object-oriented and logic models. Ontology models are well-known for their ability to facilitate the sharing of knowledge amongst entities. The knowledge can easily be retrieved by using logic inference.

Ontologies have been extensively used for modelling contextual information due to their high and formal expressiveness and the possibilities for applying ontology reasoning techniques. Various context-aware frameworks use ontologies as underlying context models (Chen, 2004, Wang *et al.*, 2004, Feld & Muller, 2011).

In order to develop a context-aware application using ontologies, the web ontology language (OWL) (Horrock, 2011) was adopted by the World Wide Web Consortium (W3C) as a standard for the semantic web and semantic web services (World Wide Web Consortium, 2009). This language resulted from an evolution of DARPA agent mark-up language + ontology interchange language (DAML+OIL), which extends the resource description framework (RDF) and RDF Schema with richer primitives. OWL tends to be better than the previous languages in its ability to represent machine interpretable content on the Web in an easier way (Da Costa, Laskey & Laskey, 2008).

By using OWL-description logic (DL) it is possible to model a particular domain by defining classes, individuals, characteristics of individuals (data type, properties) and relations between individuals (object properties).

The design of a mobile, context-aware ICCS should preferably use techniques and data representations that make optimal use of the mobile phone resources (processor, memory and battery). Ontology-based and logic-based models are very powerful; however, a high processing power is needed to implement these models. This could have a negative impact on the battery life. Key-values and object-oriented models have a good potential for mobile ICCS. A key-value model can be used for data collection and an object-oriented model when the mobile ICCS uses the context for adaptation. Various classes of such a model can encapsulate common operations and can easily be re-used in different applications. The architecture of a context-aware application should be loosely coupled to add flexibility, which is needed when extending or maintaining the application.

3.4 Approaches used to Design Context-Aware Applications

Several approaches have been used to design context-aware applications. This section discusses two approaches that are frequently used for context-aware applications. These approaches are often referred to as the *Context Toolkit* and the *Context Middleware*.

The Context Toolkit was introduced in order to provide a framework for development of context-aware applications. As depicted in Figure 3.3, the context toolkit includes sensors and their widgets, an aggregator, interpreters and the application that adapts to the context. Widgets are pieces of software responsible for hiding the complexity of the sensor from the developer. For example Android's *SensorListener* is a widget provided to the developer to interact with sensors embedded on Android mobile devices.

Interpreters raise the level of abstraction of sensor information. For example, a location sensor may provide geographical coordinates to the system, the interpreter will analyse the geographical coordinate to return street names. An interpreter can get information from several sensors in order to produce new context information. Interpretation can also be performed by applications; the separation of interpreters from applications has the benefit of enabling the reuse of interpreters by different applications.

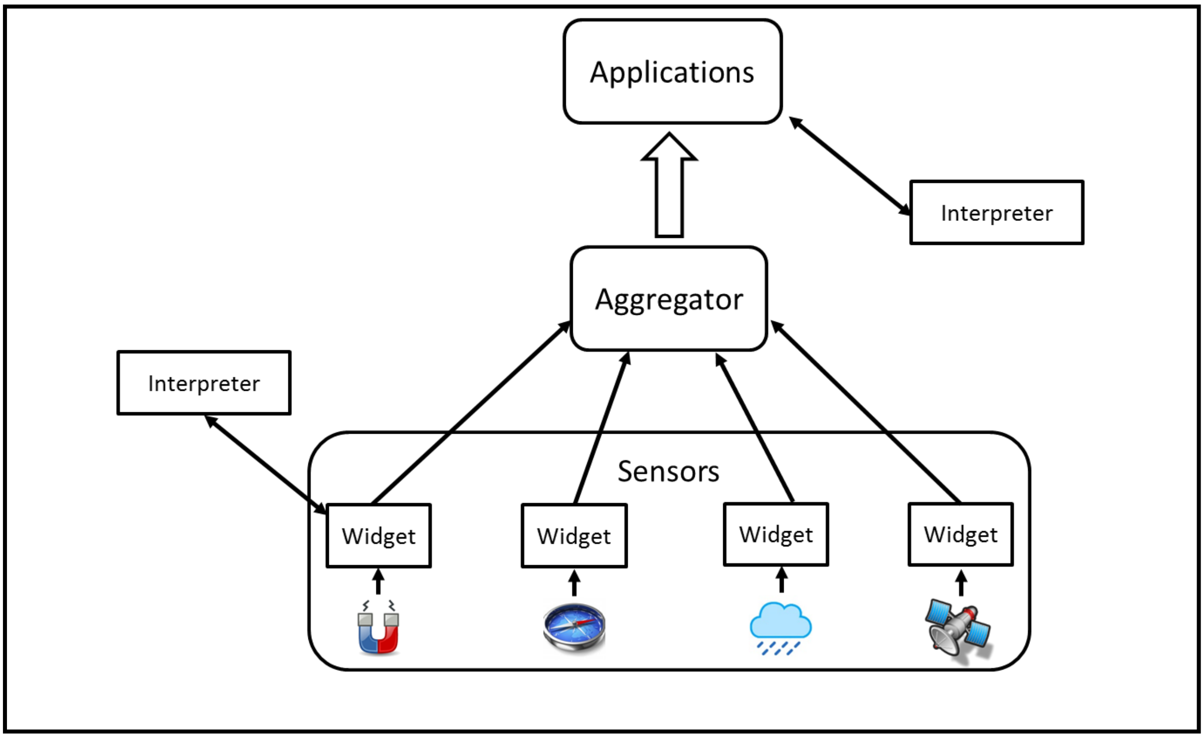


Figure 3.3: The Context Toolkit - Adapted from Dey *et al.* (2001)

Aggregators collect related context information in order to facilitate their use by applications. Aggregators are often used in distributed environments where several applications work remotely with the same set sensors. In this case the application does not have to query every sensor widget and rather query the aggregator.

The second approach, depicted in Figure 3.4, is the Context Middleware; this approach always includes a middleware that has the responsibility of aggregating context information from sensors.

The lowest layer is the set of sensors that are going to be used; it is called the Adaptor layer. The management layer is middleware, which has a main component, the ContextServer that enables applications to subscribe or retrieve context information. The upper layer is the Application Layer; it is made of applications that use context provided by the middleware. Then the context is inferred and made available for any client that needs it. An example of this approach is the *Hydrogen* Context Framework (Hofer *et al.*, 2003).

A similar approach called context broker agent (CoBrA) was proposed (Chen, Finin & Joshi, 2003). This approach is highly distributed. CoBrA provides an agent called the *Context Broker* to manage and maintain a shared model of context information. The context brokers can infer context knowledge that cannot be easily acquired from the physical sensors and can detect and resolve inconsistent knowledge that often occurs as the result of imperfect sensing.

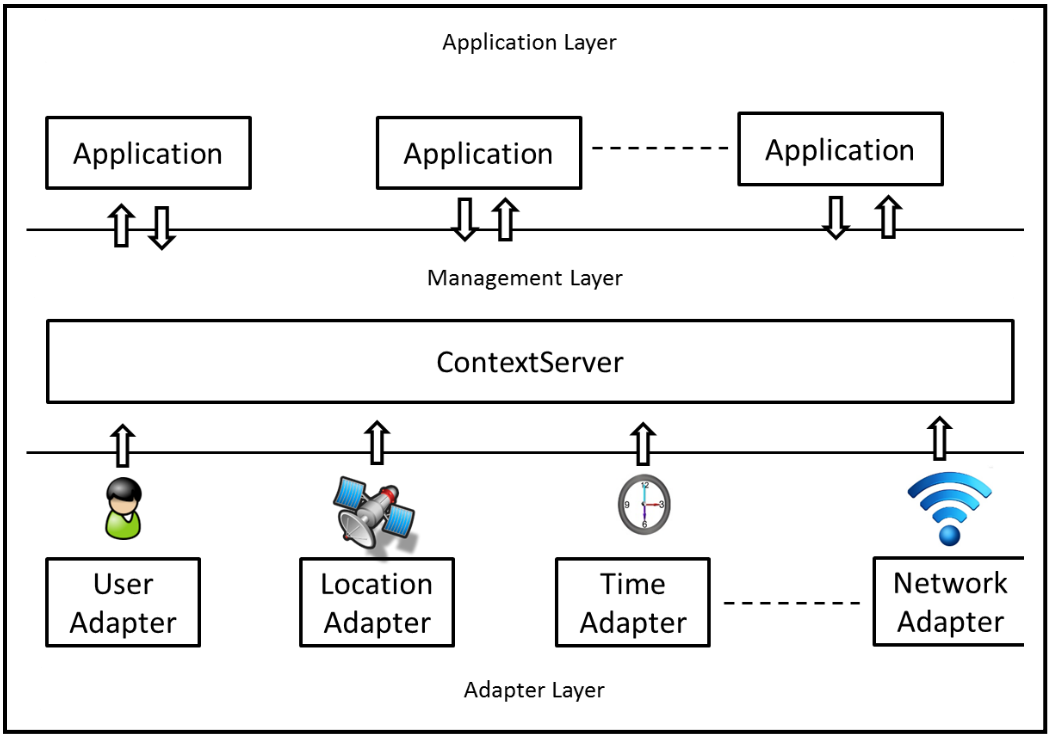


Figure 3.4: The Context Middleware - Adapted from Hofer *et al.* (2003)

The first step in designing context-aware application is to get the context information from a source. The following section discusses the source of context information.

3.5 Sources of Context Information

Definitions of context-awareness always mention the context information. This shows how critical the sources of information are to any context-aware application. It would not be possible to determine the context without an information context. Sources of information can be grouped into five categories. These include geographic (static), dynamic (sensed), informative (derived or static), technical (profiled or static) and personal (profiled) (Henricksen & Indulska, 2006). Geographical context includes location information that can be retrieved from a map. Dynamic context is often understood as sensed context since sensors are used to retrieve information. All these types of information can be obtained from mobile phones.

In the case of sensed context information, several frameworks can be used for context-data acquisition (Chen, Finin & Joshi, 2004). These include direct sensor access, middleware infrastructure and context servers.

- *Direct sensors*: These have access to sensors, which are built into the device. In this case raw context data cannot be shared with other entities willing to re-use these data,
- *Middleware infrastructure*: This allows other applications to share sensed information. This can be achieved locally using a framework, which encapsulates low level sensor data,
- *Context servers*: These allow multiple accesses to the sensed information, which makes it possible to share information among several clients.

Middleware frameworks that provide sensor data for mobile context-aware applications to be used in a car are difficult to find. Direct sensor access can be appropriate for a mobile context-aware application. A layer of the application will be responsible for collecting raw sensor information and processing it so as to obtain usable data.

The following subsections describe the most common sensors used to retrieve context information. This includes accelerometers, gyroscopes, compasses, GPS, microphones and light sensors.

3.5.1 Accelerometer

Accelerometers are electronic devices used to measure the multidimensional acceleration of an object. Acceleration is the change of velocity over time; this provides evidence for any variation of speed.

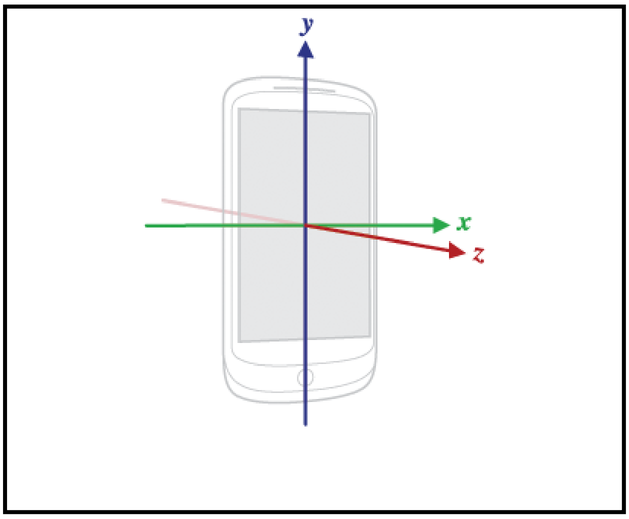


Figure 3.5: A Three-Axis Mobile Device (Google Inc, 2008)

Accelerometers are sensors that measure the acceleration of a device; this is often given on all three axes: X, Y and Z. The absolute acceleration is calculated with the formula $a = \sqrt{x^2 + y^2 + z^2} - g$, where g is the acceleration of gravitation $(9.8ms^{-2})$. The vertical acceleration is given by: $a = y - g$. As depicted in Figure 3.5 the acceleration is often measured on each of the three axes of the device.

Accelerometer data are likely to be degraded by noise (that is, undesired data), for example, when a mobile phone is on a table, the accelerometer measures X, Y and Z accelerations almost equal to zero. This noisy data can be corrected by using low or high level filtering techniques such as the Kalman filter (Welch & Bishop, 2000). The values should stay more or less stable while the device is on the table. However, this is not the case because of the noise. Gravity also affects accelerometer values negatively. The linear acceleration is a virtual sensor found on Android devices. This sensor removes the gravitation from the acceleration.

The accelerometer has been used to infer several kinds of contexts. For example, *Phone Point Pen* (Agrawal, Constandache, Gaonkar *et al.*, 2011) is a system that uses the in-built accelerometer in mobile phones to recognise human writing. By holding the phone like a pen, a user should be able to write short messages or draw simple diagrams in the air. The acceleration due to hand gestures can be translated into geometric strokes and recognized as characters.

Accelerometer sensors embedded in mobile phones have been used extensively to infer context. For example, the detection of potholes using accelerometer data was proposed by (Mednis *et al.*, 2011) using a range of algorithms. These algorithms assume that the phone is in a flat position in the car. The system was implemented on an Android mobile phone and when evaluated, revealed that the difference of two consecutive vertical accelerations was the best way of detecting potholes with a 99% accuracy rate.

3.5.2 Gyroscope

Gyroscopes are electronic devices embedded in objects used to measure the angular velocity (tri-axial) of these objects (Lane, Miluzzo, Lu *et al.*, 2010). Gyroscopes are becoming very popular in mobile devices. The Wii video game controller released by Nintendo in 2006 was the first popular application of gyroscopes. This game controller allows players to use gestures in order to control the game.

Gyroscopes tend to accumulate a lot of error, which makes it difficult to use them alone. The gravity that affects the accelerometer does not affect the gyroscope. The corrected gyroscope is a virtual sensor found on Android devices. The corrected gyroscope eliminates the drift from the raw gyroscope values. It is common to combine the gyroscope with the accelerometer to compensate for their respective disadvantages.

Gyroscopes have been used to design gesture-based interaction techniques for ubiquitous computing (van Tonder & Wesson, 2010). This work presented novel gesture-based interaction techniques for panning and zooming maps. The results showed that the user experience and the performance were enhanced, especially for selection tasks.

3.5.3 Compasses (Orientation)

Compasses are devices used to determine geographic direction (North, East, West and South). Compasses are usually made of a magnetic needle or needles horizontally mounted or suspended and free to pivot until aligned with the magnetic field of the earth. Compasses can be used to detect changes in orientation and motion. The compass is used mostly to determine rotation around the X, Y and Z axes. These can be used together with other sensors such as gyroscopes in order to compensate for noisy input.

Compasses embedded in mobile phones provide three measures, which represent angles in degrees: the azimuth (Z), the pitch (X) and the roll (Y). The azimuth is the angle between the magnetic north direction and the Y-axis, around the Z-axis (0^o to 359^o): 0^o = North, 90^o = East, 180^o = South, 270^o = West. The pitch represents the rotation around the X-axis (-180^o to 180^o), with positive values when the Z-axis moves toward the Y-axis. Finally, the roll represents the rotation around the Y-axis (-90^o to 90^o), with positive values when the X-axis moves toward the Z-axis.

3.5.4 Global Positioning System

A global positioning system (GPS) is a navigational system involving satellites and computers that can determine the latitude and longitude of a receiver on Earth. This is done by computing the time difference for signals from different satellites to reach the receiver (Kaplan & Hegarty, 2005). The GPS is controlled by the United States (US) Department of Defence that maintains a constellation of 24 satellites. GPS rely on the strength of the signal from satellites to calculate the latitude and the longitude of the device accurately. One of the major shortcomings of GPS is the unavailability of the signal indoors.

Information, such as travelled distance and speed, can be calculated from the readings of the GPS. Mobile operating systems, such as Android, have developed some APIs that help programmers to obtain GPS information such as the speed and the altitude (Android Developers, 2012). The Android *LocationManager* provides access to location information through the GPS and the mobile network.

GPS has the advantage of being more accurate than other location sensors such as mobile networks (Cell ID). However, it involves extra costs and consumes battery. The availability of GPS information can also be intermittent in cities owing to satellite low visibility around high buildings.

3.5.5 Microphone

Some applications require a noise level in order to adapt to a specific volume. An example of adaptation could be to increase the volume of the phone in a noisy place and to decrease it when the user is back to a normal noise level. The ambient noise is measured in decibels (dB).

Mobile devices record sound through a microphone and process the sound to measure its volume. Mobile operating systems, such as Android, do not provide direct access to ambient noise information; this has to be done by the developer after obtaining and calibrating information from the microphone of the device.

Noise level (dB)	**Description**
0 dB	Threshold of hearing
60 dB	Business office
80 dB	Shop noise
94 dB	Jackhammer
100 dB	Large truck
120 dB	Aeroplane take off
140 dB	Threshold of Pain

Table 3.2: Mapping of Noise Levels to Real World Situations (Cowan, 1993)

The formula $dB = 20\,log\,\frac{P}{P_0}$ is used to calculate the volume of noise. P represents the power value or the intensity of the sound. P_0 is the initial power value (no noise). Table 3.2 shows an example of noise level measured in a real world situation. An application can use such information to infer in which environment the user is.

3.5.6 Light Sensors

Light sensors measure the level of ambient light in SI lux units. Lights sensors are becoming standard and are integrated in smartphone operating systems such as iOS, Android, Symbian and Windows Phone 8.

On Android devices, the ambient light sensor is used to adjust the brightness of the screen and hence to optimise the battery power. In automotive applications, the light sensor could be used to determine whether the road visibility is poor or not. This can help to provide information about visibility. The time of day might not be sufficient in some situations, for example, when driving under a long bridge or tunnel during the day.

3.5.7 Web Services

The W3C defines a web service as "*a software system designed to support interoperable machine-to-machine interaction over a network*". Web services have an interface, which can be parsed easily by most browsers (specifically Web Services Description Language, known by the acronym WSDL) (Curbera, Duftler, Khalaf *et al.*, 2002).

Several web services are available on the Internet. Some provide updated information on weather or traffic. Weather current conditions and forecasts can be obtained by using the following web services:

- *Google weather API*: www.iGoogle.com: service discontinued since 2012,
- *Yahoo weather API*: provides forecasting and current weather information,
- *Yr.no weather API*: provides free access to meteorological data using web services,
- *Weather underground*: a commercial weather service that provides an hourly forecast for several weather stations around the world.

Current weather conditions available from web services include the following: wind power (kilometre per hour), humidity (percentage), pressure (hecto Pascal), visibility (kilometre), ultraviolet (UV), sunrise (for example 5:18 AM South African Standard Time), sunset, length of day, moon rise, moon set and moon phase. More aggregate information can also be obtained, including: partly cloudy, cold, foggy, heavy rain, mostly cloudy, snowing and lightning. This could be used to determine whether a driving situation is safe or not.

Most devices used to sense contextual information are not always hundred percent accurate. This inaccuracy must be taken into account and rectified in order to use more relaiable context information. This requires pre-processing of information coming from sensors prior to the information being used. Sensor fusion can be used to check the consistency of the data gathered by integration into a context model. On the other hand, some data are meaningless unless combined with other data; in this case, sensor fusion can help to obtain data that cannot be obtained directly from a sensor.

3.5.8 Sensor Fusion

Information from various sensors may be inaccurate. Some sensors can overlap or conflict with each other. Some sources of sensor data, such as GPS, can be unavailable at times. This has led researchers to introduce *sensor fusion* in order to address the shortcomings of isolated sensors. Sensor fusion can be defined as information processing that collects sensory data from multiple sensors or from the same sources over a period of time. This can be used to produce knowledge that is otherwise not obtainable, or that is more accurate or more reliable than information gathered from single sensors (Klein, 2004).

For example, GPS data can be combined with cellular network information in order to improve the accuracy of the captured location. Compasses also help by providing orientation information from the magnetic field of the Earth. Thus, when GPS satellite signals are not available, the system can determine a generalised neighbourhood from the network-based method and provide that information immediately for use by the system. Once the GPS receiver is able to detect satellite signals, the device can then provide this more accurate location to the system.

The following sections discuss several techniques used to combine information from isolated sensors. This includes the Dempster-Shafer algorithm and fuzzy logic (Frigui, Zhang, Gader *et al.*, 2012).

3.5.8.1 Dempster-Shafer Theory

The Dempster-Shafer theory is a mathematical theory of evidence based on a belief function and plausible reasoning (Wu, Siegel, Stiefelhagen *et al.*, 2002, Zhang *et al.*, 2009). The Dempster-Shafer decision theory is a generalised Bayesian theory. It allows distributing support for a proposition, not only to the proposition itself, but also to the union of propositions that include it. In a Dempster-Shafer reasoning system, all possible, mutually exclusive context *facts* (or *events*) of the same kind are enumerated in the frame of discernment. The frame of discernment is often denoted by the letter Θ. For example, if we know that there is a person in an instrumented room and we want to recognise whether that person is the already-registered user *A*, user *B*, or somebody else, then our "frame of discernment" about this person is:

$$\Theta = \{A, B, \{A, B\}, \{someone\ else\}\}.$$

This means that the person is "*user-A*", "*user-B*", "*either user-A or user-B*", or "*neither user-A nor user-B, must be somebody else*". Each sensor, for example sensor S_i, will contribute its observation by assigning its beliefs over Θ. This assignment function is called the *probability mass function* of the sensor S_i, denoted by m_i. So, according to the observation of sensor S_i, the probability of the event "the detected person is user A" is indicated by a *confidence interval.*

Figure 3.6 depicts an architecture used to combine the information from several sensors with the Dempster-Shafer algorithm. Prior to being sent to the sensor fusion mediator, sensor data are pre-processed by a specific module called an interface widget. After the fusion, information is gathered in the context data entity to be interpreted by artificial intelligence algorithms so that they can be meaningful to the application.

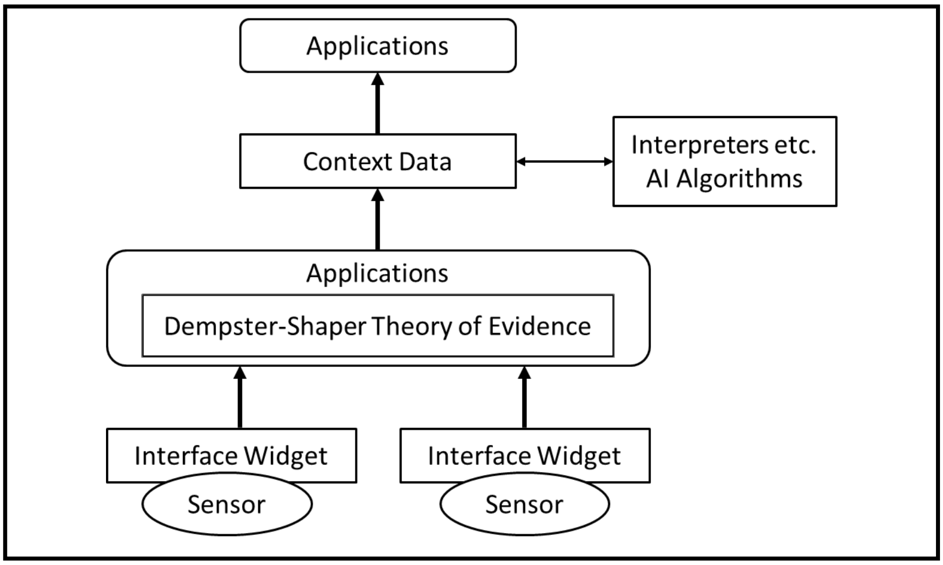


Figure 3.6: Sensor Fusion Architecture – Adapted from Wu *et al.* (2002)

A similar architecture can be used for a mobile, context-aware application. Context data can be obtained using different sensor fusion techniques. AI algorithms can use context data in order to determine the context, which will be used by the application.

3.5.8.2 Fuzzy Logic for Data Fusion

Mathematical logic is generally used to model situations. This logic consists of attributing *false* or *true* values to predicates. This is made under the assumption that everything can be defined precisely, which is not the case in real world situations. For example, the notion of speed cannot be defined precisely. It is difficult to say if an entity is fast or slow because it depends to what this entity is being compared.

Fuzzy logic differs from classical logic in that statements are no longer labelled as true or false. In Boolean logic an object takes on a value of either 0 or 1; in fuzzy logic, a statement can assume any real value between 0 and 1.

The fuzzy set introduces the notion of *membership function*. A membership function indicates the degree of membership of an element to a specific set. Assuming A is a set and X the universe, the membership function of the set A will be $\mu_A(x)$.

$$\mu_A(x): X \rightarrow [0,1]$$

Where $\begin{cases} \mu_A(x) = 1 & if\ x\ is\ totally\ in\ A; \\ \mu_A(x) = 0 & if\ x\ is\ not\ in\ A; \\ 0 < \mu_A(x) < 1 & if\ x\ is\ partially\ in\ A \end{cases}$

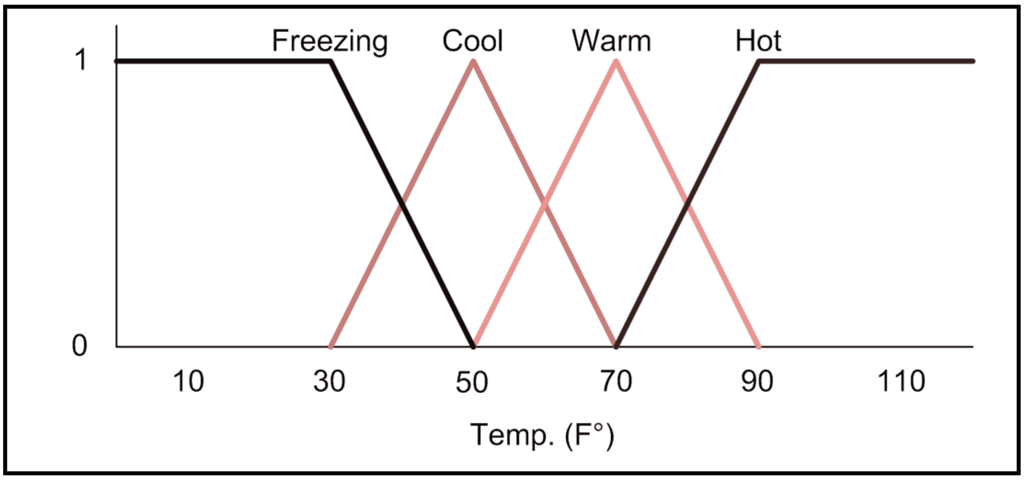


Figure 3.7: An Example of fuzzification of a system

In a fuzzy logic process all input has to be fuzzified, which means that a membership function has to be defined for each of the inputs. Figure 3.7 shows an example of fuzzification of a set that contains the inputs freezing, cool, warm and hot.

Mobile phone sensors provide a large amount of data. Some of the data could sometimes have poor accuracy. Fuzzy logic can help in combining the data carefully. The membership function has to be well defined so that the result is realistic.

3.6 Determining the Context

When designing a context-aware application, the most important task is to determine the context accurately. Several techniques are often used to help developers to make appropriate use of the acquired input in order to infer the current context accurately. In the following paragraphs AI techniques that are used to determine the context are discussed. These techniques include Bayesian networks, support vector machines, naïve Bayes, nearest neighbours, decision tree and artificial neural networks.

3.6.1 Bayesian Networks

A Bayesian Network (BN) is a supervised machine learning technique based on the theory of probability. A BN is a graphical model for probabilistic relationships amongst a set of variables, a Conditional Probability Table (CPT).

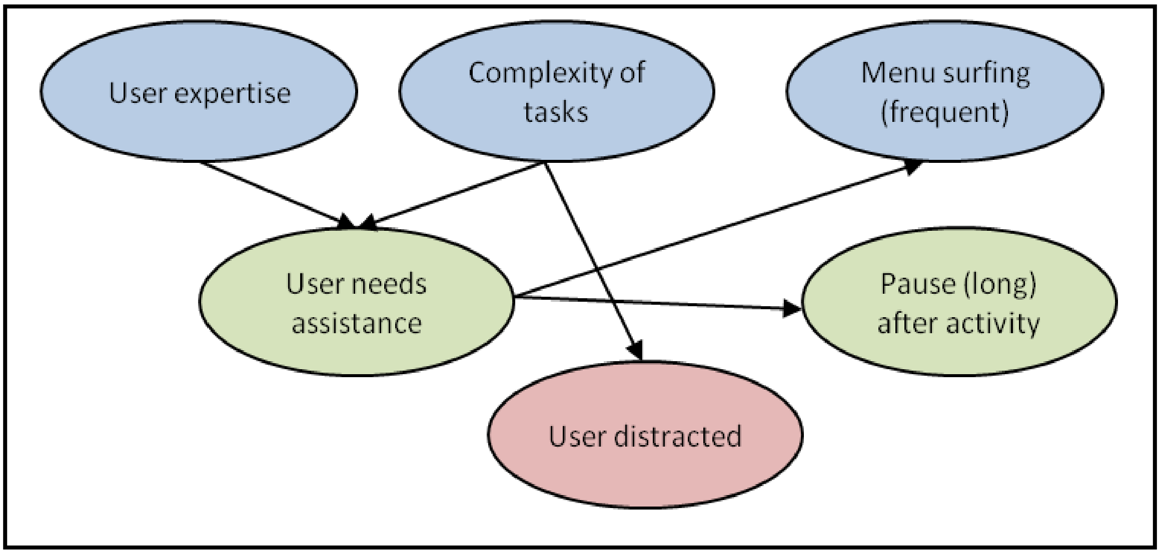


Figure 3.8: Example of a Bayesian Network (Horvitz, Breese, Heckerman *et al.*, 1998)

As depicted in Figure 3.8, a BN is a directed acyclic graph whose nodes represent discrete or continuous variables. Arcs represent causal dependencies between connected nodes. Node A is called a *parent* of node *B* and *B* is a *child* of *A* if a direct arc is present from *A* to *B*. The arc implies that *B* depends directly on *A*. A node is called a *root* node if it has no parent and depends on nothing. Each root is associated with a prior probability specified by experts. Each of the other nodes is associated with a CPT, which records all possible combinations of the values of all its parents when the variable represented by that node is discrete.

Figure 3.8 shows a portion of the BN used by Microsoft to design *Lumière* (Horvitz *et al.*, 1998), a context-aware application that detects when a user needs help. This network states that the need of the user for assistance from the system will depend largely on the expertise of the user in using the system and the difficulty of the task that the user is currently dealing with. In turn, the need of the user for assistance will influence the occurrence of behaviours of the user, such as surfing through menus or pausing after performing some actions.

Several benefits can be gained when using a BN. BN can handle incomplete data, which often happens when a source of context information is temporarily unavailable. Causal relationships amongst variables can also be derived by using BN. Over-fitting is an issue faced by several learning techniques preventing the model obtained from being generalised. BN can avoid this issue.

3.6.2 Naive Bayes

Naive Bayes are similar to Bayesian networks except that the algorithm assumes that each feature from the training set is independent of any other feature. Therefore, there is no need to learn a joint distribution of all features. The number of parameters is therefore dramatically reduced from $2(2^n - 1)$ to $2n$, n being the number of features in each vector of the training set (Mitchell, 1997). When applied to a suitable problem, naïve Bayes can be very efficient and can run faster than some more complex algorithms. Some variants of naïve Bayes are also available such as Gaussian naïve Bayes.

3.6.3 Support Vector Machines

Support Vector Machines (SVM) are classified as a supervised learning method. Supervised learning methods are often used for classification and regression.

For example, a classification problem could be to determine whether a set of data can correspond to whether a car is moving or not moving. Similar to other supervised learning techniques, SVMs build a decision model that predicts in which category a new example will belong. SVMs are often represented in a two dimensional graph (x and y axis), where data to be separated are points drawn in two different colours.

In theory, a linear SVM comprises a set of support vectors z and a set of weights w (Figure 3.9). The calculation of the output given N support vectors is given by the formula $f(x) = \sum_{i=1}^{N} w_i \langle z_i, x \rangle + b$.

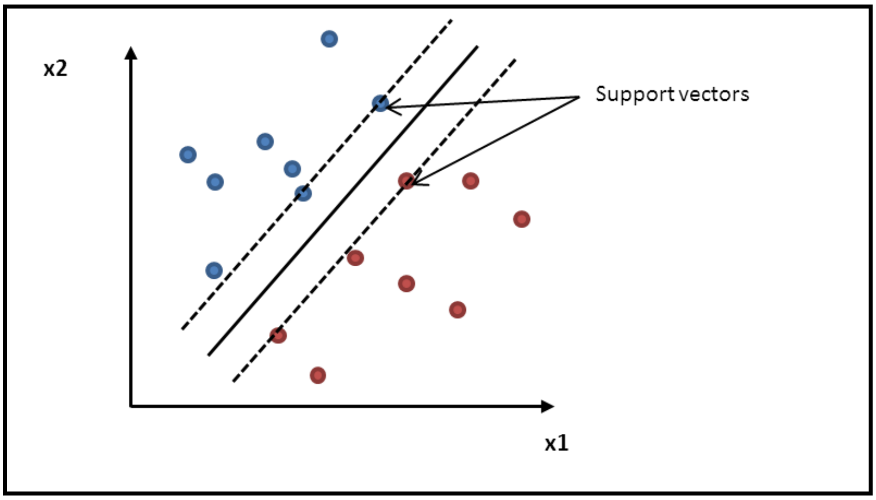


Figure 3.9: Separation of Two Categories of Data by SVM (Bishop, 2006)

A decision function is then applied to calculate the output $f(x)$. The result is used to determine the class in which the input x belongs. Usually, the function $sign(x)$ is used, so that positive outputs are taken as one class and negative outputs are taken as the other.

SVMs, as described, perform a linear separation of the data. In real life situations, it is not always possible to separate data with a straight line. Researchers have introduced the kernel trick as a way to design non-linear SVMs (Schölkopf, Smola & Müller, 1998, Schölkopf & Smola, 2002).

Kernel functions, often used to design kernel SVMs, include linear kernels, polynomial kernels and Gaussian kernels. The linear kernel is the simplest kernel function and can be expressed in the form $K(x, y) = x^T y + c$, where c is a constant. Kernel algorithms using linear kernels are often equivalent to non-kernel techniques, such as Principal Component Analysis (PCA). Polynomial kernels are well suited for normalised data; the function is $K(x, y) = (\alpha x^T y + c)^d$. Gaussian kernel functions are more complex; K is written as $K(x, y) = \exp\left(-\frac{\|x-y\|^2}{2\sigma^2}\right)$. This is a radial basis function, which is also used to model some neural networks. The Gaussian kernel function is more suitable in situations where little prior knowledge is known about the data.

There are a number of software packages available for implementing a SVM. These include LibSVM (C++) (Chang & Lin, 2011), SVMLight (C) (Joachims, 1999), Spider (Matlab) (Weston, Elisseeff, BakIr *et al.*, 2005), and Weka (Java) (Hall, Frank, Holmes *et al.*, 2009).

These algorithms require high computation power; therefore they should not be run online on a mobile phone.

3.6.4 Artificial Neural Networks

Artificial neural networks (ANN), also known simply as neural networks, are a type of supervised learning algorithm. These models are built like biological neurons in the human brain. The connections of artificial neural networks represent axons and dendrites in biological neurons. Each connection has a weight, which represents the synapses of biological neurons. The threshold provides an approximation of the activity of the soma (Jain & Mao, 1996).

Neural networks can be described as non-linear statistical data modelling tools that are used to model complex relationships between inputs and outputs or to find patterns in data. Neural network (NN) analysis is considered to be an alternative approach for the investigation of non-linear relationships in engineering problems. Figure 3.10 shows a three-layer NN, the input layer contains new data (e.g. speed, and turning angle) to be classified and the output layer specifies classes (e.g. distracted or not distracted).

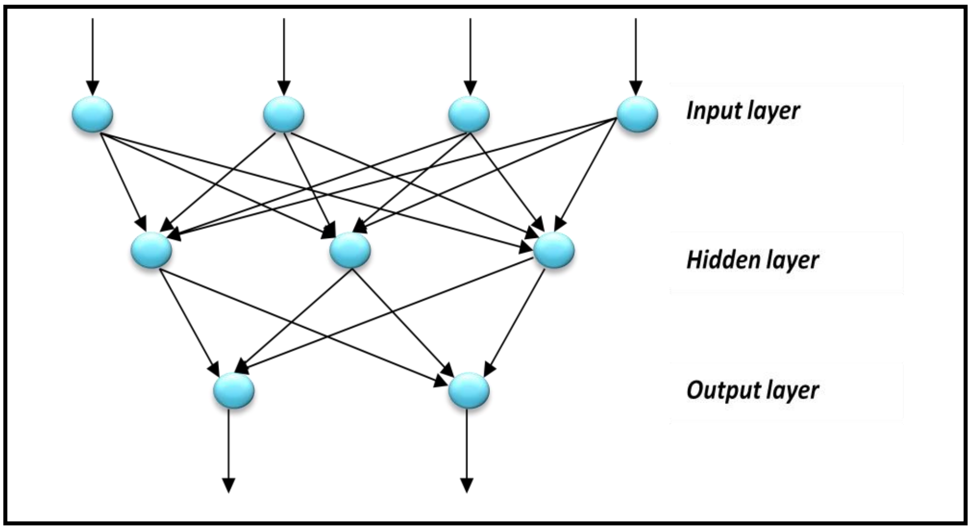


Figure 3.10: Example of a Three-Layer Feed-Forward NN (Jain & Mao, 1996)

As a supervised learning technique, ANNs need to be trained. There are several training techniques that can be used to train an ANN. These include back propagation, genetic algorithms and particle swarm optimisation.

Back propagation is an optimisation technique that aims to find out the best set of weights. The best set of weights is the one that has the minimum mean squared error. The mean squared error is calculated by using the formula $MSE = \sum_{i=1}^{P}(o_i - t_i)^2$, where o_i represents the output of the current neural network for the i^{th} pattern and t_i represents the target for the i^{th} pattern.

Genetic algorithms are based on the theory of evolution. The fittest members of a population are chosen to reproduce in order to improve the fitness of the next generation. After a certain number of generations, the population is expected to be very fit, which means close to the solution of the problem. When training a neural network with a genetic algorithm, each weight is a gene and the whole set of weights represents a chromosome. A population of chromosomes is randomly generated and is expected to move closer to the solution by using the fittest to reproduce and by modifying the population through mutation and crossover.

Particle swarm optimisation techniques model a swarm that moves in a specific direction, which represents the desired solution. The principle of swarm optimisation is that each member of the swarm should go in the same direction as the individual that performs better than they do. The direction and the velocity of each member are then modified so that they correspond to the direction of the local and the global best individuals. For the purpose of training a neural network, each individual will be the set of weights of the neural network. After a certain number of runs, the global best individual will be fit enough to be considered for the neural network that minimises error the most.

Neural networks offer an alternative to regression analysis in the solution of nonlinear engineering problems. The advantages of neural networks over regression analysis are that in regression analysis, the analyst has to choose a model to fit the data, while neural networks are not required to pre-select a model, sufficient hidden nodes can provide the accuracy required for many different response surfaces.

Some research projects in the transportation industry have used neural networks successfully. Crashes can be predicted accurately by using a neural network trained by back propagation (Akin & Akbas, 2010). In this case, the following information was used as inputs to the neural network: day of the week, hour of the day, month of the year, weather conditions, light conditions, road surface condition, traffic control device, accident type (overturned, head-on, rear-end and angle turn), number of moving vehicles involved, type of vehicle, age of driver, age group, gender, intention of the driver, violations of the driver, contributing circumstances, drinking or drug use, object hit by vehicle, size of vehicle, visual obstruction and condition of vehicle.

The infractions that were committed by the driver included the following: speeding, moving slowly, failing to yield, wrong way, improper lane use, improper turn, no signal, improper backing and following too closely. The main findings were the following: 8% of accidents occur at intersections and interchanges; the last working day of the week had the highest number of accidents by 17.8%, which is 1.3 times higher than the first working day of the week. Accident occurrence during afternoon-peak hours (8.9%) was 1.9 times higher than the number of accidents during morning-peak hours. This can be explained by the fact that drivers may be affected by tensions and stress owing to a tiring day at work.

Neural networks are supervised learning techniques that are often used in machine learning to determine in which class example data belongs. Neural networks mimic the functioning of the human brain with interconnected nodes (neurons) structured as a network. Each connection between nodes carries a weight (Angelbrecht, 2007). A multilayer perceptron contains at least three layers. The input layer receives the vector to be classified; this will be passed through the hidden layer and finally, the output layer will determine the class. An activation function is often used to calculate the output of each neuron from the weighted input received. Identity function, step function and sigmoid function are examples of popular activation functions. The training phase aims to determine the set of weights that minimises classification errors. Back propagation, genetic algorithms and particle swarm optimisation are often used to train neural networks. The Waikato Environment for Knowledge Analysis (WEKA) implements a multilayer perceptron with learning rate and momentum that can be set prior to the experiment. WEKA also uses the Radial Basis Function Network (*RBFNetwork*), which is an implementation of the neural network using radial basis function activation functions.

3.6.5 Decision Trees

Decision trees are a class of algorithm that use a tree-like model of decisions to support decision making. The result can often be displayed graphically. New inputs are classified into known categories provided for the training (Supervised learning). At each node of the tree the algorithm chooses one attribute or feature that splits the data effectively. There are two types of nodes in a decision tree: decision nodes and leaves. Leaves are the terminal nodes of the tree and they determine the ultimate decision of the tree. Decision trees are very flexible as they can be used to model complex decision boundaries. Continuous and discrete attributes can be handled by decision trees.

Decision trees are good classifiers when the training set contains missing values and outliers (Quinlan, 1993, Kim, Zhang, Wu *et al.*, 2011). This is made possible by implementing a pruning process which reduces the number of nodes of the tree generated by the algorithm. Decision trees can also handle a large amount of data and, unlike for many machine learning algorithms, the resulting model can be subject to interpretation. However, when using a small set of training data, decision trees can perform poorly.

Several implementations of decision trees are available in commercial and open source tools. WEKA (Hall *et al.*, 2009) implements several decision tree algorithms, including J48 and LMT. J48 is an open source Java implementation of the C4.5 decision tree generator (Quinlan, 1993); WEKA also provides a variant J48graft. The C4.5 is an improvement of the ID3 algorithm as C4.5 handles both continuous and discrete classes. Logistic model tree (LMT) is a combination of logistical regression and tree induction.

3.6.6 Nearest Neighbours or Instance Based Classifiers

Nearest neighbour algorithms work as follows: all training data are stored in a set **S,** which is referred to as the training set. The set consists of a list of feature vectors with an indication of the class to which they belong. The training set will be searched for in the vector most similar to X, which is the non-classified vector. X will therefore be classified in the same class as the most similar vector in **S** (Mitchell, 1997). The similarity is measured by the distance between vectors. The Euclidian distance is often used; however, other distance functions such as Manhattan distance can be used. A variation of this algorithm, namely the nearest k neighbours, is often used to reduce the impact of outliers. Larger values of k address the issue of noise in the training set. WEKA provides an implementation of IB1 (Instant Based 1) and IBk, where IB1 denotes a nearest neighbour algorithm that selects the closest vector from the training set. IBk ($k > 1$) denotes a similar algorithm except for the fact that the prediction is made based on the nearest k neighbours. Classification using nearest neighbour algorithms have been used successfully in many in-car projects. This algorithm was used to detect the emotion of the driver in order to prevent the driver from distractions (Nasoz *et al.*, 2010).

The nearest neighbour model that is generated by the algorithm contains all of the training set that was used to generate it. It can be a disadvantage in terms of computational cost in the case where the training set is very large.

Another approach used in the field of machine learning is Boosting. Boosting is an iterative process using other machine learning algorithms to derive rough rules of thumbs that can be used for classification (Mitchell, 1997). *AdaBoost* is an implementation of a Boosting approach that is a meta-algorithm. *AdaBoost* is used in conjunction with other learning algorithms. This helps deal with noisy data. All learning algorithms involved complement each other. WEKA also provides an implementation of this algorithm.

In some experiments SVM performs better than other algorithms when determining driver distraction (Liang, 2009) with a specific set of input variables. However, it is not possible to understand how the decision is made as opposed to the case with decision trees. Nearest neighbour algorithms can also be useful for a mobile ICCS. These algorithms do not require a large set of training data to reach an acceptable level of performance.

The algorithms discussed above will be used in Chapter 5 where the most suitable algorithm(s) to determine the driving context (distraction level and driving events) will be determined.

3.7 Adaptation Effects

The definition of context-aware applications refers to the adaptation of the application to the current situation. Adaptations often affect the user interface, the interaction with the user, the resources used or the algorithm. Retroactive and proactive adaptations are the types of adaptation commonly used in context-aware applications (Subramanian & Chung, 2000).

3.7.1 Retroactive Adaptation

Retroactive adaptation relies on the user to make a decision about the adaptation. The information about the context is presented to the user who decides whether the interface will change or not. This technique is mostly used for interface changes as it will be difficult to apply it to resource adaptation.

Lumière (Horvitz *et al.*, 1998), which detects when a user needs help, uses retroactive adaptation. A window, presenting help options, pops up and the user can decide to click on one of the options to find out how to carry on with the task being performed. This mechanism does not interrupt the user and is believed to be less annoying.

Status sharing is another example of retroactive adaptation. Sharing status information from the *callee* to the *caller* (Lindqvist & Hong, 2011) has been used to mitigate driver distraction. The new concept car from Nissan has implemented several technologies to detect drunken driving, driver drowsiness and driver inattentiveness (Nissan Technology Magazine, 2007). The system issues recommendations to the driver in order to stop the car in a safe place.

3.7.2 Proactive Adaptation

Proactive adaptation decides on the action to be taken on behalf of the user. The system does not require the decision of the driver to perform the adaptation (Figure 3.11).

A study conducted by Ho & Intille (2005) suggested that proactive messages, such as messages delivered by a mobile computer when the user is transitioning between two physical activities (e.g. sitting or walking), may be received more positively than the same messages delivered at random times. The results showed that the perceived burden of context-aware mobile computing devices may be minimised by time-shifting some proactive messages to times when the user is already transitioning between different physical activities.

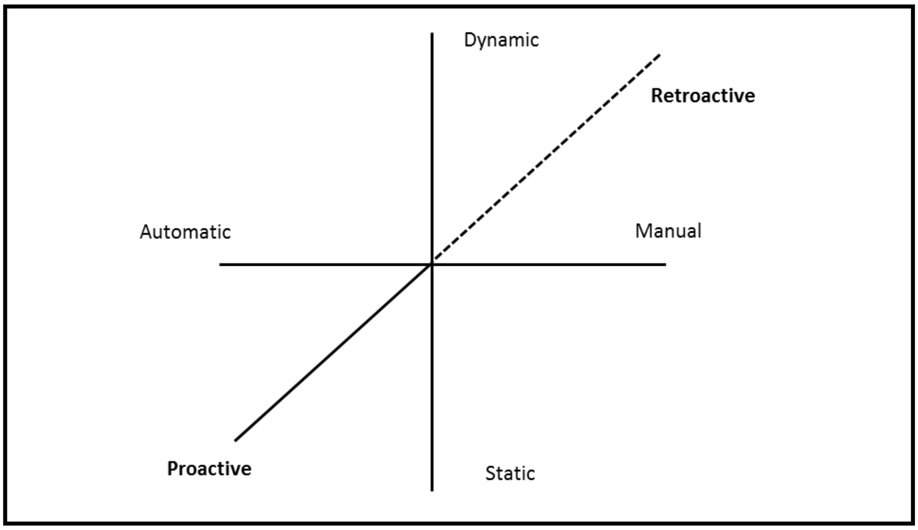


Figure 3.11: Axes of Software Adaptation (Subramanian & Chung, 2000)

Proactive adaptation is better in the case of a mobile, context-aware ICCS. Asking for confirmation from the driver can cause distraction even if it is done through speech interaction.

3.8 Requirements for Context-Aware Applications

The design of context-aware applications generally follows three steps. These include the acquisition of context information, the determination of the context and the use of the context to optimise the performance or the user experience. Research has identified a number of requirements that context-aware applications should meet in order to provide maximum benefits to users (Anagnostopoulos, Tsounis & Hadjiefthymiades, 2007). These requirements include the following:

- Context acquisition,
- Context aggregation,
- Context consistency,
- Discovery,
- Query,
- Adaptation,
- Reasoning,
- Quality indicators,
- Integration.

The structure of contextual information is also of high importance. Carefully choosing a context model will allow accurate context determination, which is fundamental for the success of any context-aware application.

Criteria for context modelling techniques (Moore, Hu, Zhu *et al.*, 2007) are:

- *Distributed Contribution (DC)*: must accommodate characteristics of ad-hoc networks and dynamic distributed systems because pervasive computing is often implemented in these environments,

- *Partial Validation (PV)*: addresses errors that can occur when defining relationships between entities,

- *Richness of Quality and performance (QUA)*: must support quality and richness indication because sensor-derived data are continuously available and depend on the quality of each context source,

- *Incompleteness and ambiguity (INC)*: must incorporate the ability to handle incomplete data by interpolation,

- *Level of formality (FOR)*: contextual facts should be described in a precise way so that a common understanding and interpretation exists,

- *Adaptability (AD)*: enables the use of the existing domain, systems and infrastructures.

Table 3.3 shows the evaluation of context modelling techniques in terms of the criteria listed above. A cross indicates that the criteria are not met by the modelling approach; a tick shows that the criteria are partially met and two ticks indicate that the criteria are fully met by the modelling technique. Ontology-based and object-oriented models are the only modelling techniques that meet all the requirements.

Modelling Approaches	**DC**	**PV**	**QUA**	**INC**	**FOR**	**AD**
Key-value models	✗	✗	✗	✗	✗	✓
Mark-up scheme models	✓	✓	✗	✗	✓	✓✓
Graphical models	✗	✗	✓	✗	✓	✓
Object-oriented models	✓✓	✓	✓	✓	✓	✓
Logic-based models	✓✓	✗	✗	✗	✓✓	✗
Ontology-based models	✓✓	✓✓	✓	✓	✓✓	✓

Table 3.3: Evaluation of Context Modelling Approaches (Moore *et al.*, 2007)

The key-value model is the most simple to implement but it only guarantees the adaptability. The mark-up scheme model is not flexible enough as it does not support erroneous context sources and incomplete data. The graphical model cannot be used in a distributed context, because errors cannot be addressed due to a lack of partial validation and the model does not handle incomplete data. The logic-based model does not do partial validation; poor quality context sources are not handled as well as incomplete data.

In terms of context reasoning, literature shows that Bayesian networks, support vector machines and neural networks can be used successfully. However, Bayesian networks require prior knowledge about the domain being studied. This can be difficult to obtain.

3.9 Conclusion

This chapter aimed to answer the second research question of this project *RQ2: What are the existing models for context-aware applications?* Models used to manipulate contextual information were discussed. These included key-value, mark-up scheme, object-oriented, logic-based and ontology-based models. It was determined that both object-oriented and ontology-based models met all the requirements for a successful context-aware application. Two approaches to designing context-aware applications were reviewed: a layered approach involving a Context Middleware and a Context Toolkit approach. These approaches are similar in the fact that they handle sensor information separately. The context aggregator and the context server help applications in sharing context information. While the sharing can benefit a mobile context-aware ICCS, the remote processing of the context may be a source of problems. The sending of information to the remote sever could use up the battery life of the mobile phone in a short period of time.

Several machine learning techniques were reviewed, which can be used to determine the driving context. Bayesian networks can achieve good results, however, they can be difficult to design and implement. The use of a well-trained artificial neural network could be the best option to determine the driving context. This is due to the simplicity of implementation and its offline character that will help in saving battery power. The support vector machine can also be considered to be a good option since it has been used successfully in similar projects (Liang, 2009).

In-car applications are good candidates for context-aware applications. This is due to the fact that cars are in a dynamic environment. The workload of the driver changes constantly as the driving situations change. Applications that are context-dependent could possibly prevent drivers from engaging in secondary tasks when their full attention is required on the road.

All sources of context information reviewed can be obtained by using a mobile phone. This suggests the use of a mobile platform in order to host such an application. Furthermore, the popularity of smartphones will facilitate access to such technologies in any car instead of having an embedded system.

The next chapter will use the requirements for context-aware applications in order to propose a new model for a mobile, context-aware ICCS. This model aims to support the design of an ICCS that minimises driver distraction and enhances the driving experience. A usability evaluation will be conducted in order to assess the feasibility of a speech-based, mobile ICCS.

Chapter 4: Proposed Model for a Speech-based ICCS

4.1. Introduction

This chapter aims to answer two related research questions. The first question, *RQ3: What should a model for a speech-based, mobile ICCS comprise?* will be answered by proposing a model for speech-based, mobile in-car communication system (ICCS). This model is called the Multimodal Interface for Mobile Info-communication with Context (MIMIC). The proposed model uses a speech user interface (SUI) that was chosen because it is one of the solutions that have been used by car manufacturers in order to reduce driver distraction (Section 2.7). A SUI helps drivers to keep their hands on the steering wheel instead of manipulating their mobile phones. When using an SUI, drivers also keep their eyes on the road.

The second question, *RQ4: Is a speech-based, mobile ICCS feasible?* will be answered by conducting a usability study. This usability study will help identifying possible issues with speech-based, mobile ICCS.

This chapter is organised as follows: Section 4.2 discusses the design of MIMIC as well as the implementation of a speech-based MIMIC-Prototype. The design of MIMIC contains the Input, Dialogue and Output modules. The implemented prototype was deployed on an Android phone. The usability evaluation of the SUI is discussed in Section 4.3, while Section 4.4 presents the results of the user study and a discussion of these results is presented in Section 4.5. Section 4.6 concludes this chapter.

4.2. Design and Implementation of MIMIC

The goal of this chapter is to design a model for a speech-based, mobile ICCS. The mobile aspect of the model is the use of the mobile device itself. Figure 4.1 shows the proposed model, which contains several components that interact with each other. These include the Input Module, the Dialogue Module and the Output Module. This model is based on the typical architecture for a speech-based application (Figure 2.5) and an existing model, called Multimodal Interface for Mobile Info-communication (MIMI) (Figure 2.7).

4.2.1. Input Module

The Input Module contains several important sub-modules. These are the Automatic Speech Recogniser (ASR), the Natural Language Understanding (NLU) unit and the mobile phone itself. The mobile phone of the caller can provide valuable information to the Dialogue Module (DM). The mobile phone of the callee (Peer), can share contextual information obtained from sensors, global positioning systems (GPS) and web services.

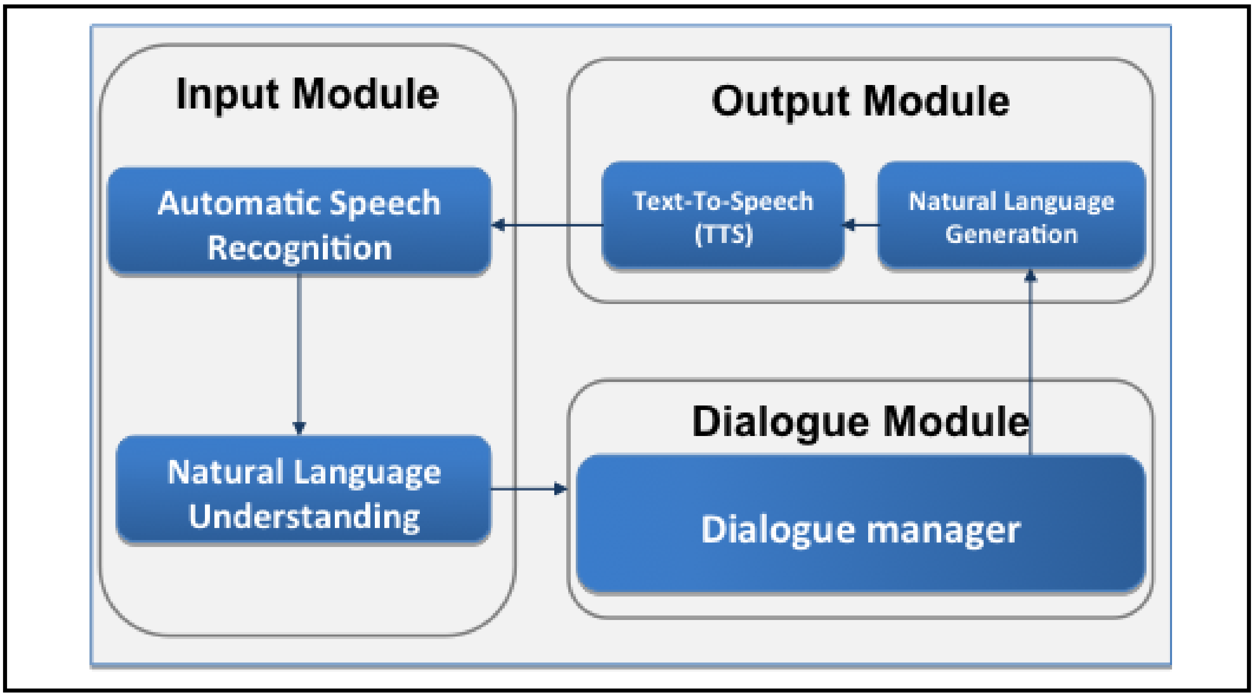


Figure 4.1: Proposed Model for a Speech-based ICCS (MIMIC)

Speech recognition for mobile devices is often not performed on the actual phone because of the limited processing power of the phone. Instead, most mobile speech providers, such as AT&T, Nuance and Google perform the processing of spoken inputs remotely (Gilbert & Feng, 2008, Chang, Hung, Wang *et al.*, 2011). The mobile phone is used as a microphone. The ASR, or speech-to-text module, detects speech from the microphone of the mobile phone and returns the corresponding text. This text or list of possible results is passed to the NLU that will determine the semantics. The NLU uses a database of homonyms and synonyms to reduce the possibility of errors.

The implementation of the ASR in the MIMIC-Prototype used the Android Speech Application programming interface (API). Android gives access to the remote speech engine to developers. This API sends compressed recorded sounds to Google speech servers. The speech servers process the sound and return the possible results ordered by their confidence scores.

4.2.2. Dialogue Module

The Dialogue Module (DM) contains only the Dialogue Manager, which is responsible for determining the next move of the dialogue and for generating responses to the user. Several design schemes are often used in designing dialogue systems. A frame-based approach was selected to implement the DM because it is less resource intensive than other dialogue approaches. The following commands were implemented: CALL (number), CALL (contact), SMS (number, content), SMS (contact, content), REDIAL, REPEAT and CANCEL.

Prior experiments with the speech recogniser resulted in poor accuracy when dictating a full text message. Pre-defined text messages were chosen based on text messages that are likely to be sent while driving (Ford, 2008). A few messages were therefore chosen in order to not require the driver to remember a long list of options. The following four options were selected:

- I will call when I get there,
- I cannot talk right now, I am driving,
- I am running a few minutes late,
- I am on my way.

The SUI was implemented on a Samsung Galaxy S2 running Android Gingerbread 2.3.3 (Google Inc, 2011). Android was chosen because it is a growing mobile platform that is supported by a wide range of mobile phone brands. The speech was converted into text using the speech recognition API shipped with Android (*android.speech*) (Google Inc, 2009). This API records and compresses spoken input; then sends it to the Google speech servers, which process it and return the results. In MIMIC-Prototype, whenever a spoken input is recognised by the ASR, a “beep” sound is played as feedback.

The recogniser listener library (*RecognitionListener*) was preferred to the traditional recogniser intent (*RecognizerIntent*) library because the latter needs to be activated manually.

This would force the driver to use his/her hands to operate the system. The library of the recogniser is only available for devices running Android 2.2 (Google Inc, 2010).

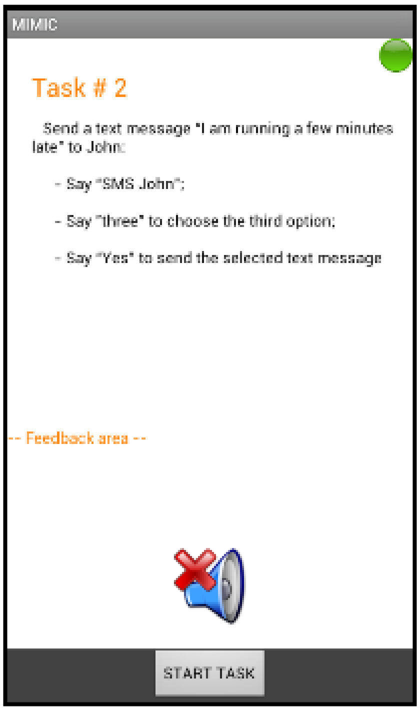


Figure 4.2: Screenshot of the MIMIC-Prototype

Figure 4.2 depicts a screenshot of the MIMIC-Prototype sending a text message. After the user says "SMS John", the system dictates a numbered list of available options and the user is asked to choose one option by saying the number corresponding to the chosen option. The user is then prompted to confirm or cancel the command by saying "YES" or "NO".

4.2.3. Output Module

Outputs are processed by two components: the Natural Language Generation (NLG) module and the Text-To-Speech (TTS) module. A lightweight NLG module was designed and implemented in order to arrange data in such a way that the user will have a better understanding. Short responses were chosen when designing the dialogue and the volume is adjusted. Examples of such output include telephone numbers and incoming text messages written using common abbreviations.

The TTS engine synthesises the information that needs to be sent to the user. The default Android TTS was used to implement this component.

4.3. User Study

The MIMIC-Prototype was implemented based on the proposed model (MIMIC). A user study was then conducted to investigate the feasibility of a SUI. The metrics used included usability, workload and user satisfaction. The impact of the MIMIC-prototype on driver distraction was not investigated. This was done in a subsequent study using a field study (Chapter 7). The aims of the user study and the research methodology are discussed in the following subsections.

4.3.1. Aim of the Study

The goal of the user study was to investigate the feasibility of using the MIMIC-Prototype. Participants used different strategies to make calls and send text messages. The research design used in the user study is discussed in the following sections. These include the selection of the participants, the apparatus, the procedure, the tasks performed and the metrics used.

4.3.2. Selection of Participants

Research has shown that ten users are sufficient to discover approximately 95% of usability issues (Nielsen & Landauer, 1993); therefore ten volunteers were recruited and completed the user study. The main selection criteria were the first language spoken (English, Afrikaans or any African languages) and the age of the participants. All the participants had a valid South African driving licence and a strong computing background. Most participants had never used a mobile ICCS prior to the study. This was done to investigate the ease of learning of the MIMIC-Prototype. The age of participants ranged from 18 to 29; this age group was chosen because the majority of drivers likely to use a mobile phone while driving are young (Lee, 2007, National Highway Traffic Safety Administration, 2012b, Zhao, Reimer, Mehler *et al.*, 2013).

4.3.3. Apparatus and Procedure

The user study was conducted in a laboratory environment. This was done to control the variables to be analysed, in order to be able to replicate the study and avoid the complexity of a field study. The environment was designed using the Lane Change Task (LCT) test software (Mattes & Hallen, 2009). As depicted in Figure 4.3, a 42” multi-touch screen was used to simulate the windshield of the car. In addition to that, a mobile phone was used by participants to perform communication tasks.

Each participant was required to sign an informed consent form prior to the start of the evaluation session. A brief demonstration was given by the test moderator on how to use the MIMIC-Prototype. During the demonstration, participants performed one of each type of task; that is, calling using a contact name, calling using a telephone number and sending a text message.

Figure 4.3: Laboratory Environment Settings

4.3.4. Tasks Performed

Each participant was asked to perform several tasks. These included making a call and sending a text message (Table 4.1). When making a call, participants had three options: using the name of a contact, dictating the phone number or redialling the last outgoing number.

Telephone numbers used followed the format valid in South Africa. South African telephone numbers consist of 10 digits starting with a “0”.

4.3.5. Metrics

Performance and self-reported metrics were captured. A log file was created on the mobile phone in order to save all performance data. Questionnaires (Appendix A) were used to collect the subjective data. These metrics included the following:

Performance Metrics:

- *Time-on-task (seconds):* the time spent in performing a task,
- *Error on task*: the number of errors made while performing a specific task,
- *Task completion*: whether the task was completed or not,
- *Success rate:* whether the task was successful or not.

ID	Task Description
T01	Please call the contact Maria: Say “Call Maria”; Say “Yes”;
T02	Send the text message “I am running a few minutes late” to John: Say “SMS John”; Say ”three” to choose the third option; Say “Yes” to send the selected text message;
T03	Redial the previous outgoing call: Say “Redial”; Say “Yes”;
T04	Please call Diana Say “Phone Diana”; Say “Yes”;
T05	Send the text message “I will call you when I get there” to Peter: Say “Text Peter”; Say ”one” to choose the first option; Say “Yes” to send the selected text message;
T06	Send the text message “I cannot talk right now, I am driving” to Janet: Say “SMS Janet”; Say “two” to choose the second option; Say “Yes” to confirm the message;
T07	Call a number : Say “Call 074 456 1245”; Say “Yes” to confirm the number.

Table 4.1: List of Tasks

Self-reported Metrics:

- *Overall satisfaction*: a System Usability Scale (SUS) questionnaire (Brooke, 1996) was used as an instrument to collect self-reported usability data. It provides more accurate results than other questionnaires for small sample sizes (Tullis & Albert, 2008),
- *Speech recognition and dialogue performance*: participants were asked to rate the accuracy of the speech recogniser as well as the flexibility of the dialogue,
- *Workload*: mental, physical and temporal demands were collected in addition to performance, effort and frustration experienced by participants. The NASA task load index (TLX) (Hart & Staveland, 1988) was used.

For each task, the user had to go back to the starting point and then press start on the phone placed next to the steering wheel. When the task had been performed, the user then pressed the stop button. In the log file, the system then recorded the task duration, the completeness and the success.

4.4. Results

The results are presented in three sections, namely performance, workload and satisfaction. Graphs and tables showing the results are discussed.

4.4.1. Performance

Figure 4.4 shows the number of errors logged by the system. The highest number of errors was recorded for Task 3 (redial the last outgoing number). Further analysis of the log showed that the command "REDIAL" was often not recognised. The use of a constrained grammar could reduce the occurrence of such an issue. Task 7 consisted of making a call using a telephone number. Most participants had to try at least twice in order to perform this task successfully.

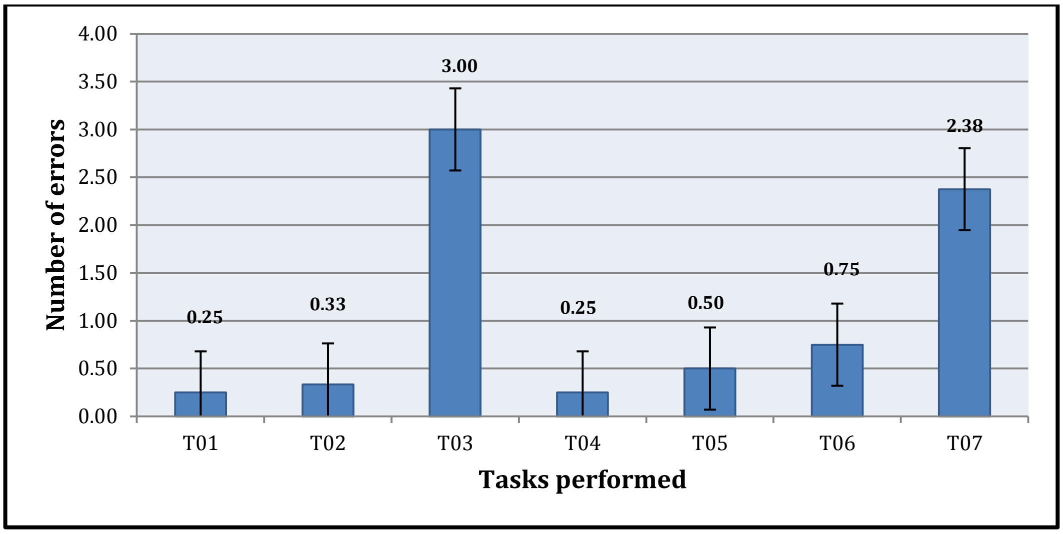


Figure 4.4: Means of Errors for Each Task (n=10)

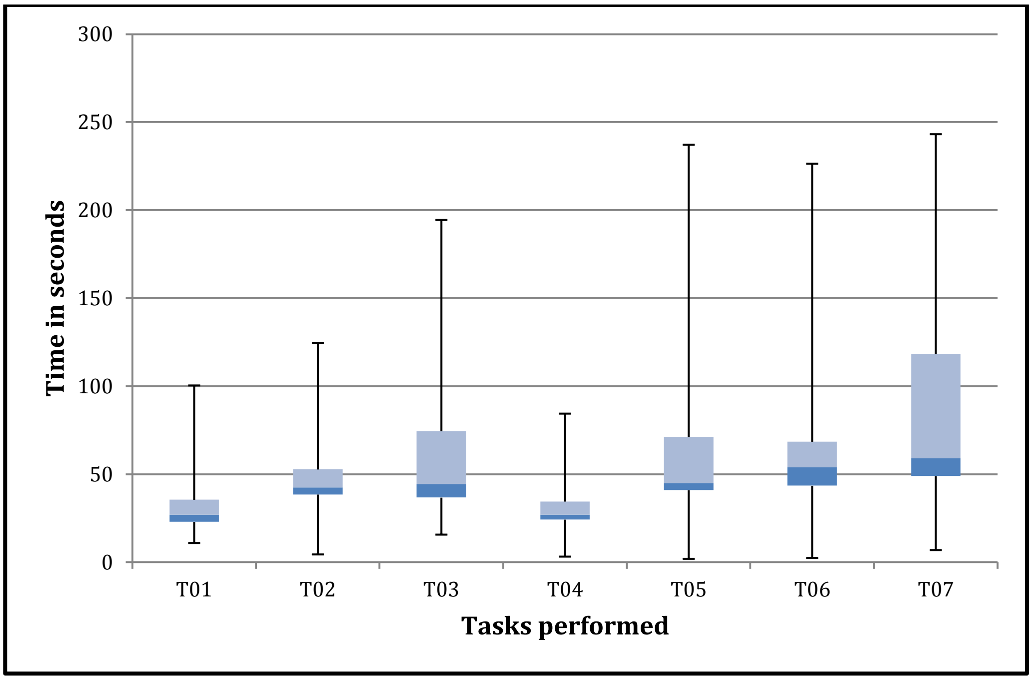


Figure 4.5: Time (seconds) Spent on Each Task (n=10).

Figure 4.5 depicts a box and whisker graph of the time-on-task. The minimum, the first quartile, the median, the third quartile and maximum are represented for each task. Calling a contact (T01 and T03) took less time than the other tasks; selecting and sending a text message to a contact (T02, T05 and T06) were performed, on average, within 45 seconds.

This is reasonable because after recognising the command, the system has to inform the user about the available options so that the user can choose the preferred option.

However, for these tasks many participants performed in the third quartile. Calling a contact using the telephone number (T07) was the most time-consuming task; most of the time was wasted trying to get the telephone number recognised correctly.

ID	Completion	Success
T01	10	9
T02	10	9
T03	6	6
T04	10	10
T05	10	9
T06	10	10
T07	7	6

Table 4.2: Task Completion and Success (n=10)

The completion and success rates of the tasks (Table 4.2) were encouraging for making a call using a contact name (T01 and T04). However, the use of the shortcut command “REDIAL” resulted in only sixty percent completion rate (T03). This was due to the poor recognition of the command. Sending text messages using a contact name (T02, T05 and T06) was all completed with a success rate of at least ninety percent. Thirty percent of participants failed to complete T07, which consisted of making a call by dictating a telephone number.

4.4.2. Workload

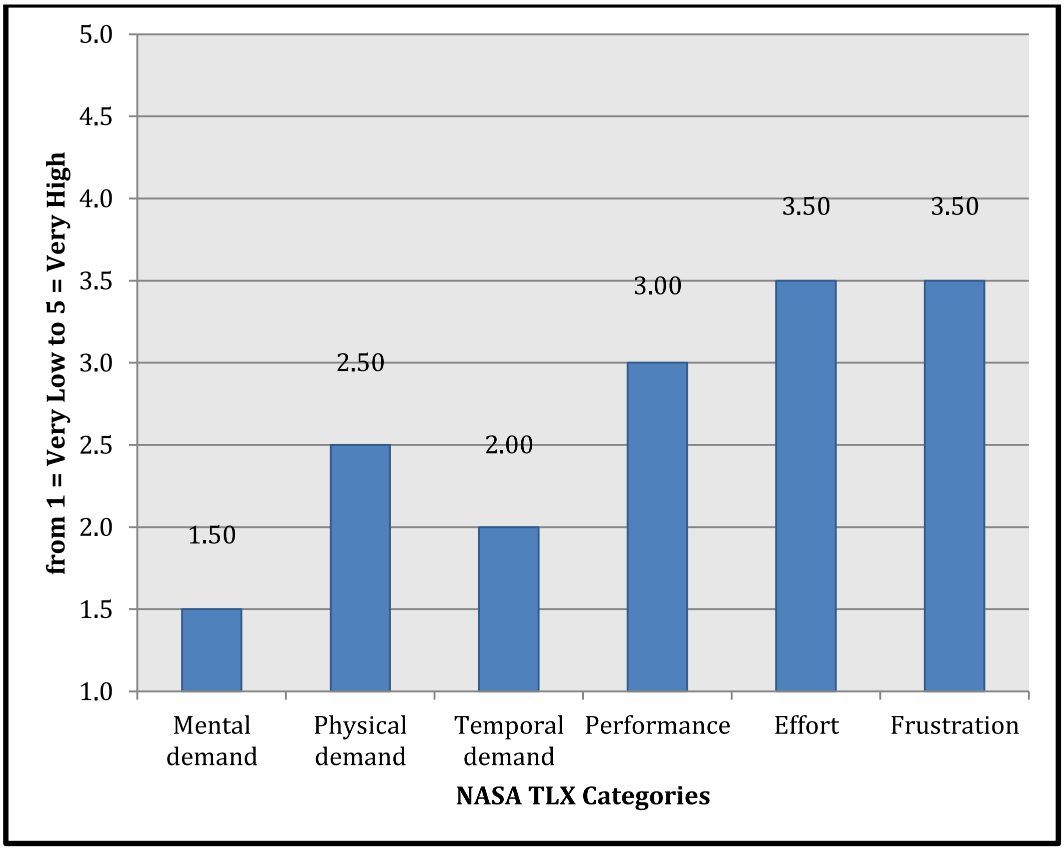


Figure 4.6: Means of the User Workload (n=10)

A post-test questionnaire was completed after completion of the tasks. The NASA TLX (Hart & Staveland, 1988) was used to obtain workload data (Figure 4.6). A 5-point semantic differential scale, ranging from 1 to 5 (1 = very low to 5 = very high) was used to rate each variable.

Mental demand was very low (mean = 1.50), which means that the tasks performed did not require a high mental demand from the participants. Physical demand was found to be high (mean = 2.50), because some participants had to bend towards the mobile phone when their commands were not recognised. Temporal demand was also low (mean = 2.00), which means that the time pressure due to the pace of each task was not rated negatively. The Performance was rated as acceptable (mean = 3.00); participants felt that they generally did well. Effort (mean = 3.50) needed to perform the tasks and the Frustration (mean = 3.50) were also given a high rating.

4.4.3. Satisfaction

The results of the user satisfaction questionnaire (Figure 4.7) were generally high. A 5-point semantic differential scale, ranging from 1 to 5 (1 = very low to 5 = very high) was used to rate each variable.

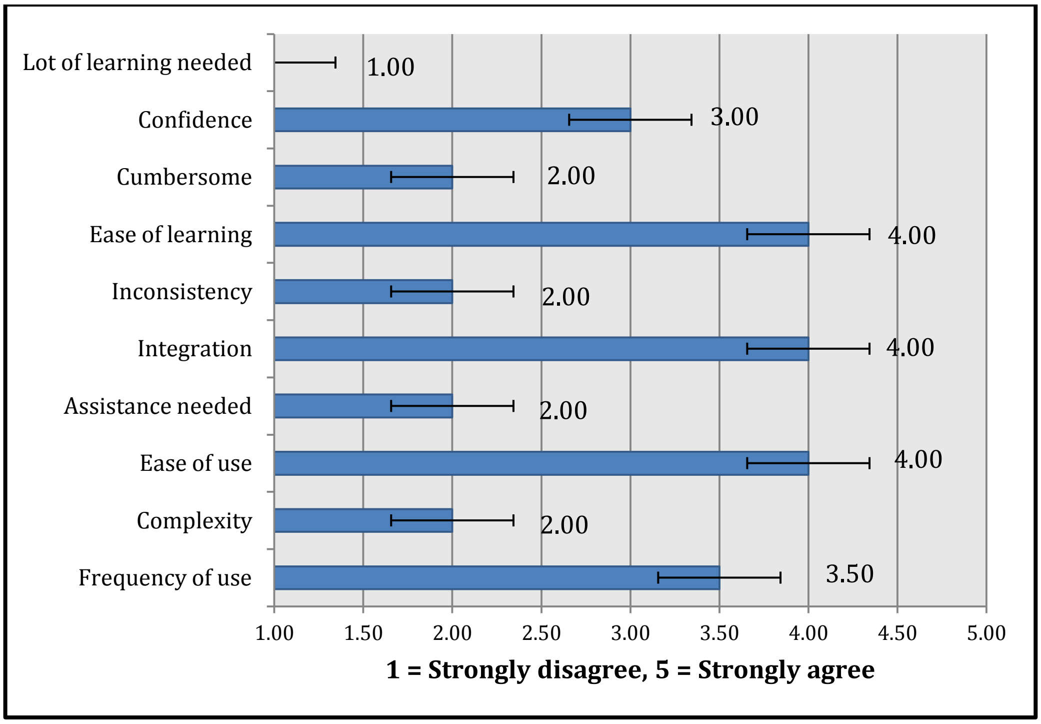


Figure 4.7: Means of User Satisfaction (n = 10)

The ease of use, integration and ease of learning were rated highly by the participants (median = 4.00). This was followed by the willingness of the participants to use the system frequently (median = 3.50) and the confidence that participants had in the system (median = 3.00). Few participants found the system complex, cumbersome or thought they would need assistance to use it. Very few participants (median = 1.00) felt that they would need a lot of learning (training) to be able to use the system efficiently.

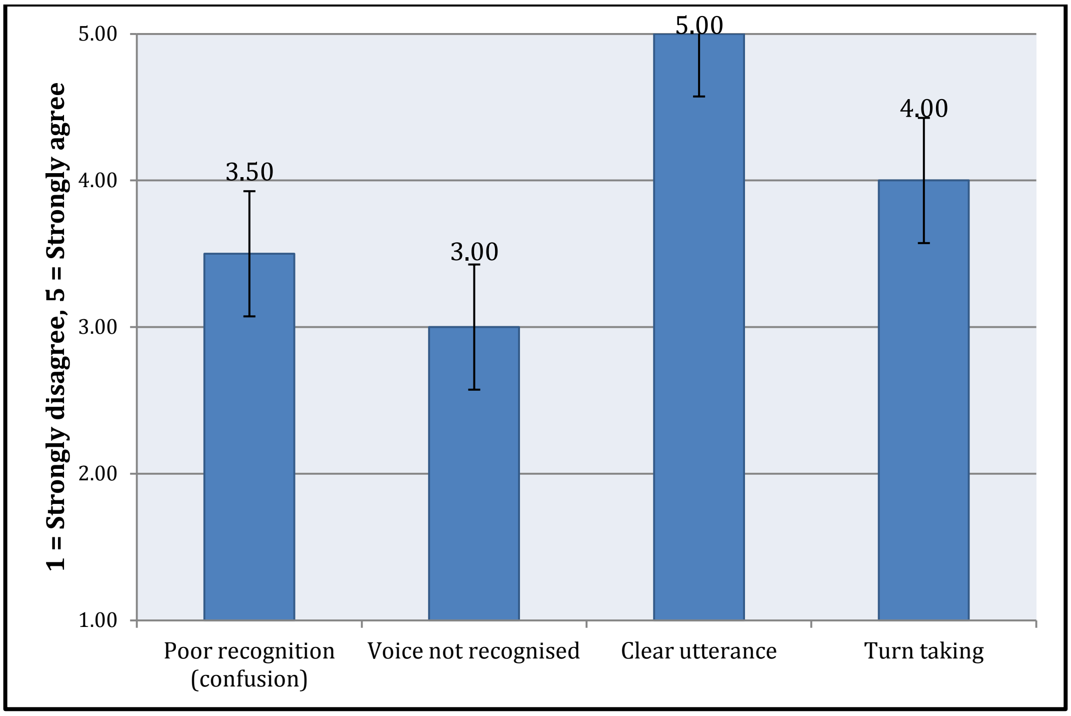


Figure 4.8: Means of Self-Reported Metrics on the SUI (n=10)

Figure 4.8 summarises the self-reported metrics regarding the SUI. Participants complained about being confused as a result of the poor recognition rate of the prototype. Some participants (mean = 3.00) felt that their voices were not recognised by the prototype. Clear utterances from the system's TTS (mean = 5.00) and a good turn-taking strategy (mean = 4.00) were rated highly.

4.5. Discussion

Overall, the participants gave positive feedback after interacting with the MIMIC-Prototype. However, some aspects of the prototype needed to be improved. The high number of errors in performing Task 3 and Task 7 contributed to frustrating the users. Despite the fact that the study was conducted in a controlled environment with little background noise, the speech recognition did not perform as well as expected. This raises the issue of the reliability of speech recognition for mobile devices. Most mobile devices use an on-cloud speech server, which does not support a grammar to narrow the search space. These issues can be resolved by using an offline speech recognition engine.

An analysis of the general comments given by participants highlighted the need for a *barge-in strategy*. A barge-in strategy enables the user to take control by interrupting the system. This will speed up the process of selecting a text message. When the proposed model was being implemented, some restrictions of the ASR did not allow the integration of a barge-in facility.

The implementation of the MIMIC-Prototype also served to verify the design of MIMIC. The model was successfully used to design and evaluate the prototype and did not require any changes, other than those discussed above relating to the ASR Module (Figure 4.1).

4.6. Conclusion

This chapter discussed the design of a speech-based model for mobile ICCS, called MIMIC and the implementation of the MIMIC-Prototype using an SUI. The system was found to be highly effective in sending pre-recorded text messages and making calls. The majority of the participants indicated that they were willing to use such an ICCS in future. However, the dictation of telephone numbers was a source of errors that frustrated the participants. In addition, the system frequently failed to recognise the “REDIAL” command. This implies that this command should be left out in a future implementation of the speech-based ICCS, unless the speech recognition engine used is of a better quality than the one used in this usability evaluation.

A full research paper, based on the proposed model and the user study, was submitted and accepted for the 2012 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists (SAICSIT 2012) (Tchankue, Wesson & Vogts, 2012).

The next chapter will address the determination of the driving context. This will include the implementation of the Context-Aware Module and experiments conducted to select the most effective techniques to acquire, infer and use contextual information.

Chapter 5: Inferring the Driving Context

5.1. Introduction

The aim of this chapter is to investigate how the driving context can be inferred using non-intrusive techniques. This will be done by answering two research questions. The first question, *RQ5: How can an inference engine be designed for a mobile in-car communication system (ICCS)* will be answered by proposing a design based on mobile phone data. The second question, *RQ6: What are the most efficient techniques to determine the driving context* will be answered by performing a set of experiments.

The driving context is defined in terms of driving events and distraction level. As described in Section 3.4, the source of context data will be sensors and other web services available from mobile phones. These include sensors such as the accelerometer, gyroscope and compass as well as global positioning systems (GPS) and weather information.

This chapter is structured as follows: Section 5.2 defines the driving context and how it will be used in this project and Section 5.3 provides an overview of machine learning as a tool to learn from data. Section 5.4 describes the design and the implementation of the data collection. Section 5.5 describes the experiments conducted to investigate possible relationships among mobile phone sensor data. Section 5.6 discusses the training process and the results of the experiment used to select the most effective algorithm to determine the driving events. Section 5.7 discusses the results of the experiment that was used to select the most effective algorithm to determine the perceived distraction level. Section 5.8 contains a discussion on the results and Section 5.9 concludes this chapter.

5.2. Driving Context

Driving context in this research is defined as a combination of driving events and the distraction level (Figure 5.1). Driving events describe situations in which the car can be, such as driving on a straight road, turning, going over a speed bump or changing lanes. These variables affect the tactical level of the driving model (Section 2.3.2). Other research has also identified similar driving events as crucial (Richard, Campbell & Brown, 2006).

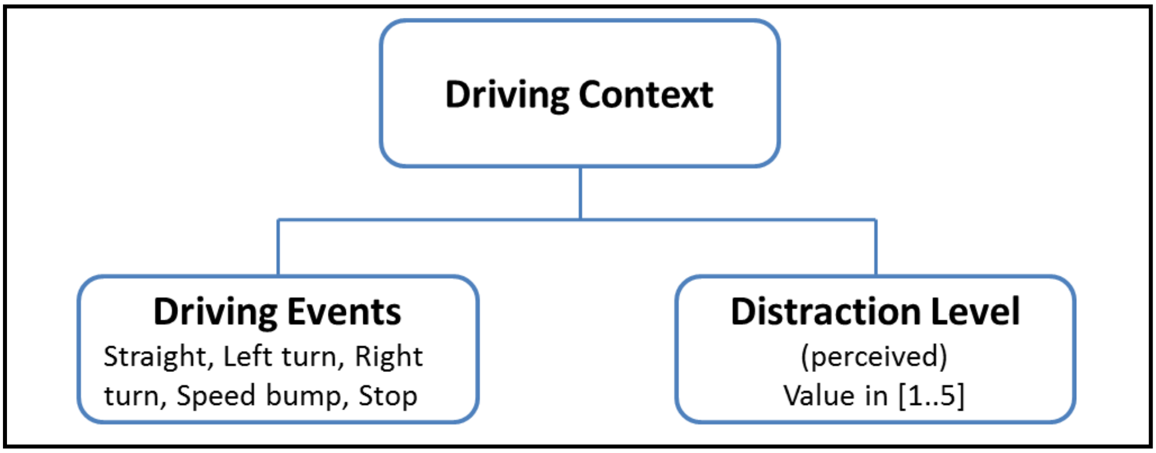


Figure 5.1: The Definition of the Driving Context for this Research

The distraction level is a measure of the perceived distraction experienced by the driver. It ranges from 1 (very low) to 5 (very high).

Many car accidents occur when the driver reaches an intersection or overtakes another car (Road Traffic Management Corporation, 2008). Driving at high speed contributes significantly to car accidents (Road Traffic Management Corporation, 2008). At high speeds, the braking distance increases because the car takes more time to slow down. It is therefore more difficult to react to an external event.

Several studies have found a strong correlation between bad weather conditions and driving errors (Staubach, 2009). The following reasons are often mentioned as a cause of these driving errors: a decrease in visibility, increased difficulty in controlling the car and an increase in the braking distance. When it is raining, roads are wet and become slippery. This results in an increase in the risk of car accidents, injuries and damage to the car. The models to be generated in the following sections will not make use of weather information. The reason for this is that the experiments were not going to be conducted over a period of time long enough (e.g. a calendar year) to provide a significant variation of weather conditions.

Several machine learning techniques have been used either to detect driver distraction (Liang, 2009) or to monitor driver activities (Veeraraghavan, Bird & Atev, 2007). The performance of these techniques often depends on the type of input vector used and sometimes, on the pre-processing operations done before training.

5.3. Machine Learning

Section 3.6 reviewed some commonly-used machine learning techniques. There are three types of machine learning algorithms: supervised, unsupervised and reinforcement learning (Angelbrecht, 2007). Supervised learning techniques take as input examples, which provide desired outputs for given inputs (training set). Unsupervised learning techniques do not require examples of classified data. The model is generated from unclassified data using some mathematical models that the data are likely to follow. Reinforcement learning techniques learn by using an indication of correctness at the end of some reasoning.

Reinforcement learning is often used in a domain where the computer needs to be guided by a human expert, for example, playing chess or piloting a helicopter (Ng, Coates, Diel *et al.*, 2006). This class of algorithm would have been appropriate for a self-driving car. On the other hand, unsupervised learning can be unpredictable in terms of the number of categories that is generated at the end of the process. It is therefore easier to use a *supervised learning algorithm*. The observer will provide the correct target to the system and the training will be based on that.

5.4. Design and Implementation

The model proposed in Figure 4.1 (MIMIC), was updated in order to implement data capturing that will be used for training. Supervised learning was used to predict the driving context. Variables (sensors, GPS coordinates, and web service responses) that are going to be captured using the mobile phone were added to the Input Module. A Context-Aware Module was also included containing the Inference Engine that will use a machine learning algorithm to infer the current driving context.

5.4.1 Design

Figure 5.2 is an updated version of Figure 4.1. The updated model takes into account the fact that the training data needs to be captured from a mobile phone. The Dialogue Module was updated with the Context-Aware Module that now includes the Inference Engine, which is responsible for determining the current driving context.

Figure 5.2 depicts the updated model for a mobile, context-aware ICCS, called the Multimodal Interface for Mobile Info-communication with Context (MIMIC). This model comprises three modules including the Input Module, the Dialogue Module and the Output Module.

The Input Module comprises the Automatic Speech Recogniser (1), the Natural Language Understanding module (2), various mobile phone sensors (3), the data pre-processing and fusion sub-module (4) and a Peer's phone (5). The Dialogue Module is critical for the context-awareness functionality of the architecture as it contains the Context-Aware Module (6) and the Dialogue Manager (7). The Context-Aware Module contains the Inference Engine. The Output Module comprises the Natural Language Generation (8) and the Text-To-Speech (9) modules.

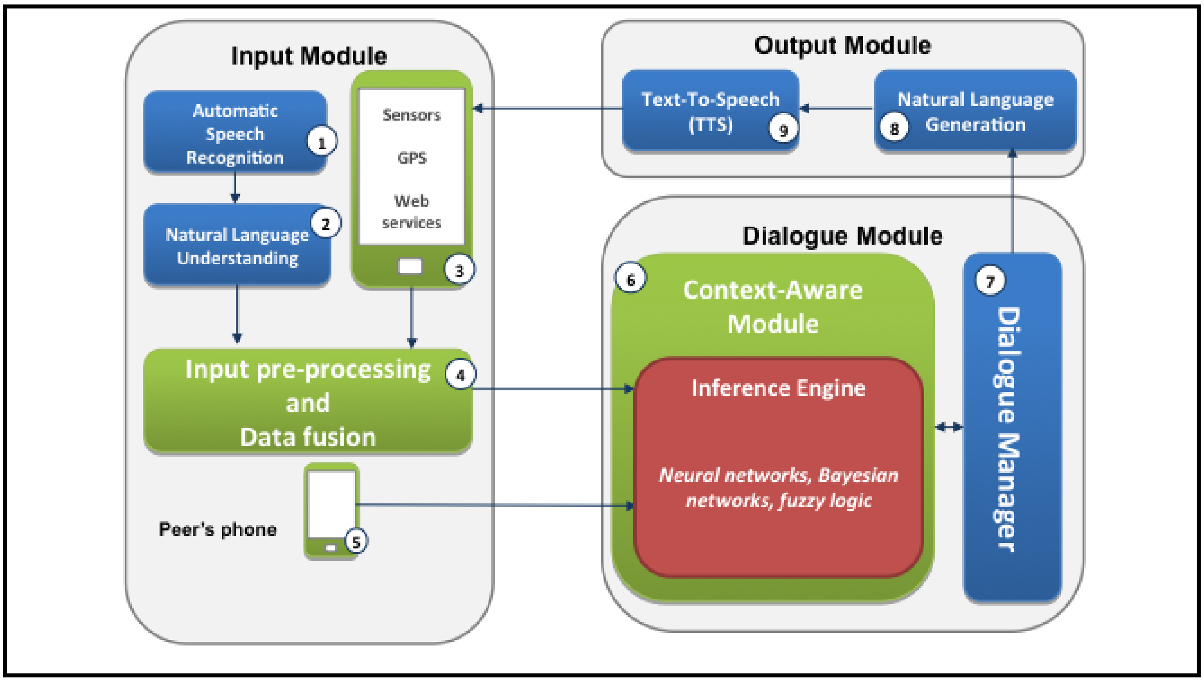


Figure 5.2: The Updated MIMIC Model

Component (4) gathers and combines pre-processed input coming from the mobile phone and the Speech Recogniser. The results obtained from the Speech Recogniser are used as an input to the Dialogue Manager, while information from the phone provides input into the Inference Engine in order to determine the context.

Several experiments were conducted in order to infer the driving context. The following section discusses the methodology used to conduct the experiments as well as the implementation of the mobile application used for this purpose.

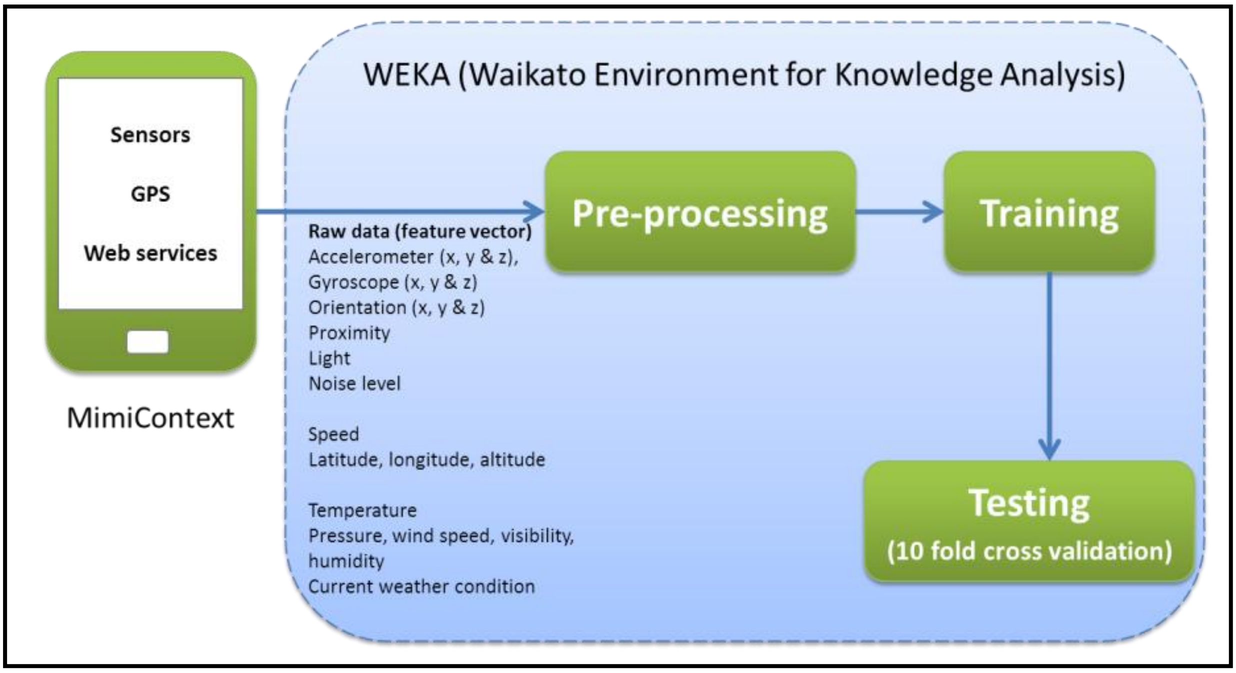


Figure 5.3: Data Capturing and Training

Figure 5.3 illustrates the process that was followed to capture the data. This is a classical supervised learning process where raw data are pre-processed for the training (Theodoridis & Koutroumbas, 2009). Raw sensor information coming from the mobile phone are collected and saved using the object-oriented model discussed in Section 3.3.4. Pre-processing using a high-pass filter was performed for data obtained from the accelerometer and the gyroscope. Methods used for training and testing phases vary depending on research projects. The experiments discussed in Section 5.6 and Section 5.7 will detail the training and the testing phase.

5.4.2 Implementation

The following section discusses the implementation of the MIMIC-prototype that collects data from the phone. The training and testing of the model are also discussed.

In order to discover the context, a mobile application was implemented and deployed on a Samsung Galaxy S3. The application was based on the updated MIMIC model shown in Figure 5.2. An attempt to use an offline speech recognition engine, as suggested in Section 4.5, was not successful. The default language and acoustic models were not effective enough to be used in the field study. Therefore the native Android's remote speech recognition engine was used.

Only those modules related to context-awareness were implemented; these included all mobile sensors and the web services gatherer (3), the pre-processing and data pre-processing (4), and the Inference Engine (6).

The input or feature vectors consisted of sensor and GPS data that were saved in comma-separated value (CSV) files. These files were later converted into ARFF files that are required by the WEKA framework used to train the Inference Engine (Hall *et al.*, 2009).

5.4.3 Training

The WEKA framework was chosen for the training owing to the fact that most learning algorithms are complex to implement. WEKA is an open source machine learning framework developed at the University of Waikato in New Zealand (Hall *et al.*, 2009). It has the advantage of having been validated and developed in Java and therefore runs on several computer platforms including mobile devices. It implements several well-known machine learning algorithms in the following categories: decision trees, neural networks, Naïve Bayes, support vector machine (SVM) and nearest k neighbours (Section 3.6).

With the release of the WEKA library for Android, the actual training was conducted on a mobile phone. After the training, a classifier was generated and saved on the system. The most accurate classifier was used to evaluate each input vector.

When using WEKA, the logistic function, neural networks and logistic model tree (LMT) algorithms took at least three minutes to generate the classifier while other algorithms took less than twenty seconds. This will not have a negative impact on the implementation of the Inference Engine of the MIMIC-Prototype because the training is done offline.

5.5. Relationship between Mobile Sensor Data

Sensor information has been used extensively in ubiquitous computing in various domains. What is true in a specific domain cannot be automatically applied to another domain without prior investigation. Several experiments were conducted on the sensor data. The aim of these experiments was to have a better understanding of the data obtained from the mobile phone because data might be subject to noise.

Depending on the type of mobile phone that is being used, some sensors are more likely to drain the battery than others. In addition, accuracy also depends on the model of mobile phone that is used to conduct the experiment.

Some mobile sensors tend to measure correlated variables. Sensors that provide similar information could be interchanged in the case where one of these sensors is unavailable or provides inaccurate information.

Data was recorded and an analysis was conducted for the same period of time (100 seconds). The recording was done for 15 minutes but the section analysed was only 100 seconds long. Figure 5.4 depicts two graphs comparing the GPS direction and the compass (the orientation x axis). Despite some noise, it can be observed that the two graphs follow the same pattern, i.e. whenever the GPS direction increases, the compass reading increases as well and when the GPS direction decreases the compass reading also decreases.

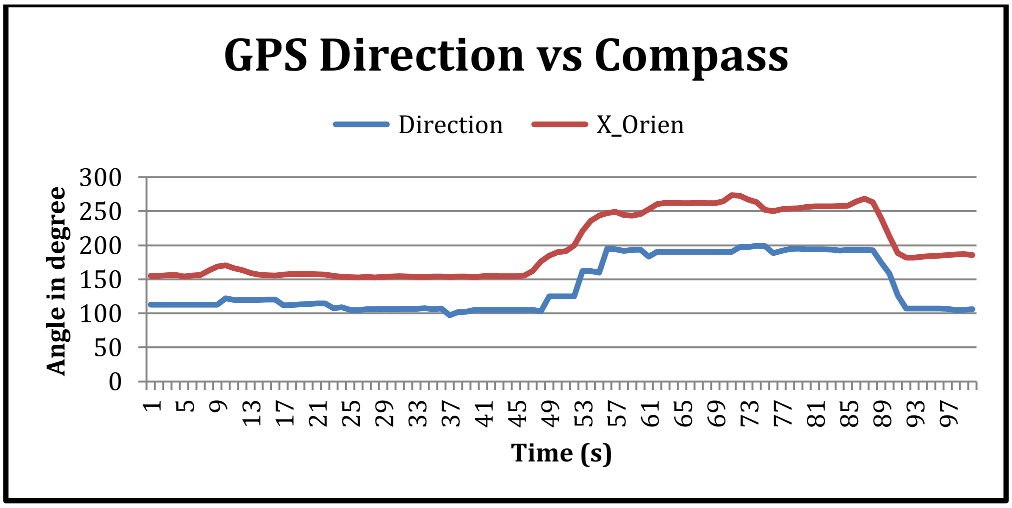


Figure 5.4: Compass and GPS Direction

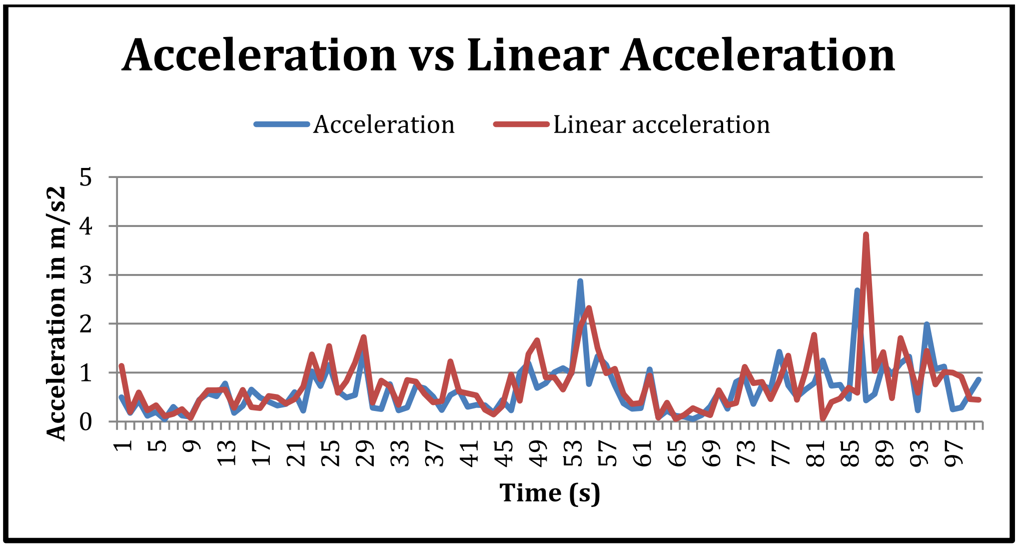


Figure 5.5: Acceleration and Linear Acceleration

Figure 5.5 shows that the acceleration (the gravity was removed) and the linear acceleration follow a similar pattern despite the differences in amplitude.

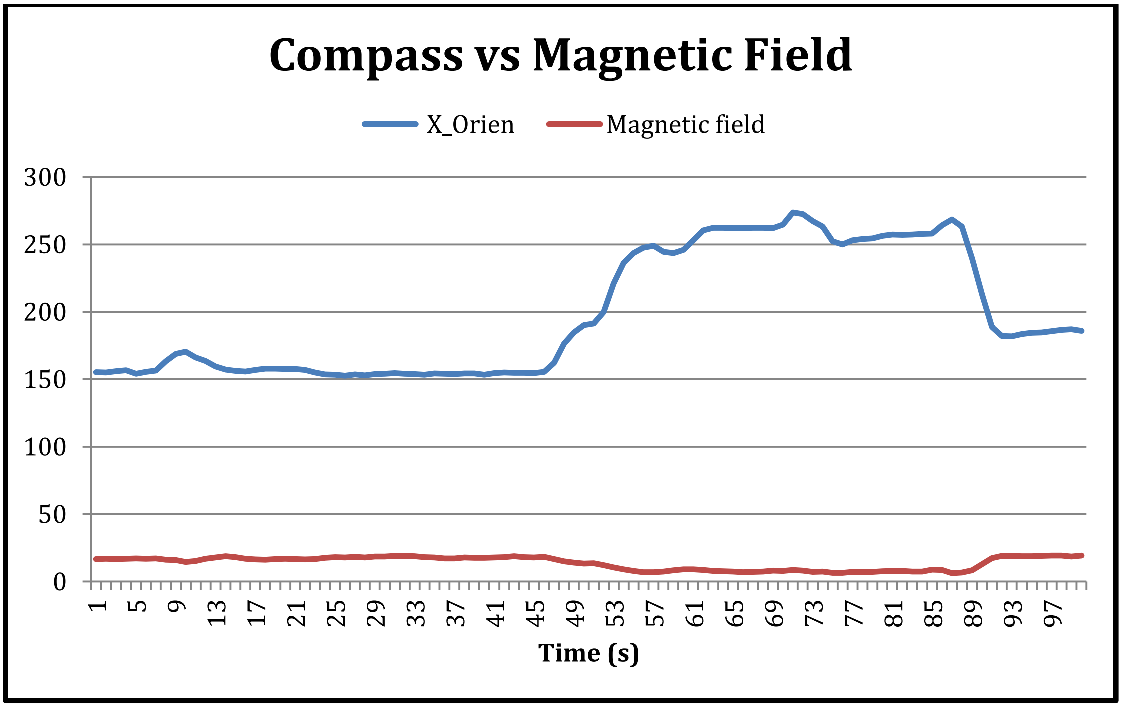


Figure 5.6: Compass and Magnetic Field

Figure 5.6 shows an inverse relationship between the magnetic field and the orientation (compass). Whenever the magnetic field increases, the compass value decreases.

The lessons that can be learned from this experiment are the following:

- The GPS direction and compass are directly proportional; with little processing, each of these variables can be substituted for the other,

- The acceleration and the linear acceleration are proportional. A direct substitution can be made using the following calculation:

$$linear\ acceleration = acceleration - g,$$

- The magnetic field and compass are inversely proportional; the inverse of each of these variables can replace the other variable in case of the unavailability or inaccuracy of one sensor.

The collection of the data to predict the driving context occurred in two different experiments. Section 5.6 discusses how the driving event model was created and Section 5.7 discusses the creation of the distraction level model.

5.6. Predicting the Driving Events

An experiment was conducted to automatically determine the current driving event of a car. This was achieved with the help of the Inference Engine that used data collected by the mobile phone. Driving events that were analysed include:

- Stop,
- Right turn,
- Left turn,
- Straight,
- Changing lane,
- Speed bumps.

The research design used in the experiment is discussed in the following sub-sections. This includes the selection of participants, the apparatus and the description of the experiment.

5.6.1 Selection of Participants

Five volunteers were recruited to complete the data capturing. The main selection criterion was driving experience. All of the participants possessed a valid South African driving licence. The age of participants ranged from 18 to 40. Three males and two females participated in the experiment.

5.6.2 Apparatus

The driver and the primary researcher were the only occupants of the car. The primary researcher used the data capturing application while the driver drove the car. This was done for safety purposes as it might have been unsafe for the driver to capture data while driving.

5.6.3 Description of the Experiment

Participants drove on an urban road with two single lanes. The distance covered was approximately 6.5 kilometres with two right turns and two left turns of about 90 degrees. The route had ten speed bumps. Changes of lane were mostly recorded before making a turn.

Figure 5.7: Screenshot of the capture of the driving events

Figure 5.7 shows the screenshot of the application that was used for the data capturing. This screen was used to record road situations (speed bump, right turn, left turn, change lane, stop and move straight). The current speed is displayed at the top of the screen and all data recorded are displayed at the bottom. For safety reasons, the data capturing was performed by the primary researcher in the car. Events were captured as soon as they occurred.

5.6.4 Results

Classifier	% Correctly classified
IB1	**89.85%**
J48	**89.38%**
IBk	**89.25%**
J48graft	**88.91%**
Multilayer Perceptron	**88.51%**
Bayesian Network	**86.57%**
LMT	**86.31%**
KStar	84.84%
AdaBoostM1	82.70%
SMO	80.90%
RBFNetwork	75.02%
NaïveBayesUpdateable	54.24%
Naïve Bayes	54.24%

Table 5.1: Classification of Driving Events

The data collected was analysed using several machine learning techniques. Some tables appearing in this section had their numbers rounded to the next integer for the sake of simplicity. Table 5.1 shows the performance determined after the application of the different machine learning techniques.

The 10-fold cross-validation was used during the test phase; data were split into ten equal sub-sets. The prediction model obtained was trained using nine sub-sets and tested on the tenth sub-set. The process was repeated until each of the sub-sets was used as a test set. The error rate was the average of the resulting errors.

IB1, J48, Multilayer perceptron, Bayesian network and LMT classified more than eighty-five per cent (bolded) of the test data correctly. IB1, IBk and J48 gave even better accuracy (89%). The Naïve Bayes classification and one of its variants (NaïveBayesUpdateable) performed very poorly (54.24%). This can be explained by the fact that the Naïve Bayes algorithm assumes that all variables are independent, but some variables such as the orientation, compass and the GPS direction, are clearly related.

Confusion matrices are often used in machine learning to understand how the algorithm actually performs on each class of data (Kampichler, Wieland, Calmé *et al.*, 2010, Yao, Gall

& Van Gool, 2010, Doubleday, McLaren, Chien *et al.*, 2011). The confusion matrix is a square matrix, which displays a class on each row and column. The intersection between a row and a column represents the number of instances that were classified. The best case scenario is to have a matrix with 0s everywhere except in the diagonal. Tables 5.2, 5.3 and 5.4 represent confusion matrices for the three best classifiers (IB1, J48 and IBk).

	a	b	C	d	e	f	Classification
a	**90**	3	2	2	2	1	a = Stop
b	1	**86**	2	1	4	6	b = Straight
c	4	15	**65**	12	4	0	c = Right turn
d	2	11	2	**79**	7	0	d = Left turn
e	0	23	0	0	**77**	0	e = Change lane
f	3	16	0	0	0	**81**	f = Speed bump

Table 5.2: Confusion Matrix for the *IB1* Algorithm (%)

As shown in Table 5.2, the “Stop” event was successfully classified as a “Stop” (90%). Eighty-six per cent of “Straight” event was correctly classified and approximately 6% was misclassified as “Speed bump”. “Right turn” was often misclassified as “Straight” (15%) and “Left turn” (11%). Seventy-nine percent of “Left turn” was correctly classified while 10% was misclassified as “Straight”. Seventy-seven percent of “Change lane” was correctly classified and 23% was misclassified as “Straight”. Eighty-one per cent of “Speed bump” was correctly classified and 16% was misclassified as “Straight”.

In Table 5.3 the “Stop” was successfully classified as “Stop” (94%) and only 5% of “Stop” was misclassified as “Straight”. The “Straight” followed the same pattern with 93% of data correctly classified and 3% misclassified as “Stop”. “Right turn” was often misclassified as “Stop” (27%) and “Straight” (46%). Sixty-one per cent of “Left turn” was correctly classified, while 16% was misclassified as “Stop” and 21% as “Straight”. Seventy per cent of “Change lane” was correctly classified and 27% was misclassified as “Straight”. Only 50% of “Speed bump” was correctly classified, while almost half (47%) was misclassified as “Straight”.

	a	b	c	d	e	f	Classification
a	**94**	5	1	0	0	0	a = Stop
b	3	**93**	1	1	1	1	b = Straight
c	27	46	**23**	4	0	0	c = Right turn
d	16	21	2	**61**	0	0	d = Left turn
e	0	27	0	0	**70**	3	e = Change lane
f	3	47	0	0	0	**50**	f = Speed bump

Table 5.3: Confusion Matrix for the *J48* Algorithm (%)

	a	b	c	d	e	f	Classification
a	**93**	6	1	0	0	0	a = Stop
b	4	**95**	0	0	1	0	b = Straight
c	38	23	**31**	8	0	0	c = Right turn
d	7	30	0	**61**	0	2	d = Left turn
e	0	67	0	0	**33**	0	e = Change lane
f	9	53	0	9	0	**28**	f = Speed bump

Table 5.4: Confusion Matrix for the *IB3* Algorithm (%)

With the IB3 algorithm (Table 5.4), the situation was very similar to the J48 algorithm (Table 5.3). Only 31% of “Right turn” was correctly classified, while 38% was misclassified as “Stop” and 23% as “Straight”. Only 28% of “Speed bump” was correctly classified, while the IB3 misclassified more than half (53%) as “Straight” and 9% as “Stop” and “Right turn”.

The training set was modified by removing all those variables that were in the minority in the nodes of the J48 algorithm. These variables included noise level, light, proximity, altitude, longitude and latitude. Table 5.5 summarises the results obtained: IB1 and IBk performed better than the other algorithms and the accuracy increased from 89% to 92%.

Classifier	% Correctly classified
IB1	**92.25 %**
IBk (3)	**92.05 %**
Multilayer perceptron	88.77 %
J48	87.50 %
Bayesian Network	85.57 %
Logistic	80.96 %
Naïve Bayes	56.64 %

Table 5.5: Performance of Classifiers Using the Reduced Training Set

5.7. Predicting the Distraction Level

The experiment used to accurately predict the distraction level is discussed in the following sub-sections. This includes the selection of participants, the apparatus and the procedure.

Distraction level	Description
1	Very Low
2	Low
3	Medium
4	High
5	Very High

Table 5.6: List of Distraction Levels

Table 5.6 shows the distraction levels that were considered for this experiment. The distraction level ranged from 1 to 5; 1 being the lowest level of distraction ("Very Low"), 2 corresponded to "Low", 3 corresponded to "Medium", 4 corresponded to "High" and 5 corresponded to "Very High".

5.7.1 Research Design

The research design used in this second experiment was similar to the one used to predict driving events. The experiment is discussed in the following sub-sections. These include the selection of participants, the apparatus and the description of the experiment.

5.7.1.1 Selection of Participants

Five volunteers were recruited to complete the data capturing. All participants had a valid South African driving licence and driving experience of at least two years. The age of participants ranged from 18 to 40.

5.7.1.2 The Apparatus

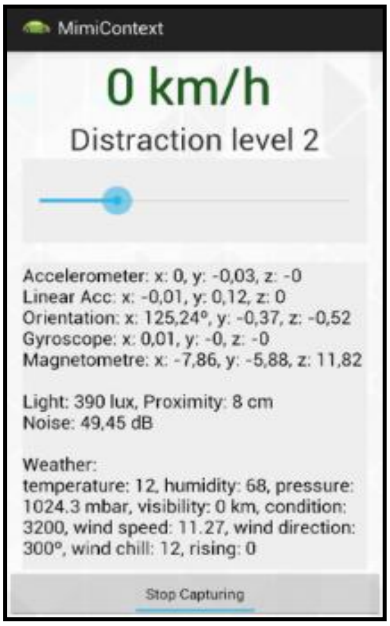


Figure 5.8: Screenshot of the capture of the perceived distraction level

The data collection took place in two phases; the first phase was similar to the previous experiment. The driver and the primary researcher were the only occupants of the car. The primary researcher changed the distraction level by using the slider to record the perceived distraction at a given time. The second scenario involved a vehicle with four passengers including the driver.

Figure 5.8 shows the screenshot of the application that was used for the data capturing. This screen was used to record the perceived distraction level (from 1 = very low to 5 = very high). The current speed is displayed at the top of the screen and all data recorded are displayed at the bottom of the screen.

5.7.1.3 Description of the Experiment

For the first phase, participants drove on an urban road with two single lanes. The distance covered was about 6.5 kilometres with two right turns and two left turns of about 90 degrees each. The route had several speed bumps. For the sake of having a wider range of speed and type of road, the second phase of the experiment was conducted on a highway. Data was captured while the vehicle travelled a distance of 131 kilometres. The results are discussed in the next section.

5.7.2 Results

The WEKA framework was also used to train, test and analyse the data. A 10-fold cross-validation was used during the test phase.

Classifier	**% Correctly classified**
KStar	**95.16 %**
IB1	**94.96 %**
RandomForest	**94.49 %**
IB2	**94.21 %**
IB5	**93.37 %**
J48	**90.96 %**
J48graft	**90.94 %**
IB10	**90.36 %**
MultilayerPerceptron	82.07 %
BayesNet	70.47 %
SMO	65.84 %
NaïveBayes	51.86 %

Table 5.7: Classification of Distraction Level

Table 5.7 shows the performance in descending order of each algorithm that was used. A total of eight algorithms scored more than 90% precision. The KStar, which is a nearest neighbour algorithm, had the best performance (95.16%). Another nearest neighbour algorithm, IB1, followed with a precision of 94.96%. The third best performing technique was a decision tree algorithm known as RandomForest (94.49%). Conversely, the

NaïveBayes and the Sequential Minimal Optimisation (SMO) had the worst performance, of 51.86% and 65.84% respectively.

	a	b	C	d	e	Classification
a	**97**	1	2	0	0	a = Low
b	5	**94**	1	0	0	b = Very Low
c	1	0	**95**	4	1	c = Medium
d	0	0	2	**96**	2	d = High
e	0	0	2	7	**92**	e = Very High

Table 5.8: Confusion Matrix for the *KStar* Algorithm

Table 5.8 shows more details about the performance of the KStar algorithm. The highest confusion was between "Very High" and "High", where only 92% of "Very High" distraction levels were correctly predicted. Seven per cent were classified as "High". Other classes had fewer errors; 97% of "Low" distraction was correctly predicted.

As shown in Table 5.9, the IB1 algorithm also gave interesting results. The classes "Low", "Very Low", "Medium" and "High" were correctly classified by 95%. Ninety-one per cent of the class "Very High" was correctly classified while 7% were misclassified as "High".

	a	b	C	d	e	Classification
a	**97**	1	2	1	0	a = Low
b	4	**95**	1	0	0	b = Very Low
c	1	0	**95**	4	0	c = Medium
d	0	0	2	**96**	2	d = High
e	0	0	2	7	**91**	e = Very High

Table 5.9: Confusion Matrix for the *IB1* Algorithm

The RandomForest algorithm seemed to work well with the lowest distraction levels. In Table 5.10, up to 98% of "Low" was correctly classified, while 94% was correctly classified as "Very Low". As we observed in the previous tables, a portion of "Very High" was misclassified as "High", while only 87% of the tested data was correctly classified as "Very High".

	a	b	c	d	e	Classification
a	**98**	0	1	0	0	a = Low
b	6	**94**	0	0	0	b = Very Low
c	2	0	**94**	3	0	c = Medium
d	0	0	3	**95**	1	d = High
e	0	0	1	12	**87**	e = Very High

Table 5.10: Confusion Matrix for the *RandomForest* Algorithm

5.8. Discussion

According to the results, a large number of "Speed bump" events were misclassified by the top performing classifiers. Classifiers typically misclassified "Speed bump" as "Stop" or "Straight". This can be explained by the fact that, when driving over a speed bump, the car carries on moving in a straight line, which confused the classifier and the event was misclassified as "Straight"; and secondly, the car slows down and the classifier assumes that the car has stopped ("Stop"). However, the IB1 classifier was 81% accurate in determining "Speed bump".

Although the top three classifiers scored at least 89% accuracy, the IB1 classifier seemed to be most effective as its worst misclassification ("Right turn") did not exceed 35%.

WEKA provides an option to generate a tree. Figure 5.9 contains an extract of the tree generated by the J48 classifier. The speed variable is the root node; this means that speed had the highest information density or was the most important variable.

The entire tree generated by the J48 classifier had a size of 157 nodes and 72 leaves. The speed variable was used on 47 nodes (30%). This shows how important speed is in determining the driving context. Acceleration and linear acceleration were used on 30 nodes. After an analysis of the tree, acceleration nodes are often parents of the leaf node "Speed bump". The x component of the orientation (heading) was only used in two nodes. This shows that speed, acceleration and linear acceleration are important in determining the current driving situation.

```
J48 pruned tree
------------------

speed <= 68.545404
|   orientation x <= 136.994629
|   |   speed <= 11.751788
|   |   |   magnetic x <= -14.16
|   |   |   |   Acc(magnetude) <= 0.303517: Very low (7.0)
|   |   |   |   Acc(magnetude) > 0.303517: Low (6.0/1.0)
|   |   |   magnetic x > -14.16: High (32.0)
|   |   speed > 11.751788
|   |   |   magnetic y <= 12.3
|   |   |   |   magnetic z <= 12.24: High (35.0/1.0)
|   |   |   |   magnetic z > 12.24
|   |   |   |   |   speed <= 35.360963
|   |   |   |   |   |   orientation y <= -52.06414
|   |   |   |   |   |   |   Acc x <= 0.056556: High (3.0)
|   |   |   |   |   |   |   Acc x > 0.056556: Very high (3.0/1.0)
|   |   |   |   |   |   orientation y > -52.06414: Very high (50.0/1.0)
```

Figure 5.9: An Extract of the Tree Generated by the *J48* Classifier

In the two experiments that were discussed, the most accurate algorithms were either decision trees or neighbour decisions (Table 5.1, Table 5.5 and Table 5.7). Although Bayesian networks and SVM perform better with other sensor data, these algorithms did not produce satisfactory results with mobile sensor data. This confirms other research stating that decision trees and Bayesian networks seem to work in opposite ways (Phyu, 2009).

The analysis of confusion matrices (Section 5.6) revealed how the classifier misclassified either the prediction of the driving events or the distraction level of the driver. It is noteworthy that in the case of misclassification, most instances were classified as belonging to a class adjacent to it. This problem can be overcome by maintaining a classification history. For example, the distraction level is unlikely to go from a level of 5 to a level of 1. The classification history will help to rectify possible errors made by the classifier. The determination of the perceived distraction level produced slightly better results than the driving events.

Since the confusion matrices (Table 5.8, Table 5.9 and Table 5.10) showed good results, prediction of the distraction level of the driver can be achieved by using a classifier as described in Section 5.4. However, the prediction of the driving context will need some adjustment of the classifier. Some poorly recognised driving events can be excluded, such as "Speed bump", "Right turn" and "Left turn". Speed bumps are often confused with "Straight", because when driving over a speed bump the car is moving forward. So the two events are similar. "Right turn" and "Left turn" are useful to determine the level of distraction because the driver requires full attention in order to achieve these events; however, is it not necessary to distinguish between “Right” and “Left” turn.

This driving context can be used to design a mobile, context-aware ICCS as illustrated in Figure 5.2. When the driver tries to make or receive a call, the distraction level is calculated by the classifier. If the distraction level is “Medium”, “Low” or “Very Low”, the driving event classifier is queried. If the driver is not in a position to receive any incoming communication, a suitable adaptation mechanism should be implemented by the Context Adapter. These adaptations include delaying the incoming call notification, notifying the caller by sending a text message, sharing the driving context of the driver, or pausing the dialogue in case the driver was engaged in a conversation with the system (Lindqvist & Hong, 2011).

5.9. Conclusion

An analysis of mobile phone variables was conducted and revealed that some variables were related and could therefore be substituted for one another given some minor calculations. The direction given by the GPS was related to the x axis of the orientation from the mobile phone sensor. The sensor acceleration followed the same pattern as the linear acceleration of the sensor. An inverse relationship was found between the magnetic field and the x component of the orientation (compass).

This chapter also discussed two additional experiments conducted to generate a classifier to predict the driving context. The driving context was defined to be the combination of driving events and the distraction level. A mobile application was implemented in order to facilitate the collection of data. Several machine learning algorithms were applied to determine which provided the best accuracy. The nearest neighbour algorithms, IB1 and IB3 (Table 5.5) were found to have the highest accuracy in predicting the driving events. The distraction level was predicted accurately using a nearest neighbour algorithm (KStar), with an accuracy of 95.16%. These results show that both components of the driving context classifier can be predicted accurately using the experiments discussed in Section 5.6 and Section 5.7.

A full research paper, based on the experiments conducted to determine the driving context, was presented at the 2013 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists (SAICSIT) (Tchankue, Wesson & Vogts, 2013).

The next chapter will investigate how to reduce the driver distraction. Under conditions requiring the driver's full attention, adaptation effects will be implemented in a new component of the Context-Aware Module. The algorithms that assess the risk of being distracted will also be introduced and discussed.

Chapter 6: Adaptation Effects based on Driving Context

6.1. Introduction

This chapter discusses different adaptation mechanisms that are used to reduce driver distraction. It will answer the two following research questions: *RQ7: How can an algorithm be developed to determine if the drivinbg situation is safe?* and *RQ8: How can adaptation effects be implemented to reduce driver distraction?* The methodology to be used in order to select appropriate adaptation actions will be a review of literature on similar context-aware systems.

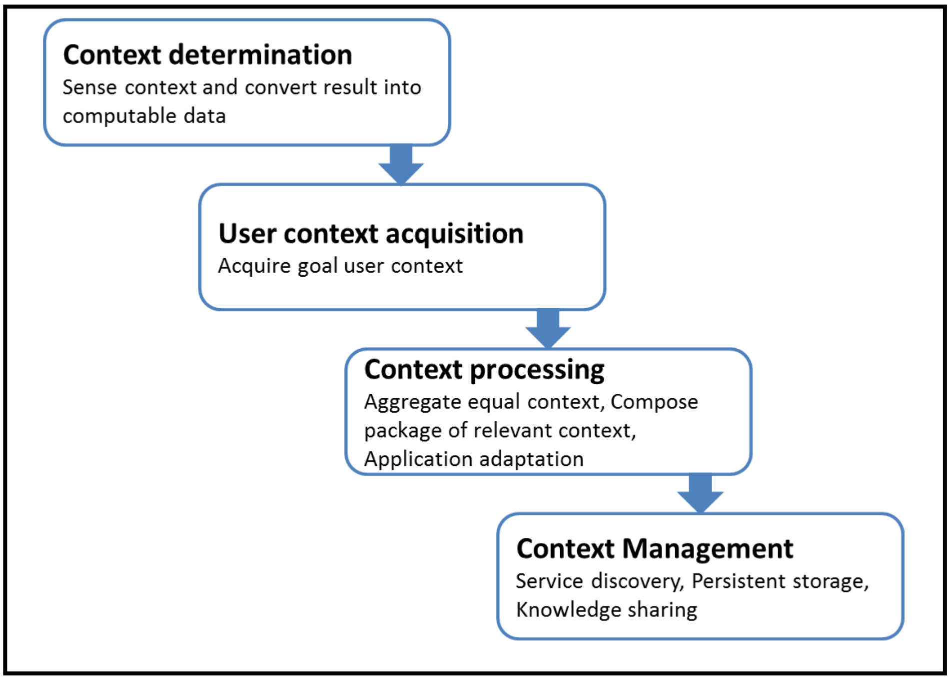


Figure 6.1: Context-Aware Operational Life Cycle (Poslad, 2009)

Research has shown that, in developing context-aware applications, issues such as inferring and fusing the context need to be addressed (Oh, Schmidt & Woo, 2007). According to the context-aware operational life cycle depicted in Figure 6.1, context determination, context acquisition of the user, context processing and context management are the main components of the process.

Chapter 5 focused on addressing the inference of the context (context determination and user context acquisition). This chapter will focus on context processing, that is finding appropriate actions so as to reduce driver distraction. The management of the context is also addressed here in order to share context information amongst parties involved in the communication.

Section 6.2 shows the updated model proposed in Chapter 5 by including adaptation effects and context sharing. Section 6.3 provides an overview of context-aware systems and the requirements for building a successful context-aware application. This will be used as a guideline to select appropriate adaptation effects. Sharing the driving context is critical when multitasking whilst driving. Section 6.4 describes the Collaborative Contrext Awareness used to design the interaction between the caller and the callee in the model for Multimodal Interface for Mobile Info-communication with Context (MIMIC). Section 6.5 defines safe driving situations as used in this project. During these safe driving situations, the system will allow telephone communications. Section 6.6 focuses on algorithms that are used to determine the safety of situations that occur during the journey. Section 6.7 describes the algorithms that were used to decide which adaptation mechanism to apply in order to reduce driver distraction. Section 6.8 details how several components of MIMIC were integrated. Section 6.9 provides implementation details of the MIMIC-Prototype. Section 6.10 provides some reflections on this chapter and Section 6.11 concludes the chapter.

6.2. Design of the Context Adaptation

The research questions answered in this chapter required some changes in the design proposed in Chapter 5. Chapter 5 described a model that can collect context information and use it to determine the driving context (driving events and distraction level). Figure 6.2 contains an updated version of Figure 5.2. The updated model takes into account the fact that the driving context has to be used in order to prevent the driver from being distracted by the mobile phone. The two models are very similar except for two additions:

- The Context Adapter in the Context-Aware Module (6): the Context Adapter will determine the adaptation mechanism to be applied,

- The peer's mobile phone (5): this will be used to implement collaborative context adaptation. Information from the callee will be used in order to determine if it is safe to allow communication.

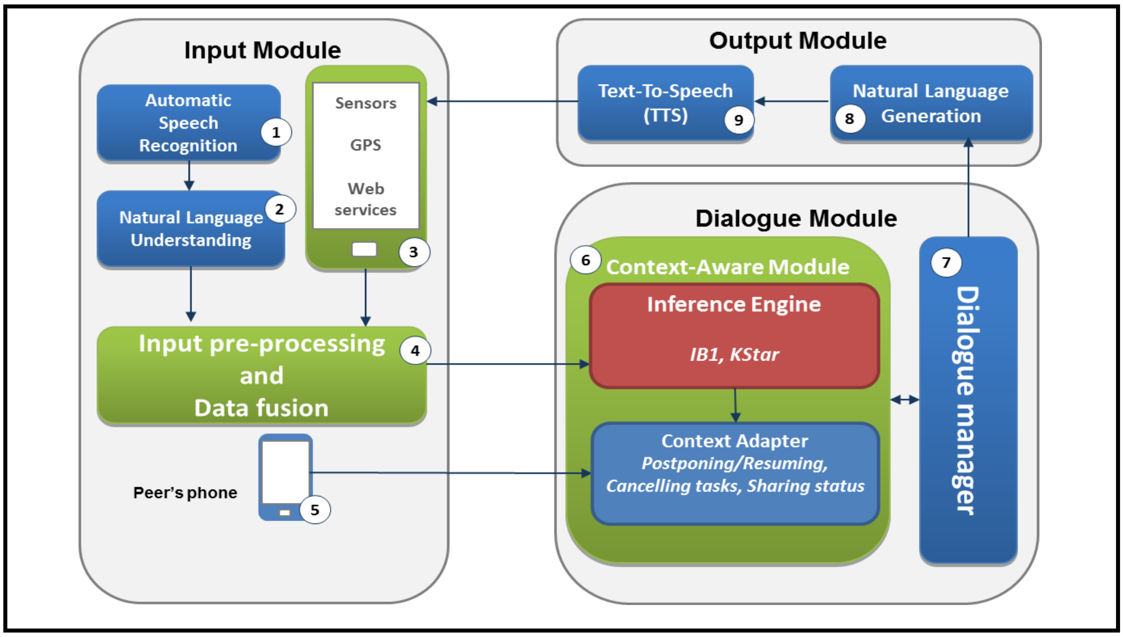


Figure 6.2: The Updated MIMIC Model Including the Context Adapter

The peer's mobile phone will share context information with the driver. This is similar to the aggregator (Section 3.4) or the Context Server that share the discovered context with several clients or applications needing that information. The intensive processing needed will quickly use up the battery life. It is therefore important to share the context only when there is a communication between the driver and a peer.

Similar to other context-aware models, the Context Adapter is part of the model. The context adapter is often not included in other models. MIMIC includes the Context Adapter because it is critical in achieving the ultimate goal, which is to reduce the distraction that a mobile phone may cause. The Context Adapter adds a collaborative dimension to the adaptation with regard to both participants in the dialogue. This is useful in the case where both drivers involved in the dialogue use the MIMIC-Prototype. The communication will only be allowed when both drivers are in situations of low driver distraction.

6.3. Revisiting Guidelines for Context-Aware Applications

As discussed in Chapter 3, various definitions of context-aware applications are available. However, two important concepts are always present in these definitions, namely using context information and adapting to context (Abowd, Dey, Brown *et al.*, 1999).

Several types of adaptation were identified in Chapter 3, including: content adaptation, user interface adaptation, functionality adaptation and device adaptation. Some of these adaptation mechanisms cannot be directly used in the case of a mobile, contex-aware ICCS. The content of the conversation (dialogue utterances from the system) may remain the same independent of the driving context. The traditional graphical user interface is non-existent because the ICCS is an eyes-free and hands-free application; which makes user interface adaptation irrelevant for this design. The functionalities and the device being used will also be independent of the driving context.

The interaction between the driver and the user should be dependent on the driving context in order to guarantee the safety of the driver. Under difficult driving situations, the interaction should be minimal compared to situations that the system has classified as being safe. This means that interaction adaptation can be highly useful in the proposed model.

Biegel (2005) derived requirements that context-aware applications should support. As discussed in Chapter 3, the architecture of a context-aware application should be loosely coupled to add flexibility, which is needed when extending or maintaining the application. The programming model chosen should be highly abstracted so as to facilitate the incorporation of new sensor data as it becomes available. Techniques for sensor fusion should also be provided in order to mitigate the uncertainty of individual sensor data. The context representation is also very important as a good context representation facilitates data processing.

The following four requirements should be followed when designing context-aware applications:

- *Context representation:* The programming model should provide an effective means to represent context information within the application. The model chosen needs to be efficient owing to the potentially large volume of data to be handled;

- *Inference engine*: A systematic and efficient approach to reasoning about context data should be provided by the programming model at a high level of abstraction,
- *Actuator abstraction*: The programming model should provide a suitable abstraction for developers to specify interaction with the environment using a range of actuator devices, including hardware and software,
- *Developer support*: An accessible and usable development environment should expose the support offered in the programming model to the application developer.

Actuators refer to actions to be performed by the application. The actuator abstraction is done at the level of the Context Adapter.

6.4. Collaborative Context Awareness

Collaborative context awareness is defined as a system that encompasses a group of entities that communicate with one another in order to achieve a common goal (Salkham, Cunningham, Senart *et al.*, 2006). The awareness of the driving situation by a callee can contribute to reducing the mental workload of the driver (Section 2.5.2). Each entity is capable of sensing, inferring and actuating. Research conducted in the area of driver distraction has highlighted collaboration as one of the solutions that can help in addressing the awareness of the driver’s situation (Iqbal *et al.*, 2011, Lindqvist & Hong, 2011). The distraction caused while a driver is talking to a passenger is lower than the distraction caused by a mobile phone conversation (Drews, Monisha & Strayer, 2008). This can be explained by the fact that a passenger is aware of the driving situation and can adjust the conversation accordingly. This suggests that a collaborative system can better address driver distraction than a non-collaborative system. This strategy has been applied to develop mobile applications that aim to prevent driver distraction. Burden-shifting was used to shift the burden (management of the call), depending on the status of the callee (available, busy or unknown) (Lindqvist & Hong, 2011).

Security and privacy are often mentioned as challenges that need to be addressed when designing context-aware applications. These are a real challenge regarding the information exchange between entities. Two entities are available in the case of the proposed system. The first entity is the driver (Driver 1) who tries to communicate with one of the contacts on the mobile phone that runs MIMIC-Prototype. The second entity is another driver (Driver 2) or a peer who may be driving or not. The mobile phone of the second entity may not run MIMIC-Prototype. In the following paragraphs, the assumption that Driver 2 uses MIMIC-Prototype is made.

Figure 6.3 depicts how the information can be shared between the two entities which, in this case, are two drivers. This collaboration is used by MIMIC-Prototype in order to determine whether the communication between the two drivers can be made without affecting their safety. The process can be broken down into four steps:

- The first step is to request the driving context from a web application hosted on a remote web server. This web application maintains a database that stores information about every registered driver,
- The web application checks the authenticity of the request and sends it to MIMIC-Prototype, running on the mobile phone of Driver 2,
- MIMIC-Prototype, running on the mobile phone of Driver 2, returns the driving context to the web application,
- The web application on the remote web server returns the result to Driver 1.

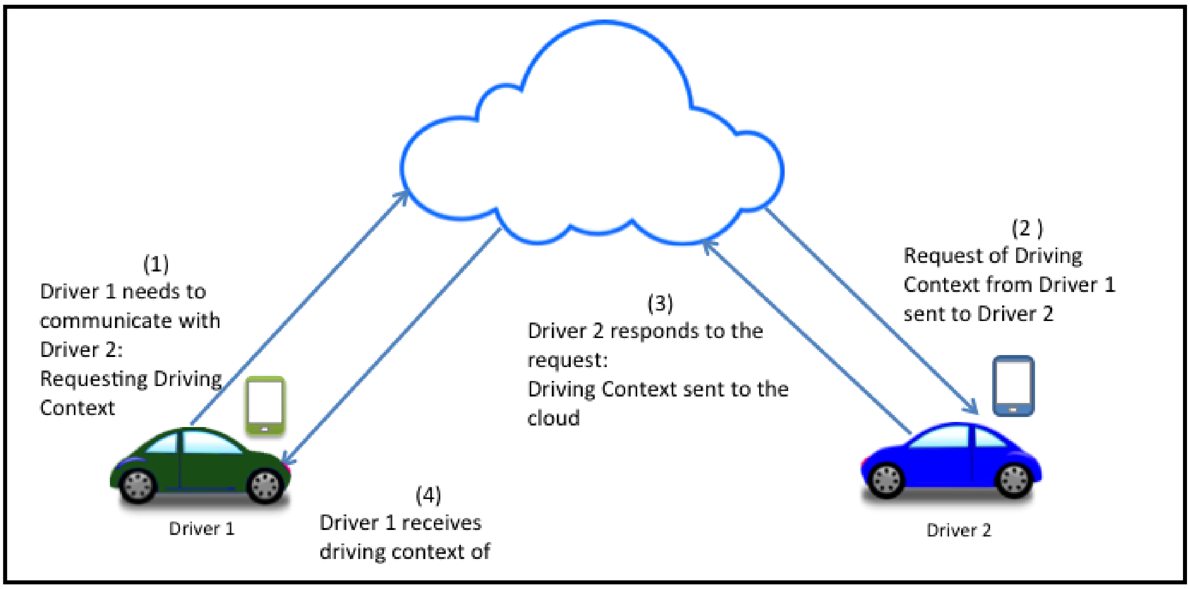


Figure 6.3: Collaboration to Share Driving Context Information between Two Drivers

This process occurs whenever a driver initiates communication with another driver. The following section describes how the safety of a driving situation is assessed by MIMIC.

6.5. Safe Driving Situations

The assessment of a driving situation can be an intricate task. The concept of the safety of a driving situation is not well defined and depends on many variables. Fuzzy logic can be used to determine whether a driving situation is safe or not providing that a well-researched membership function is used. An alternative to this is to define the safety of a driving situation depending on which situation the driver is in.

Severe weather conditions and low visibility were identified as situations where the attention of the driver should be totally focused on driving. It is therefore important to distinguish all combinations of these two variables in order to assess a driving situation. Good visibility and normal weather conditions can be treated as a normal driving situation. The presence of low visibility and severe weather conditions should be treated with extreme caution. No secondary task should be allowed, such as notifying the driver of an incoming text message.

A safe driving situation allows drivers to allocate enough attention to the primary task, which is driving. Any driving situations that need full attention from the driver will be considered as *unsafe*. It will therefore not be possible to engage in a secondary task.

6.6. Assessing the Driving Situation

This section provides details on how the safety of a driving situation is determined. This process depends on data provided by the Inference Engine. This data includes the following: distraction level, driving event, noise level, time of the day and ambient light. Visual distraction, as discussed in Chapter 2, will be addressed by applying specific actions when the visibility is reduced. The time of the day and the ambient light sensor of the mobile phone can be used in this regard, but it might not be enough to accurately predict the level of visibility.

Low visibility (Dark)	**Severe Weather conditions**	
0	0	Normal circumstances (Section 6.6.1)
0	1	Severe weather conditions (Section 6.6.2)
1	0	Low ambient light (Section 6.6.3)
1	1	Low light and severe weather conditions (Section 6.6.4)

Table 6.1: Various Contexts in which the Safety of the Driving Situation is Assessed

Astrological data, such as sunrise and sunset times, are provided by weather web services (Section 3.5.7), and can be used as a complement to the time of the day and the ambient light value. This will help in improving the determination of the visibility level. Astrological data are provided by the weather web service. Weather conditions can also cause driver distraction.

The following paragraphs describe how to determine the safety of a situation under several circumstances. Section 6.6.1 describes the process under normal conditions, that is, moderate weather conditions during daylight. Section 6.6.2 gives a description of the algorithm that is used to determine the safety of the driving situation when it is dark. The determination of the safety of the driving conditions under severe weather conditions is described in Section 6.6.3. Section 6.6.4 describes how the safety of the driving situation is determined when the weather conditions are severe and it is dark.

The assessment of the safety of a driving situation depends on the variables returned by the Inference Engine discussed in Chapter 5. These variables include the distraction level (Table 5.6) and the driving events.

The Boolean function "*FullAttention*" used in the algorithm described in Figure 6.3 returns true if the full attention of the driver is required and false otherwise. This decision is based on the driving event currently being experienced by the driver. If this driving event is different from "Stop" or "Straight", true will be returned. This means that the driver must pay more attention to the following driving events: "Turn", "Speed Bump" and "Change Lane".

The Boolean function "*Dark*" determines the level of visibility. The time of the day, the ambient light sensor and the astrological data are used to implement this function.

The Boolean function "*SevereWeather*" is used to assess the severity of weather conditions. The implementation of this function uses weather information from a web service. Weather conditions are updated every fifteen minutes.

6.6.1. Determining Safety under Normal Conditions

This section discusses the algorithm used to assess the risk of engaging in a conversation with the system during the day and under moderate weather conditions. In Figure 6.4 this corresponds to Dark = No and SevereWeather = No (1).

The driving context variables, Distraction Level (DL) and Driving Event (DE) are used in this algorithm. If the DL is "Very Low" or "Low", the driving situation is regarded as safe.

Otherwise, if the DL is "High" or "Very High" the driving situation will be considered to be unsafe for receiving calls or text messages. The third possibility to be examined is when the DL is "Medium". In this case, the system analyses the situation further by determining the DE. If the DE is "Straight" or "Stop", the situation is considered as safe, otherwise the situation will be assessed as unsafe.

This algorithm was implemented slightly differently from the description given in Figure 6.3 because the time had to be taken into account when retrieving the driving context. Instead of using the DL and DE as provided by the Inference engine, the average DL and DE was actually used. This average is calculated by using the recording taken over a period of ten seconds.

When a driver using a phone running the MIMIC-Prototype tries to make a call, the current DL is fetched from the Inference Engine. If the DL is greater than or equal to 4 ("High"), the call is delayed and put in the list of waiting outgoing calls. If the situation is safe (DL < 3), the system requests the driving context of the peer, providing that the peer has a phone running MIMIC-Prototype. If the driving context of the peer is safe, the call goes through.

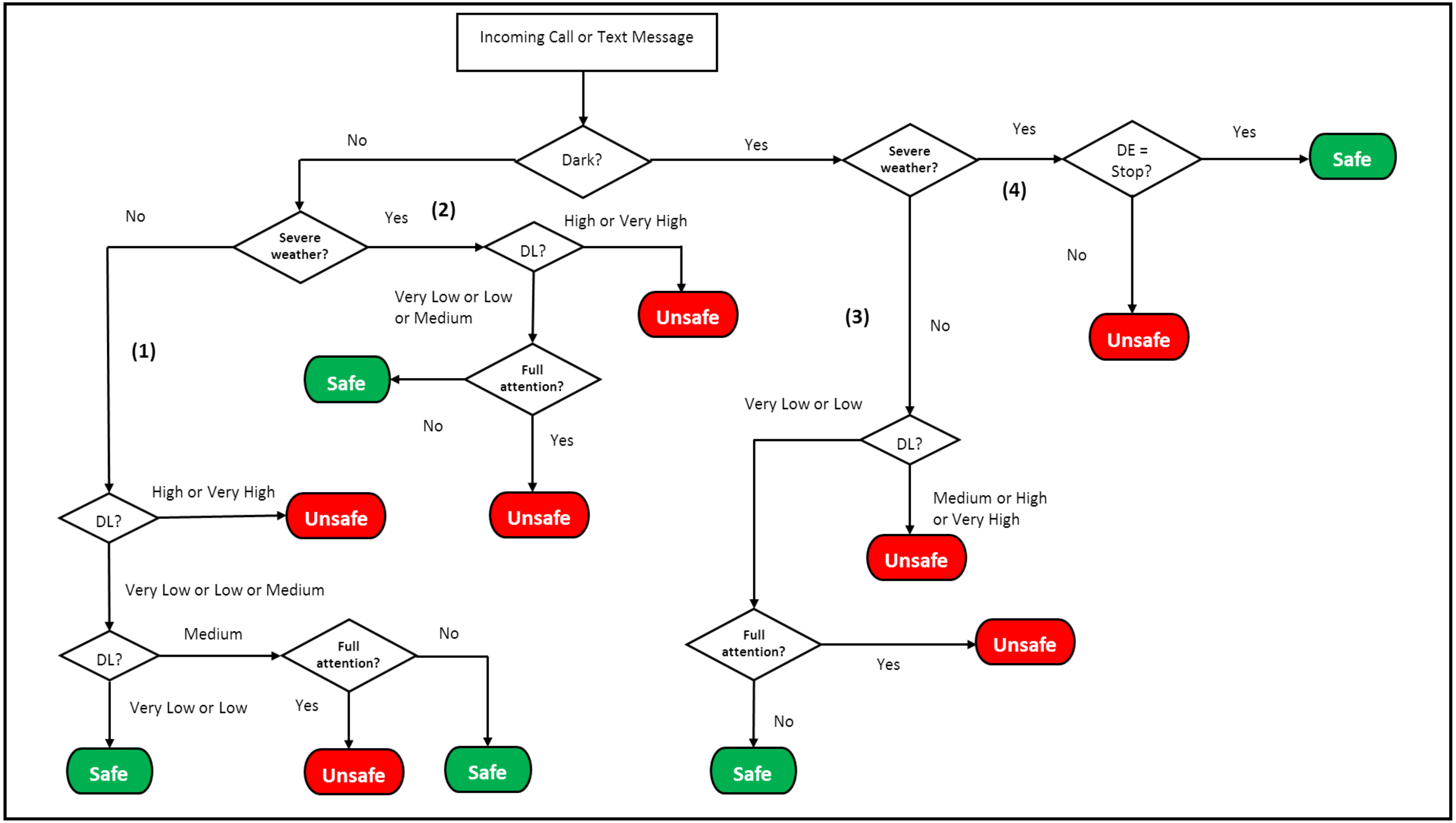


Figure 6.4: Algorithm for Classifying a Driving Situation

6.6.2. Determining Safety under Severe Weather Conditions

Severe weather conditions are more dangerous for drivers (Pisano, Goodwin & Rossetti, 2008). The driving task demand is affected by several factor including weather conditions (Section 2.4.5). This design considers the following situations as severe weather conditions:

- External temperatures > 25 degrees Celsius,
- Rainy or high humidity (> 75%),
- Windy (speed of wind > 50 km/h).

This section discusses the algorithm used to assess the safety of the driving situation under severe weather conditions. In Figure 6.4 this corresponds to Dark = "No" and SevereWeather = "Yes" (2).

If the DL is "Very Low", "Low" or "Medium", the system checks if the full attention of the driver is needed. If it is, the driving situation will be unsafe, if not it will be safe. If the DL is "High" or "Very High", the driving situation will be considered to be unsafe for receiving calls or text messages.

6.6.3. Determining Safety in the Dark

Driver distraction is affected by the time of the day (Section 2.6). At night it is more likely for a driver to be distracted by a secondary task because of the low visibility. The risk is higher in the case where the driver is an elderly person who has reduced sight (Konstantopoulos, Chapman & Crundall, 2010, Verster, Taillard, Sagaspe *et al.*, 2011).

This section discusses the algorithm used to assess the safety of engaging in a conversation with the system when the visibility level is low. In Figure 6.4 this corresponds to Dark = "Yes" and SevereWeather = "No" (3). If the DL is "Very Low" or "Low", the system checks the DE; if the DE is "Straight" or "Stop" the driving situation is safe. Otherwise, the driving situation is unsafe. In the case where the DL is "Medium", "High" or "Very High", the driving situation will be considered to be unsafe for receiving calls or text messages.

6.6.4. Determining Safety in the Dark and under Severe Weather Conditions

This section discusses the algorithm used to assess the safety of the driving situation under severe weather conditions and low visibility. In Figure 6.4 this corresponds to Dark = "Yes" and SevereWeather = "Yes" (4). In this scenario, when the weather conditions are severe and it is dark, the rigour of determining the driving situation is increased. The DL is no longer considered by the algorithm. Communication is only allowed when the car is stationary. If the DE is "Stop", then the driving situation is safe. Otherwise the driving situation is unsafe.

Algorithms discussed in previous sections are used in the adaptation mechanism. The next step is to define actions that will occur for specific adaption factors. The following sections discuss the adaptation effects that will be used in MIMIC.

6.7. Adaptation Effects

As depicted in Figure 6.1, one of the most important components of context-aware applications is the context processing, which encompasses the application adaptation. Adaptation plays a critical role in any context-aware system. Literature defines *contextual adaptation* as the ability to execute or modify a service automatically based on the current context (Abowd *et al.*, 1999). After discovering the context, the application needs to adapt its behaviour. Adaptation can be defined as a combination of three conceptual entities (Efstratiou, 2004):

- A *monitoring* entity to monitor a number of contextual attributes (the Input Module in Figure 5.2) that may trigger the application to adapt. The monitoring entity can either be part of an application or the system itself. The information monitored may be of interest to more than one application,

- An *adaptation policy* is responsible for deciding if and when the application should adapt based on the information gathered by the monitoring entity. An application is designed with a set of policies that implement the default behaviour of the application. These default policies cannot perform special purpose coordinated decisions, mainly because the application developer is not aware of the possible configuration of the target system,

- The *adaptive mechanism* performs the necessary changes when triggered by the adaptation policy. The adaptive mechanism is tightly coupled with the semantics of the application.

Section 3.6 discussed several types of context-aware applications. Retroactive adaptation requires the involvement of the driver in order to decide on the adaptation. Some authors (Poslad, 2009) use the term *passive context-aware* system to refer to systems that present the context to users to allow them to make decisions. *Proactive* adaptation mechanisms do not require the involvement of the user; all decisions are taken by the system. The system is aware of the environment context on behalf of the user, automatically adjusting the system to the context without the user being aware of it.

There are four categories of adaptation (Abowd *et al.*, 1999):

- Presentation of information and services to a user,
- Automatic execution of a service,
- Tagging of context to information for later retrieval. This is used to track objects, including people and goods in real time (passive context) (Chen & Kotz, 2000),
- Selection of information and services (Rothermel, Dudkowski, Durr *et al.*, 2003).

Table 6.2 lists the adaptations that were chosen to design the Context Adapter Module in MIMIC.

6.7.1. Adaptation 1: Postponing a Call Notification

This adaptation mechanism is triggered when the following scenario occurs. A peer decides to contact the driver by making a call. The driver is driving and the system assesses the driving situation as unsafe. The sound system of the mobile phone is muted and the call notification is delayed for later (Table 6.3, #2).

When detecting an incoming call, the system uses the information from the inference engine to assess if the situation is safe. An “Incoming Call Dialogue” is started in the case where the driving situation is safe. Otherwise the system postpones the call notification dialogue and resumes it when the situation becomes safe.

#	**Adaptation mechanisms used in the MIMIC Context Adapter Module**	**Type of adaptation**
1	Postponing a call notification when it is unsafe to take a call	Proactive
2	Postponing the text message notification when it is unsafe to receive it	Proactive
3	Notifying the user that a contact attempted to make a call	Retroactive
4	Notifying the user of a pending message that can be read if the user agrees	Retroactive
5	Pausing the dialogue when the driving situation becomes unsafe	Proactive
6	Adjusting the dialogue volume (if safe but noisy during dialogue)	Proactive

Table 6.2: List of Adaptation Mechanisms used in the Proposed Model

6.7.2. Adaptation 2: Postponing an Incoming Text Message Notification

This adaptation mechanism is triggered when the following scenario occurs. A caller decides to contact the driver by sending a text message. The driver is driving and the system assesses the driving situation as unsafe (Table 6.3, #4). The sound system of the mobile phone is muted, to prevent auditory distraction, and the text message notification is delayed for later.

When detecting the presence of an incoming text message, the system assesses the safety of the situation. In the case where the situation is safe, the dialogue is allowed and the "Incoming Text Message Dialogue" is started. Otherwise, the notification of the incoming text message is postponed until the system assesses the driving situation to be safe.

6.7.3. Adaptation 3: Notifying the Driver that a Contact Attempted to Make a Call

This adaptation mechanism is triggered when the following scenario occurs. *Adaptation 1* occurred and the driver did not get the call notification because the driving situation was determined to be unsafe. When the driving conditions return to safe (Table 6.3, #5), the sound system of the mobile phone is enabled so that the system can respond to the driver. The system then starts the "Resume Incoming Call" dialogue. In this dialogue, the system notifies the driver about the previous call attempt and provides the contact name. Then, the driver can decide whether the contact should be called back or not.

#	Incoming Event	Event Status	Safe	Action
1	Incoming call	N/A	Yes	Start "Incoming Call Dialogue"
2	Incoming call	N/A	No	Delay "Call notification"
3	Incoming text message	N/A	Yes	Start "Incoming Text Message Dialogue"
4	Incoming text message	N/A	No	Delay "Text Message notification"
5	N/A	Postponed call	Yes	Start "Resume Call Dialogue"
6	N/A	Postponed text message	Yes	Start "Resume Text Message Dialogue"
7	N/A	N/A	No	Pause Dialogue
8	N/A	N/A	Yes	Resume Dialogue
9	N/A	N/A	Yes/No	Adjust system sound

Table 6.3: The Adaptation Effects in MIMIC

6.7.4. Adaptation 4: Notifying the Driver of a Pending Message that can be Read if the User Agrees

This adaptation is triggered when the following conditions are true. Firstly, the driving situation is assessed as being safe for the driver (Table 6.3, #6). Secondly, the system received a message earlier, but *Adaptation 2* occurred because the driving situation was unsafe to start a dialogue to read the text message.

The system notifies the driver of the availability of a text message and then allows the driver to decide whether the text message should be read or not. If the driver agrees, the text message is read out aloud and removed from the queue. If the driver refuses, the dialogue is cancelled and the text message is removed from the queue.

6.7.5. Adaptation 5: Pausing the dialogue when the driving situation becomes unsafe

This adaptation is triggered when the system and the driver are conversing, and the driving situation becomes unsafe (Table 6.3, #7). The system pauses (interrupts) the dialogue; mutes the sound system and saves the current state of the dialogue. The dialogue is resumed as soon as the situation is assessed to be safe.

6.7.6. Adaptation 6: Adjusting dialogue volume (if not distracted, but noisy during dialogue)

The interaction between the system and the driver is exclusively speech-based. It is important for the driver to be able to hear the conversation with the system. In this case the driver and the system are the only entities involved in the dialogue. The conversation can be affected negatively by a noisy environment, which is common in a driving situation. The driver can speak louder in order to interact with the system. Adaptation 6 deals with the system adjustment of the volume to enable the driver to hear the responses of the system clearly (Table 6.3, #9).

Six adaptation algorithms were defined, which were integrated with the MIMIC-Protototype. The following paragraph discusses the how the final system was integrated.

6.8. Integrating the Dialogue System into MIMIC-Prototype

Chapter 4 recommended the need for a solution for speech recognition errors. This recommendation was made because the usability evaluation identified a high number of speech errors (Section 4.5). These errors occurred when using specific commands, such as the “REDIAL” command, and when dictating telephone numbers. The “REDIAL” command and the telephone number dictation were therefore not considered in this implementation in order to prevent too many speech recognition errors.

The speech recognition can be improved significantly when the API allows the creation of a speech profile. The speech profile builds a training set that is used by the speech engine in order to customise the algorithm. The speech engine captures the accent of the user and adjusts to the way the user pronounces specific phrases. Unfortunately, the Android speech API does not currently allow the creation of a speech profile.

The speech recognition rate was improved by using the following process. A larger number of people used the system and the recognition results were logged. As a result of that, several variations of same utterances were logged. This was used to correct erroneous recognition results. The analysis of the recognition results were used to tune the natural language understanding (NLU) to make sure that the most credible result was returned as being the best result.

Although the Android speech application programming interface (API) was used to implement this version of the prototype, Android 4.1.2 [ref] (url: http://developer.android.com/about/versions/android-4.1.html) works differently from Android 2.3 (Google Inc, 2011). The speech engine stops listening a few seconds after being invoked. The developer has to invoke the speech recogniser periodically to ensure that the speech recognition is done continuously.

6.9. Implementation of the Context Adapter Module of MIMIC-Prototype

The proposed model for a mobile, context-aware in-car communication system was implemented as MIMIC-Prototype. This was done to assess the feasibility of the Context Adapter and also to evaluate its contribution in reducing driver distraction.

Features	**Descriptions**
Display	4.8 inch High Definition Super AMOLED (1280x720) display
Sensor	Accelerometer, Red Green Blue (RGB) light (proximity), Digital compass, Proximity, Gyro, Barometer
Connectivity	Wi-Fi a/b/g/n, Wi-Fi HT40; GPS/GLONASS; Near Field Communication (NFC); Bluetooth® 4.0 (LE)
Operation system	Android 4.1.2

Table 6.4: Device Specifications for the Implementation of MIMIC-Prototype

Table 6.4 shows the specifications of the Samsung Galaxy S3 that was used for implementing the prototype. The operating system provided the speech recognition API. The noise suppressor was also provided by the operating system. The following sensors were available on the device: accelerometer, gyroscope, compass and proximity.

The Android mobile platform was chosen to implement MIMIC-Prototype. The reason for this choice was the availability of the Android operating system (OS) with several mobile phone manufacturers as well as the easy access to development tools such as Eclipse and Android Studio (Android Studio, 2013). The availability of documentation and programming resources was also critical in choosing Android.

The Inference Engine was implemented as a service that launches at boot time. This means that whenever the user switches on the mobile phone, the service starts together with other system services such as Google Maps. This allows the system to monitor the speed of the car and to be aware of the beginning of a journey. When the Inference Engine service detects motion, the system prompts the user to enable MIMIC-Prototype. This addresses the issues with existing mobile applications that are used to disable communication whilst driving. The user can forget to do so sometimes and be put at risk of being distracted during the journey.

6.9.1. Collaboration

The collaboration is managed by the Context Adapter Module. The design of this component of the prototype required a remote server running an application that can receive and authenticate requests for driving context; obtain the information from the peer and return the result to the driver who requested it. This process depends on the speed of the connection; hence getting information can be delayed. However, in most cases this information was obtained within a second.

The design shown in Figure 6.2 uses the cloud to implement the collaboration. For the sake of simplicity, the communication between drivers was implemented using Bluetooth. An Android Service was implemented (*MyBluetoothService*) to manage the exchange of the driving context data. This component encompasses the following functions:

- *Starting the Bluetooth server and listening to incoming messages*: The device starts a thread that allows it to be discovered by other Bluetooth devices. The server listens permanently to incoming Bluetooth connections,

- *Starting Connection*: The device starts a connection thread to specific paired Bluetooth devices that are in range,

- *Sending a message to connected mobile phones using MIMIC-Prototype*: The mobile phone sends a message to a peer mobile phone, which has to be paired prior to the field study. The messages that can be sent include requests for driving context and the driving context information.

In the field study, the driver and the peer will be driving in the same vehicle. A Bluetooth connection will be enough for a proof of concept.

6.9.2. Context Monitoring and Reasoning

The Inference Engine was implemented as an Android service. This service is started at boot time and runs in the background. This guarantees that the context is monitored and inferred at all times as long as the phone is running. An earlier implementation highlighted the issue of battery life. The phone tries to determine the context even if the phone owner is not in a moving car.

The implementation of the Inference Engine was done with the help of a Java class called ***MyInferenceEngine.*** The main methods of *MyInferenceEngine* are the following:

- *Loading machine learning models (DE and DL):* This method loads the distraction level model (KStar) and the models of the driving events (IB1) that were obtained as a result of the two experiments discussed in Chapter 5,

- *Setting all variables*: This method collects data from the Input Module. The data comes from the mobile phone sensor (accelerometer, gyroscope, compass, proximity and ambient light), the global positioning system (GPS) (speech, altitude and orientation) and weather (ambient temperature, humidity, etc.). Specific Android services were implemented to collect data from each of the sources,

- *Classifying*: This method uses the models that were loaded previously in order to classify the current data. This returns the current distraction level (DL) and the driving events (DE),

- *Retrieving*: This method provides the current DE and DL to any other component that needs these variables.

The monitoring and the reasoning of the context take place continuously. This provides data needed by MIMIC to implement context-awareness.

6.9.3. Context Adapter

This module implements the adaptation effects that are available in the system. The safety of the driver or the Peer is checked by a method. In the case of the Peer, a driving context request is sent to the cloud and the Peer returns a list containing the history of the driving context. In the field study the driver and the Peer were in the same car; therefore, for the sake of simplicity, a Bluetooth chat was implemented to handle the communication between the two entities.

A Java class (*MyAdaptationEffects*) was designed in order to provide the following functions:

- *Determining safety*: This method uses algorithms described in Section 6.6 to determine whether the driving context of the driver is safe or not,

- *Postponing an event*: This method stores an incoming event in a queue. This occurs when the driving conditions are unsafe. The notification of the driver is postponed until the driving conditions are safe and the event is kept in a queue (Adaptation 1 or 2),

- *Resuming an event*: This method releases an event from the queue. This event was previously kept in the queue and when the situation reverts to safe, this method removes the communication event from the queue,

- *Retrieving the driving context of the Peer*: This method requests the driving context from the Peer driving context. The process described in Section 6.4 is used for this purpose (Figure 6.3),

- *Providing current driving context*: This method sends the current driving context to the other entity (Peer). This is an implementation of the context sharing or collaboration.

As mentioned in Chapter 5, the variation in predictions provided by the machine learning classifier cannot be used directly because of a possible inaccuracy issue. To solve this problem, a history of driving context has to be kept (time awareness) in order to make sure that the application uses the most accurate prediction.

6.9.4. Dialogue Manager

The Dialogue Manager was updated by adding some new dialogue frames. The frames "Resuming all" and "Resuming Text Message" were implemented in order to enable the context adaptation necessary to reduce driver distraction. The following example dialogues were implemented (Figures 6.5 – 6.9).

Dialogue for making a call
Driver: Call Maria
MIMIC-Prototype: Do you want to call Maria? Say "Yes" or "No"
Driver: "Yes"
MIMIC-Prototype: Calling Maria

Figure 6.5: Example Dialogue for Making a Call

Dialogue for answering a call
MIMIC-Prototype: You have a call from "John". Do you want to answer? Say "Yes" or "No"
Driver: "Yes"
MIMIC-Prototype: You are now connected to "John"

Figure 6.6: Example Dialogue for Answering a Call

Dialogue for resuming a call
MIMIC-Prototype: You had a call from “John” earlier. Do you want to call back? Say “Yes” or “No”
Driver: “Yes”
MIMIC-Prototype: You are now connected to “John”

Figure 6.7: Example Dialogue for Resuming a Call

Dialogue for sending a text message
Driver: Text Maria
MIMIC-Prototype: Please choose an option 1. I will call you when I get there 2. I can’t talk right now, I am driving 3. I am running a few minutes late 4. I am on my way
Driver: “Four”
MIMIC-Prototype: Do you want to send the message, “I am on my way” to Maria? Say “Yes” or “No”
Driver: Yes
MIMIC-Prototype: Sending the message “I am on my way” to Maria.

Figure 6.8: Example Dialogue for Sending a Text Message

Dialogue for resuming a text message
MIMIC-Prototype: You had a text message from "John" earlier. Do you want me to read it? Say "Yes" or "No"
Driver: "Yes"
MIMIC-Prototype: "John" wrote "Thank you for participating in this research project"

Figure 6.9: Example Dialogue for Resuming a Text Message

6.10. Discussion

Figure 6.10 depicts the updated version of MIMIC. This model was updated from the version described in Figure 6.2 to incorporate the changes in the Context-Aware Module. The design of the Context Adapter was completed with two additional modules: Collaboration and Adaptation Effects.

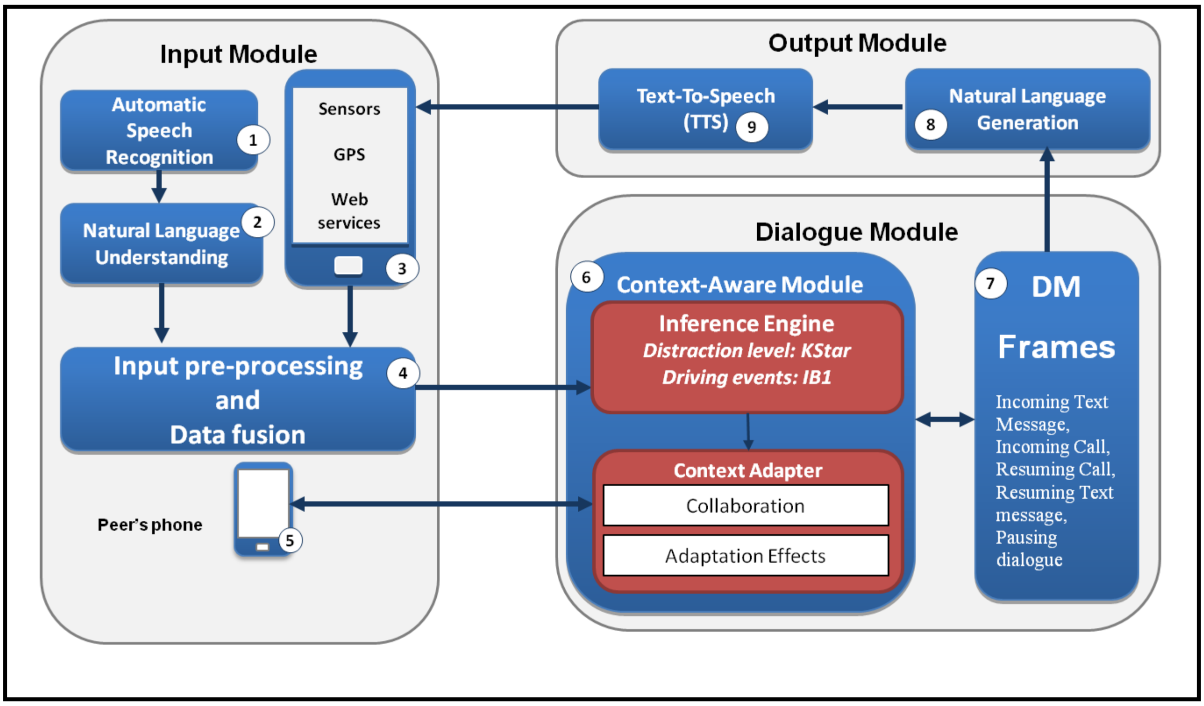


Figure 6.10: Updated Architecture of MIMIC

The determination of which adaptation effect to apply can be a difficult task. Measuring driver distraction can be difficult.

6.11. Conclusion

Adaptation mechanisms are the only actions that are noticed by the user of a context-aware application. It is therefore very important to design the adaptation effects carefully in order to reduce driver distraction. The awareness of the Peer is an important factor when a driver engages in a phone conversation. It is important to implement a collaborative context-awareness. Four algorithms were described in this chapter to determine the safety of engaging in a conversation.

The main contributions of this chapter are the following: the use of collaborative context-awareness to determine the safety of a driving situation; the description of four algorithms used to determine the safety of a driving situation; the introduction of six adaptation effects (context-aware interaction) that the Context Adapter of MIMIC will use; the implementation of the final version of the prototype ready to be evaluated; and the use of time awareness and the dialogue history to prevent errors when using driving context.

The implementation of the final version of the prototype was achieved successfully. The next chapter discusses a field study, which aimed to assess to what extent the methods described in this chapter can reduce driver distraction.

Chapter 7: Evaluation

7.1. Introduction

The aim of this chapter is to answer the research question *RQ9: To what extent does the MIMIC-prototype reduce driver distraction?* The answer to this question will be provided by the analysis of the data captured during a field study.

This chapter is structured as follows: Section 7.2 discusses some evaluation methods, Section 7.3 discusses the research design and Section 7.4 discusses the actual field study. This includes the selection of participants, the apparatus, the procedure, the tasks to be performed and the metrics. Section 7.5 summarises the results obtained in the field study, this includes performance and self-reported results. Section 7.6 contains a discussion on the results of the field study and Section 7.7 concludes the chapter.

7.2. Evaluation Methods

There are several methods that can be used to evaluate the interaction between users and mobile systems (Table 7.1). As a mobile application, the MIMIC-Prototype can be evaluated using one of these methods, which are grouped by the type of settings used during the study, namely natural, artificial and environment-independent settings. Natural settings are used for testing theories and hypotheses. The benefit is the production of rich data that are useful to test hypotheses. The setting sometimes makes it difficult to collect data and ethics clearance is required to prevent a negative impact on participants. Artificial settings are controlled and used for theory and product testing. It is difficult to generalise the results of such studies because laboratory conditions are not the same as in practice. An artificial setting was used for the usability evaluation of a speech-based mobile ICCS in Chapter 4. Environment-independent settings are also used for product development and theory building. It is often easy and inexpensive to conduct such studies, but the shortcomings include a high risk of failure, possibility of redesign and outcomes influenced by opinions.

For natural settings, action research, case studies and field studies are the methods that are used most often to conduct studies. Results obtained from action research and case studies are difficult to generalise. A field study was selected as an appropriate method to evaluate the MIMIC-Prototype because a new application was being tested. Therefore a natural setup will guarantee that similar experiments could be replicated.

Methods		Strengths	Weaknesses	Use
Natural settings	Case studies	Natural setting, rich data	Time demanding, Limited generalisability	Descriptions, explanations, developing hypothesis
	Field study	Natural setting, replicable	Difficult data collection, unknown sample bias	Studying current practices, Evaluating new practices
	Action research	First-hand experience, applying theory to practice	Ethics, bias, time, unknown generalisability	Generate hypothesis/theory Testing theories/hypothesis
Artificial settings	Laboratory experiments	Control of variables, replicable	Limited realism, unknown generalisability	Controlled experiments, Theory/product testing
Environment independent settings	Survey research	Easy, low cost, can reduce sample bias	Context insensitive, no variable manipulation	Collecting descriptive data from large samples
	Applied research	The goal is a product, which may be evaluated	May need further design to make product general	Product development, testing, hypothesis/concepts
	Basic research	No restriction on solution, solve new problems	Costly, time demanding, may produce no solution	Theory building
	Normative writing	Insight into first-hand experience	Opinions may influence outcomes	Description of practices, building frameworks

Table 7.1: Mobile Human-Computer Interaction (HCI) Research Methods (Kjeldskov & Graham, 2003)

7.3. Field Study

The field study was performed on an urban road depicted in Figure 7.1 and Figure 7.2. Using a rural road for this experiment, it would have been difficult to experience several road situations. The route driven by drivers was 9 kilometres long and a small section (1.9 kilometres) of that route was a freeway. The speed limit on the freeway is 120 km/h and the speed limit on the remainder of the route is 60 km/h. It was a high traffic road that presented a variety of driving situations. Test sessions did not take place during peak hours; hence participants did not have to be in congested traffic.

Participants were familiar with the route used for the field study. It was therefore easy to plan the journey on a strategic level (Section 2.3.1). The field study was conducted during the day at times out of rush hour. The weather was generally fair without rain and strong wind.

Figure 7.1: Tasks and Route Followed during Trip 1

The complexity of having to adapt to a new car was removed by allowing participants to use their own cars. It was therefore easy for participants to perform driving tasks on an operational level (Section 2.3.3).

The test moderator sat in the passenger seat and did not speak with the driver throughout the entire trip. This was done to minimise the level of driver distraction that the participants might experience. Performance data were collected using log files saved in a dedicated folder on the mobile phones of the participants.

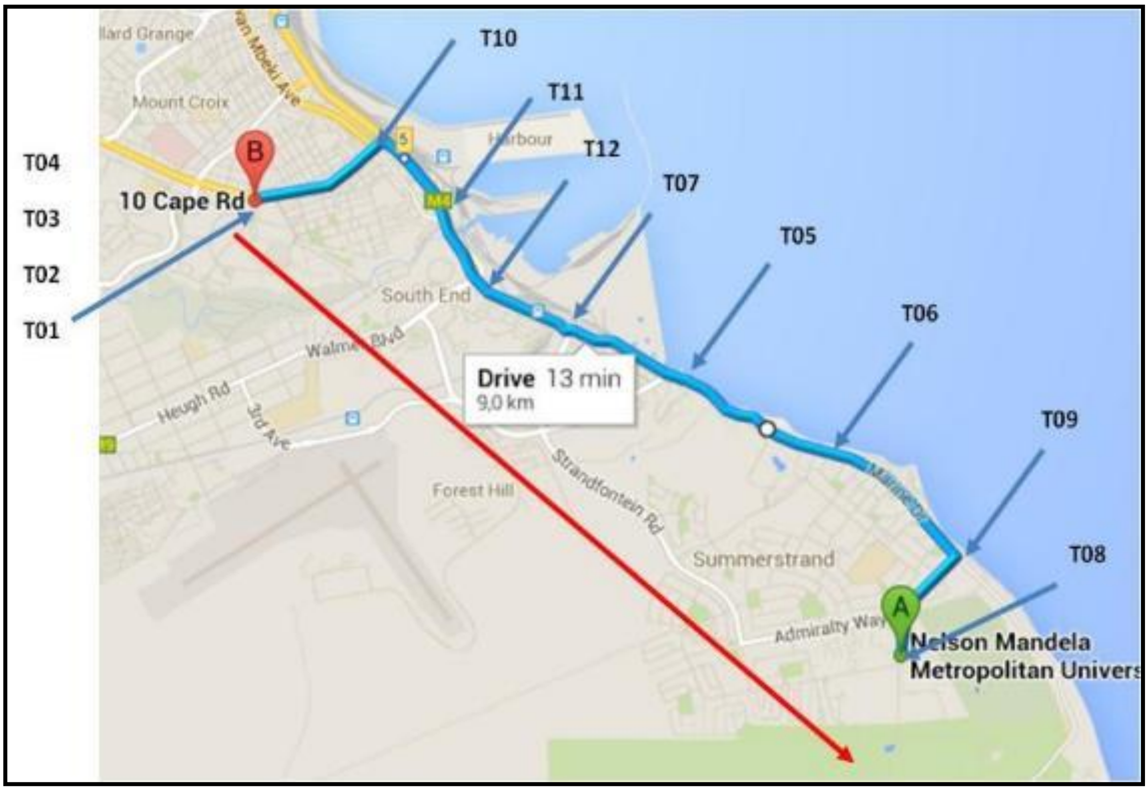


Figure 7.2: Tasks and Route Followed during Trip 2

7.4. Research Design

Following the implementation and the integration of the Context-Aware Module into MIMIC-Prototype, a field study was conducted to compare two versions of MIMIC-Prototype. The first version adapts to the driving context (Adaptive version) while the second does not adapt (Non-Adaptive version). The comparison was conducted in terms of performance, satisfaction metrics and driver distraction. The experiment was designed using a within-subjects approach with *counter-balancing* whereby each participant had to use both versions of MIMIC-Prototype in a different order. Half of the participants used the adaptive version in the first trip and the non-adaptive version in the second trip; the other half used the non-adaptive version in the first trip and the adaptive version in the second trip.

An application for the field study was submitted to the Nelson Mandela Metropolitan University (NMMU) Research Ethics committee for approval. The high accuracy of the determination of the driving context (Section 5.6.4 and Section 5.7.2) was used as a motivation that the participants would be safe during the course of the field study. The committee approved the application (See Appendix E).

The research design followed in the field study is discussed in the following sections. This includes the selection of the participants, the apparatus, the procedure, the tasks performed and the metrics used to compare the two versions of MIMIC- Prototype.

7.4.1. Selection of Participants

Most participants in this field study were students or staff members of the Science Faculty at the NMMU. The following graphs provide information about the gender distribution, the age distribution, the first language spoken by participants, the mobile phone experience prior to this field study and the driving experience of each participant prior to the field study.

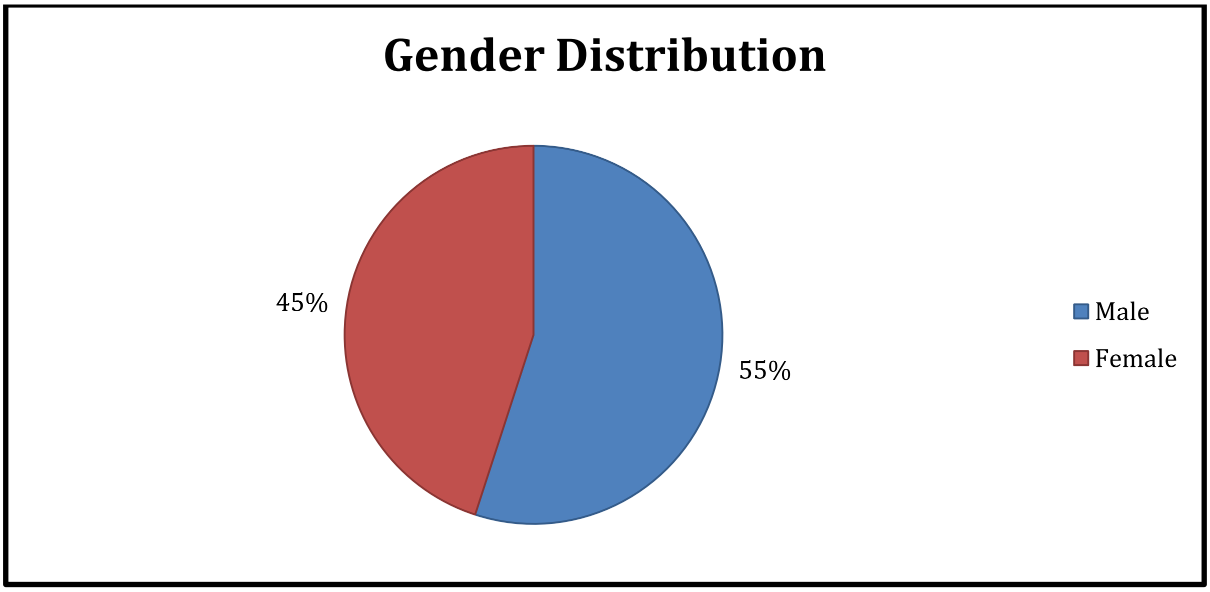


Figure 7.3: Gender Distribution (n=20)

Figure 7.3 shows the gender distribution amongst the participants. Overall, fifty-five per cent (55%) of the participants were male and 45% were female. Most participants were young drivers. As shown in Figure 7.4, a total of 70% of participants were between 21 and 29 years of age. This is an important fact, because most drivers who frequently use their mobile phones while driving are in this age category (Lenhart, Ling, Campbell *et al.*, 2010, Tison, Chaudhary & Cosgrove, 2011, Zhao *et al.*, 2013). Only 10% of the participants belonged in the age category from 30 to 39 years old. Ten per cent of participants were in the age group 40 - 49 and 10% were in the age group of more than 50 years old.

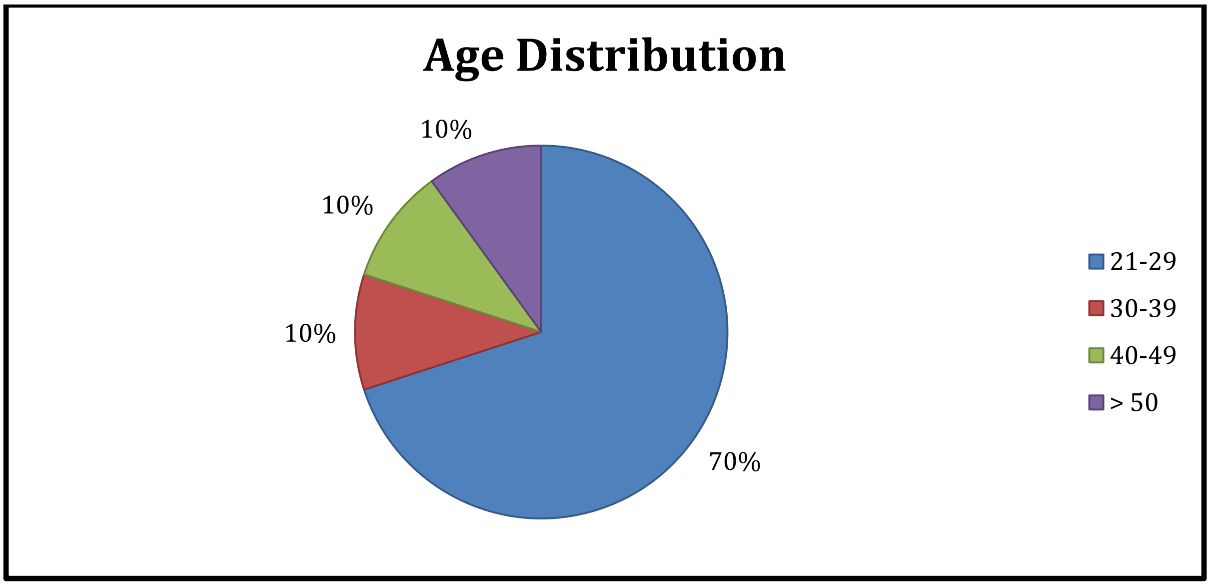


Figure 7.4: Age Distribution (n=20)

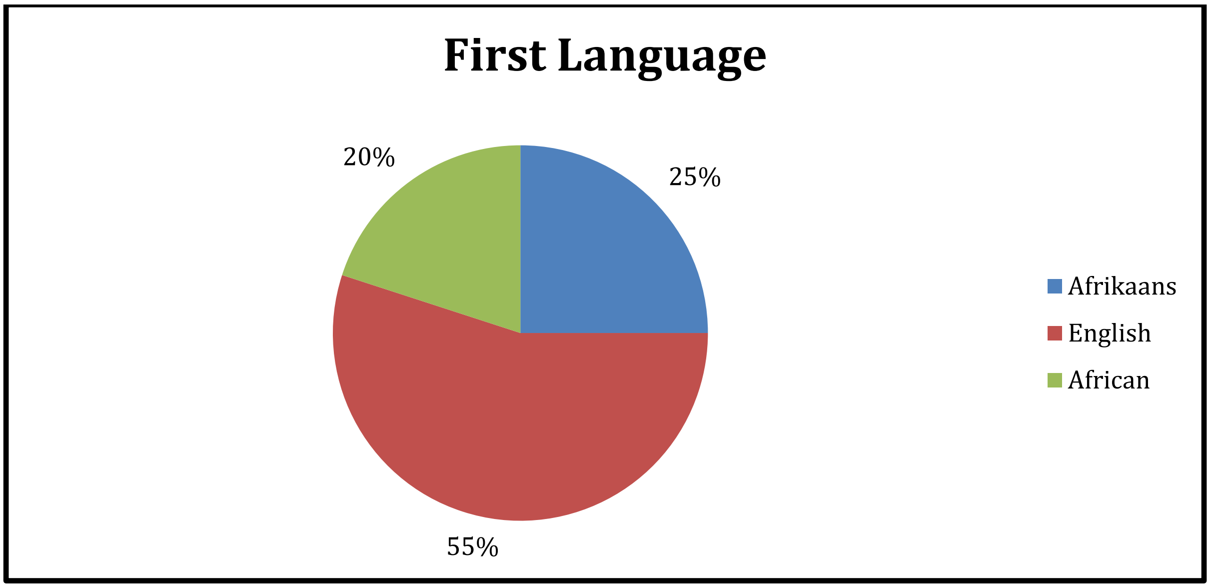


Figure 7.5: First Language Distribution (n=20)

Since the experiment involved the use of a speech-based system, the first language of the participants was an important variable that could influence the recognition rate of the system. MIMIC was implemented to recognise English. As shown in Figure 7.5, up to 55% of participants spoke English as a first language. Twenty five percent (25%) were Afrikaans speakers and the remaining 20% spoke an African language (isiXhosa, Setswana or isiZulu).

As the system being studied runs on a mobile phone, it was very important that the participants understood the use of a mobile phone. According to Figure 7.6, most participants (90%) had at least used a mobile phone for 5 years. The remaining 10% had used a mobile phone for between 1 and 2 years (five per cent) and between 3 and 5 years (five per cent).

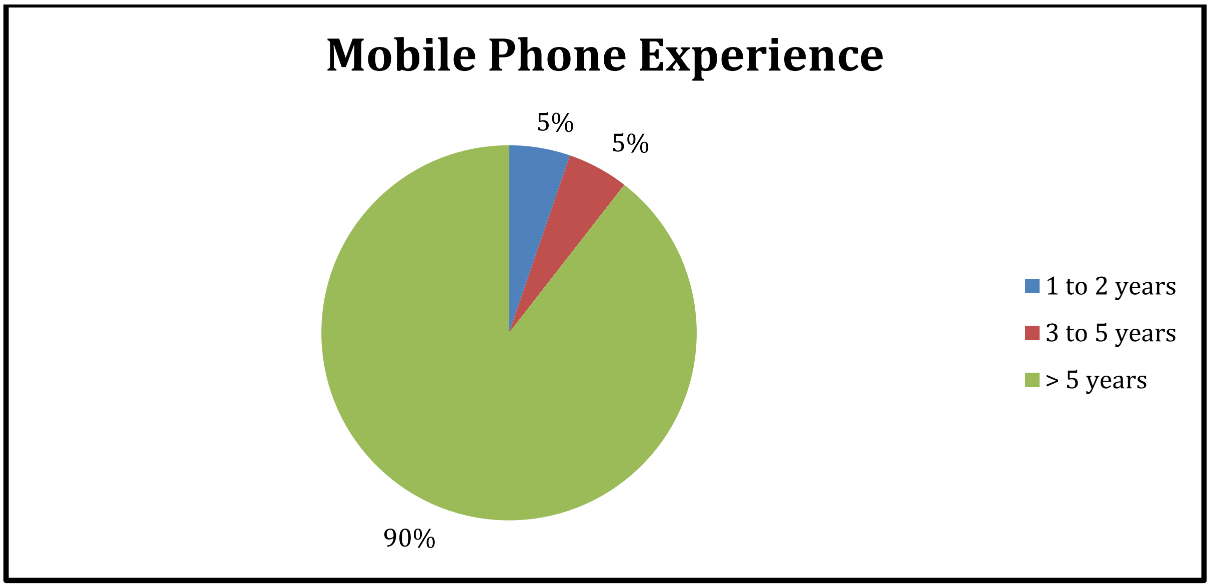


Figure 7.6: Distribution of Mobile Phone Experience (n=20)

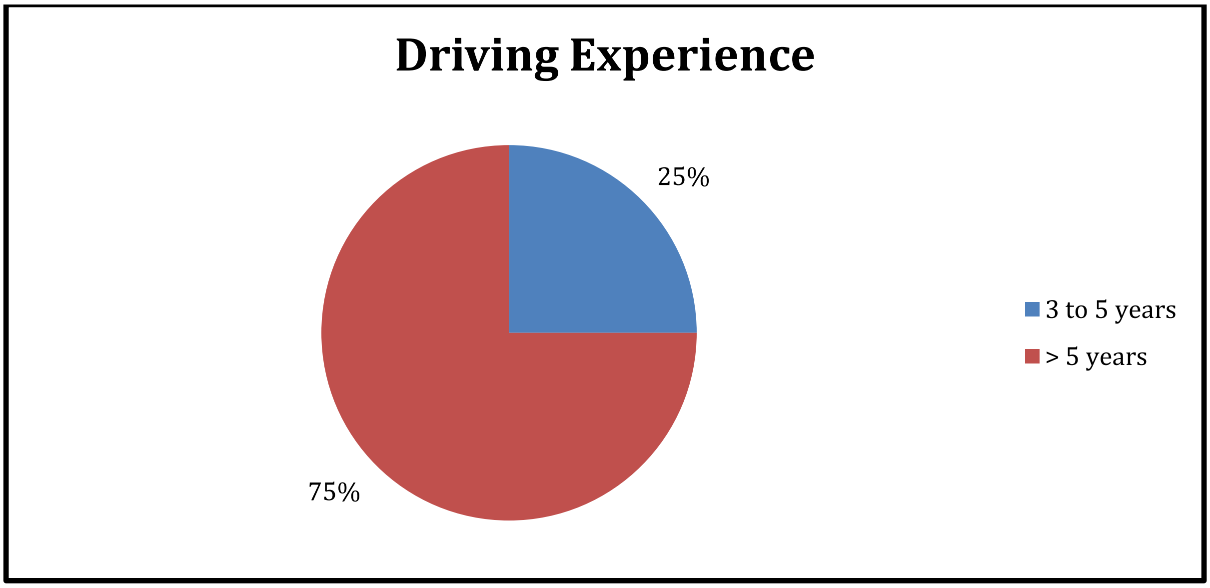


Figure 7.7: Distribution of Driving Experience (n=20)

Driving experience is an important variable of the system being investigated as the system is used mostly while the driver is on the road. Limited driving experience could affect the ability of the driver to engage effectively in a secondary task while driving.

Figure 7.7 shows the distribution of participants with regard to their driving experience. Driver experience has the potential to affect the operational level of the driving model (Section 2.3.3). All drivers were in possession of a valid South African driving license for at least 3 years. Seventy-five per cent of the participants reported having at least 5 years driving experience.

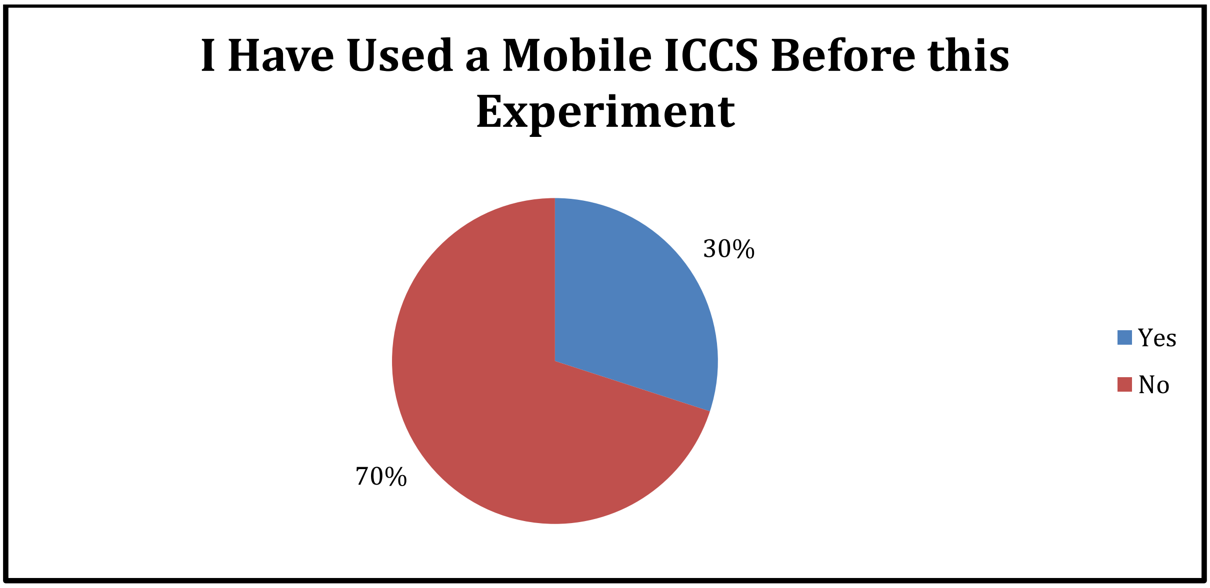


Figure 7.8: Distribution of Prior Experience with a Mobile ICCS (n=20)

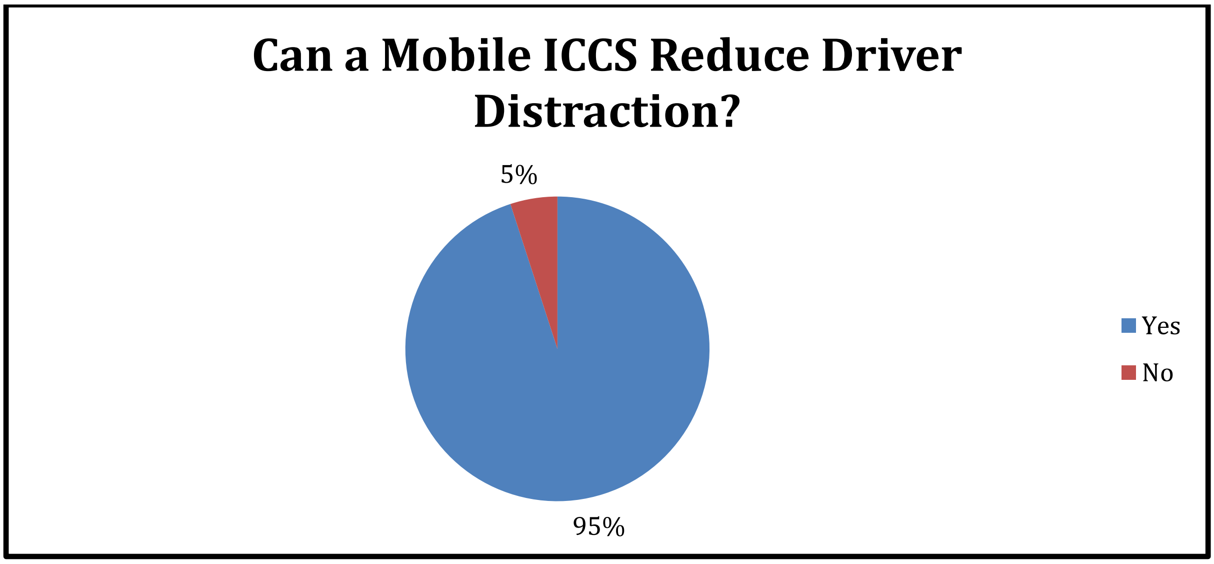


Figure 7.9: Opinion about the Ability of a Mobile ICCS to Reduce Driver Distraction (n=20)

Drivers with prior experience of ICCS do not need to learn much to be able to use MIMIC-Prototype effectively. The participants were asked whether they had used a similar application in the past. Seventy per cent had no prior experience with an ICCS, whilst only 30% had used an ICCS previously (Figure 7.8).

Participants were also asked whether they believed that a mobile ICCS could reduce driver distraction. The answer was predominantly "Yes" (90%); only five per cent of the participants did not agree that a mobile ICCS could reduce driver distraction if the driver engages in a secondary task while driving (Figure 7.9).

7.4.2. The Apparatus

The aim of the field study was to identify the benefits of the adaptive version of the MIMIC-Prototype implemented using the design presented in Chapter 6. This was done using the application in the field. The following items were necessary for conducting the field study:

- *A vehicle*: the type of car was not important; any car could have been used to perform this field study,

- *A mobile phone running MIMIC-Prototype*: this mobile phone was a Samsung Galaxy S3 that was positioned where the speech of the driver could be easily picked by the microphone of the phone,

- *A mobile phone running the monitoring application*: this application connects the phone of the test moderator to the phone of the driver in order to give instructions when it is appropriate. A task was started by the instruction that the test moderator gave by pressing a button on the monitoring application. The mobile application told the driver exactly what to say or what to expect, for example, "*Task 1. Say: Call Maria*" or "*Task 2. You will receive a call*".

A Bluetooth headset was used at first, but it was found to be problematic because it could not fit easily on the ears of all drivers. It was therefore decided to place the mobile phone under the sun visor. The mobile phone had to be fixed with an elastic cord to prevent it from falling when the vehicle was moving. During the test session, the test moderator was seated on the passenger seat and did not interact with the driver.

7.4.3. Procedure

The procedure followed during this field study is shown in Figure 7.10. Prior to the beginning of a test session, the participant was welcomed in the NMMU parking area and the purpose of the study was explained to all the participants.

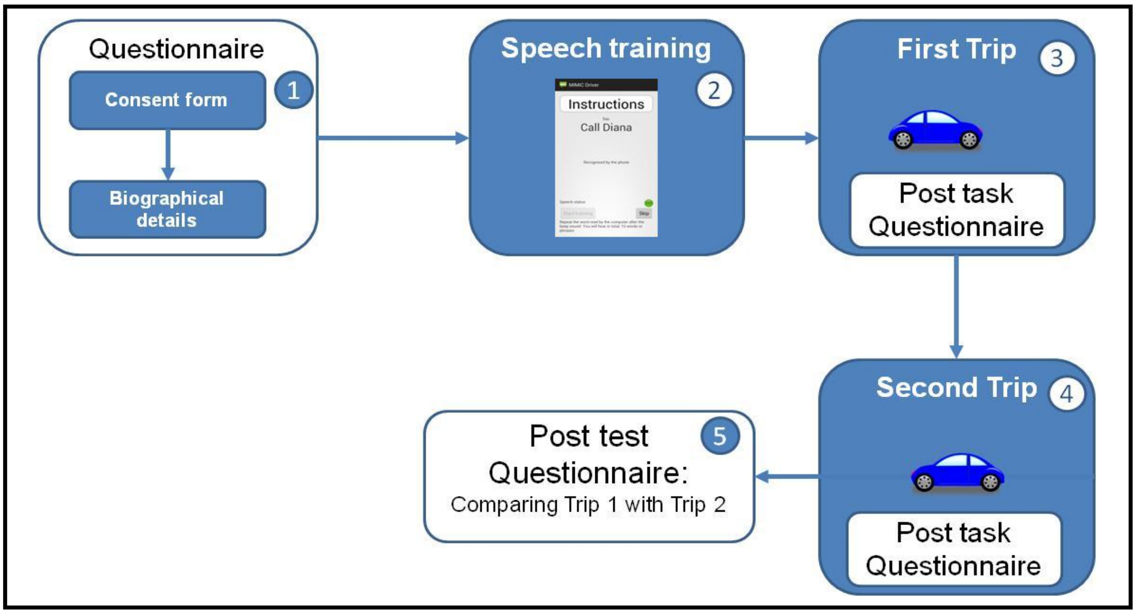
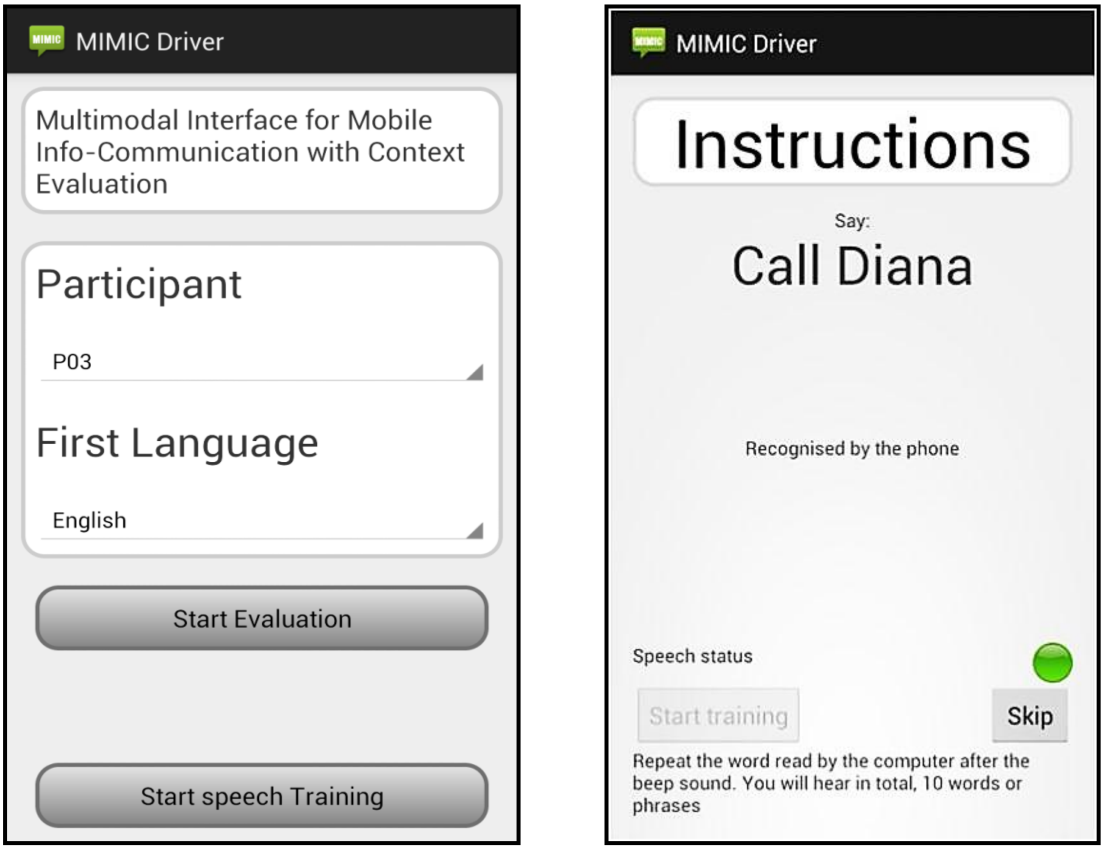


Figure 7.10: Procedure Followed during the Field Study

Each participant was required to sign a consent form and complete a biographical information form (1). A speech training session (2) was conducted with the participants to show how to engage in dialogue with the system. The participant started performing the tasks using the first system. The first four tasks were performed with the car stationary and the remaining eight tasks with the car moving. After the twelfth task of each trip, the participants were asked to complete a post-test questionnaire (Appendix C). Then an equivalent set of tasks using the second system was performed and another post-task questionnaire was answered. Lastly, the participant answered a post-task questionnaire (Appendix D), asking which trip participants preferred in terms of receiving calls and text messages as well as making calls and sending text messages. Each test session lasted approximately one hour.

Figure 7.11 illustrates the screen that was presented to the test moderator when the evaluation started. The participant's ID and the first language were captured. This was used to create a log file that contained all data related to the current participant. This ensured that the participant could use the prototype and understood how to interact with the system.



a) Starting the Evaluation, Choosing the Participant ID and the Language

b) Speech Training

Figure 7.11: Settings and Speech Training

The speech training started when the user pressed the button "Start Speech Training". The button "Skip" could be used to skip a phrase that the system could not recognise. A "beep" sound was heard when the system recognised anything, whether the recognised phrase was correct or not. The participant had to say the phrase that appeared on the screen. When the phrase was well recognised; the system notified the participant by saying "Correct". A total of ten words were available and a final score was given to the participant at the end of the process.

Two version of MIMIC-Prototype were implemented, Figure 7.12 shows the main screen of the MIMIC-Prototype used by the driver. It was used for debugging purposes and every utterance was sent to the mobile phone of the test moderator. Three pieces of information were displayed on top of the screen: the Bluetooth status, the Internet connection status and the current speed. The Bluetooth connection status showed if the current phone was connected to the mobile phone of the test moderator. It could also display messages that were sent or received. The Internet connection status was shown by displaying a round green image when the phone was connected and a round red image in case the Internet connection could not be established or was lost. This was essential because speech recognition uses the Internet connection. At the bottom of the screen, the current driving context, the current event and the distraction level were displayed. In the middle of the screen, the actual dialogue between the participant and the system was displayed. The dialogue was represented by a list view made of bubbles of different colour. The text coming from the system was displayed in blue and the text coming from the participant was in grey.

Figure 7.12: Main Screen showing the Dialogue with MIMIC-Prototype and a Driver

The main screen of the monitoring version of MIMIC-Prototype (discussed in Section 7.4.2 Apparatus) is shown in Figure 7.13. It helped the test moderator to monitor the driving context. This allowed the test moderator to send a Bluetooth message (task number) to the mobile phone of the participant. That mobile phone received the task number that was to be performed and read it out to the participant. This was done to reduce driver distraction that could occur if the test moderator spoke to the participant or if the participant read the tasks from a task list.

In order to facilitate navigation through the twelve tasks, the buttons “Next” and “Previous” were implemented. Two other functional buttons were implemented to make calls and send text messages to the mobile phone of the participant. The button “Start” was used to give an instruction to the participant about the next task to be performed.

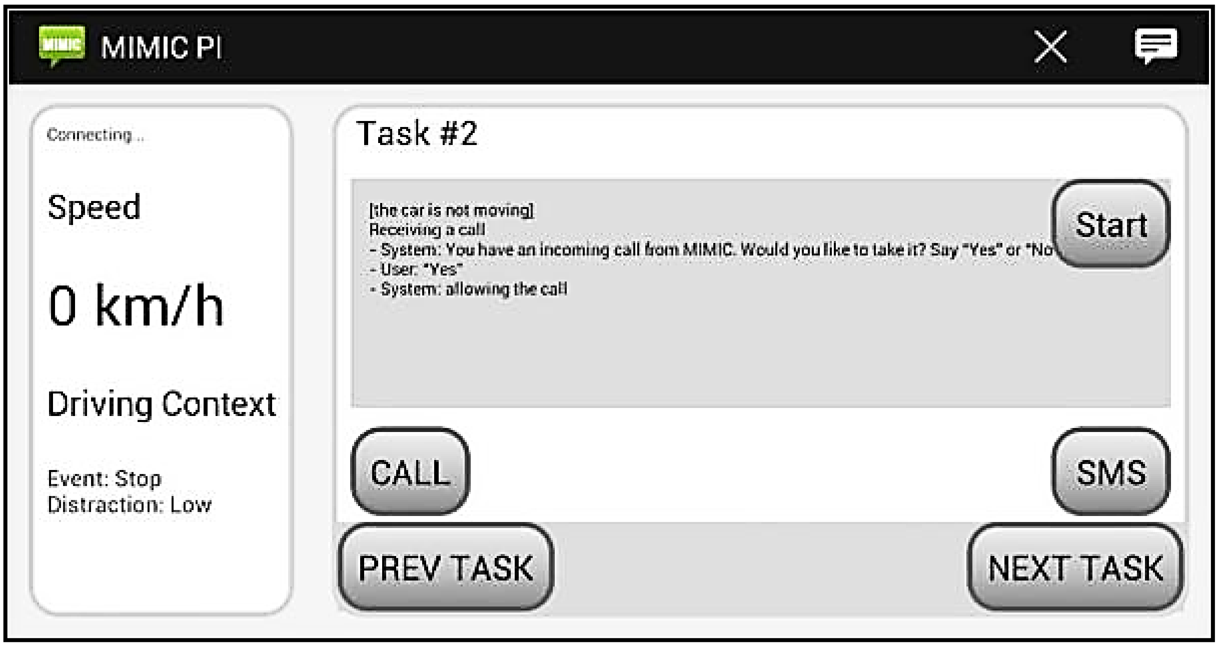


Figure 7.13: Main Screen of the Monitoring Application

7.4.4. Tasks Performed

All participants had to perform several tasks using both versions of the MIMIC-Prototype. The set of tasks performed for the two versions was equivalent, but slightly different. The names and telephone numbers were different.

Table 7.2 lists the tasks that were performed by each participant. These tasks were either driver-initiated or peer-initiated. Driver-initiated tasks were outgoing events, that is, tasks where the participant decided to issue a command and included tasks such as making calls and sending text messages. Peer-initiated tasks were incoming events, that is, tasks that were not triggered by the participant, such as receiving a text message and answering a call.

Four types of communication tasks were considered for this study, namely: making a call, sending a text message, receiving a call and receiving a text message. Each of these categories had three tasks, which occurred under different circumstances. The first task occurred when the car was not moving, the second one when the car was moving and in the third case, when the car was moving and there was a potential distraction.

Task ID	Description	Initiated by	Driver status	Peer status
T01	Please call the contact Maria - Say "**Call Maria**"; - Answer "**Yes**"	Driver	Safe	Safe
T02	Incoming call: - System notification - Answer "**Yes**" or "**No**"	Peer	Safe	Safe
T03	Send any of the text messages available to John: - Say "**Text John**"; - Say "**Yes**" to send the selected text message	Driver	Safe	Safe
T04	Incoming text message: - System notification - Answer "**Yes**" or "**No**"	Peer	Safe	Safe
T05	Please call the contact Bob - Say "**Call Bob**"; - Answer "**Yes**"	Driver	Safe	Safe
T06	Incoming call: - System notification Answer "**Yes**" or "**No**"	Peer	Safe	Safe
T07	Send any of the text messages available to Peter: - Say "**Text Peter**"; - Say "**Yes**" to send the selected text message	Driver	Safe	Safe
T08	Incoming text message: - System notification - Answer "**Yes**" or "**No**"	Peer	Safe	Safe
T09	Please call the contact Diana - Say "**Call Diana**"; - Answer "**Yes**"	Driver	Safe	**Unsafe**
T10	Incoming call: - System notification - Answer "**Yes**" or "**No**"	Peer	**Unsafe**	Safe
T11	Incoming text message: - System notification - Answer "**Yes**" or "**No**"	Peer	**Unsafe**	Safe
T12	Send any of the text messages available to John: - Say "**Text John**"; - Say "**Yes**" to send the selected text message	Driver	Safe	**Unsafe**

Table 7.2: List of Tasks Performed during the Field Study

7.4.5. Metrics

The goal of the field study was to investigate the possible benefits gained using the adaptive version of MIMIC-Prototype. Performance and self-reported metrics were captured. Some of the variables that were analysed were mentioned in Section 2.7.2 as benefits of multimodal interfaces. The performance metrics included the following:

- *Time on task (seconds)*: The time spent in performing a task. This provided a measure of the efficiency. The less time spent on a task the more efficient were the participants,

- *Completion rate*: Indicates whether the task was completed or not. This provided a measure of the effectiveness,

- *Success rate*: Indicates whether the task was successfully completed or not. This provided an indication of the number of tasks that were successfully completed,

- *Number of errors*: The number of errors made while performing a task. This provided a measure of the errors made by participants on specific tasks,
- *Mental workload*: The mental demand of the tasks,
- *Physical workload*: The physical demand of the tasks,
- *Temporal workload*: The pace of the tasks,
- *Performance*: The success in accomplishing the tasks,
- *Effort:* The work put into the accomplishment of the level of performance,
- *Frustration:* The insecurity, annoyance, and irritation experienced whilst completing the tasks,
- *Ease of use of the prototype:* The extent with which participants could complete the tasks easily,
- *Efficiency of the prototype*: The ability to perform tasks with speed and precision,
- *Effectiveness*: The ability to successfully achieve every task,
- *Effectiveness in determining safe situations*: The ability to determine safe situations accurately,
- *Effectiveness in implementing adaptation effects:* The ability to apply the right adaptation effects,

- *Distraction level*: The perceived distraction level determined by the Inference Engine. This provided an estimation of safety when participants performed the tasks.

The self-reported metrics were used to measure the usability, the workload and the driver distraction. Considering that usability can be seen as a measure of simplicity to reach a specific goal using specific technologies (Constantinos & Kim, 2006), it was important to capture usability data to make sure that MIMIC-Prototype could be used easily by the participants. The System Usability Scale (SUS) (Brooke, 1996) was used as a tool to assess the usability of the two versions used in the field study.

7.5. Results

The results are presented in three sections, namely performance, cognitive load and satisfaction. Graphs and tables showing the results are discussed. Since a comparison of two versions of MIMIC-Prototype was done, the independent *t-test*, also called the two sample t-test or the student t-test was used. This test is an inferential statistical test that determines whether there is a statistically significant difference between the means in two unrelated (unpaired) groups. The t-test provides good results when comparing small samples, generally $n < 30$ (Tullis & Albert, 2008). The null hypothesis H_0 was that the adaptive and the non-adaptive versions of MIMIC-Prototype are equivalent.

7.5.1. Performance Results

Performance results were logged during the field study. The following variables were captured for analysis purposes: time on task, completion rate, success rate and error rate.

7.5.1.1 Time on Task

Figure 7.14 depicts the time on task recorded for tasks that consisted of making a call while using the adaptive version and the non-adaptive version (Table 7.2).

While performing T01, the car was not moving. A t-test conducted on this task showed the observed difference was not significant (t (19), p = 0.7) between the adaptive version and the non-adaptive version, because drivers completed this task with almost similar times when using the adaptive version (mean = 13.61, median = 10.00) and the non-adaptive version (mean = 12.47, median = 10.00).

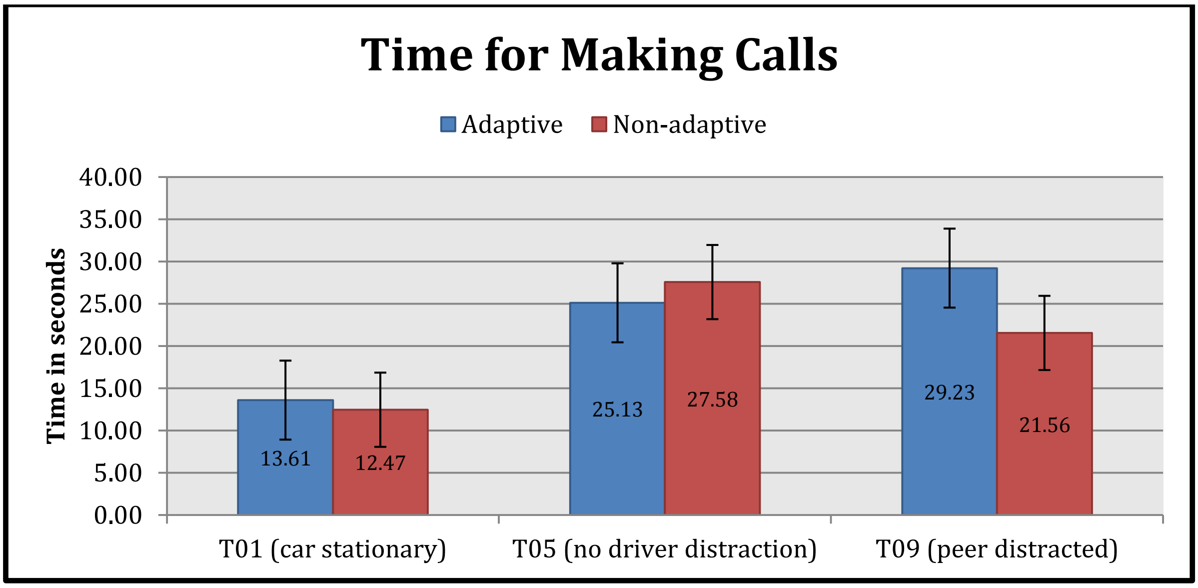


Figure 7.14: Time on Task for Making a Call (n=20)

T05 consisted of making a call while driving, but neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed that the observed difference between the adaptive version and the non-adaptive version was not significant (t (19), p = 0.78). When using the adaptive version (mean = 25.13, median = 19.00) the average time for all participants was slightly lower than for the non-adaptive version (mean = 27.58, median = 26.00).

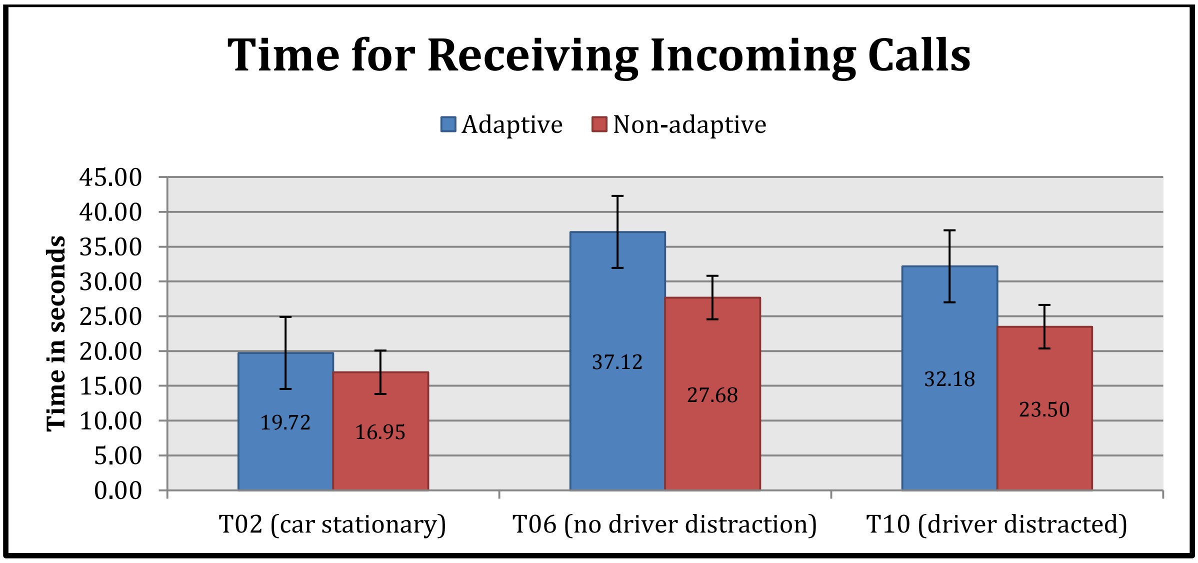


Figure 7.15: Time on Task for Receiving a Call (n=20)

T09 consisted of making a call while the driving conditions were unsafe for the peer (callee). A t-test conducted on this task showed no significant difference (t (19), p = 0.99) between the adaptive version and the non-adaptive version. It took slightly more time when using the adaptive version (mean = 29.23, median = 17.00), compared to the non-adaptive version (mean = 21.56, median = 16.00). This was due to the fact that the adaptive version took some time to communicate with the peer in order to obtain an assessment of the driving context. The dialogue was only started when the system had assessed the driving situation to be safe for both parties.

Figure 7.15 summarises the results obtained for the tasks in the category of receiving a call using the adaptive version and the non-adaptive version. The time was calculated from the moment the call was made to the moment the system allowed the driver to talk with the peer. While performing T02, the car was not moving. A t-test conducted on this task showed no significant difference (t (19), p = 0.43) between the adaptive version and the non-adaptive version. The time spent on T02 was higher for the adaptive version (mean = 19.72, median = 16.50) than for the non-adaptive version (mean = 16.95, median = 15.00).

T06 consisted of receiving a call from a peer while driving; however, neither the driver nor the peer was distracted while performing this task. Although the t-test conducted on this task showed no significant difference (t (19), p = 0.08) between the adaptive version and the non-adaptive version, the p-value was very close to the decision value (0.05). When using the adaptive version (mean = 37.12, median = 34.00), the average time for all participants was higher than the time that it took to perform the task using the non-adaptive version (mean = 27.68, median = 25.00).

T10 consisted of receiving a call while the driving conditions were unsafe for the driver. A t-test conducted on this task showed no significant difference (t (19), p = 0.19) between the adaptive version and the non-adaptive version, but it took a lot more time when using the adaptive version (mean = 32.18, median = 26.00), as compared to using the non-adaptive version (mean = 21.56, median = 16.00). This is due to the fact that the adaptive version postponed the call notification until the driver was able to answer the call safely. The dialogue was only started when the system assessed the driving situation as being safe for the driver.

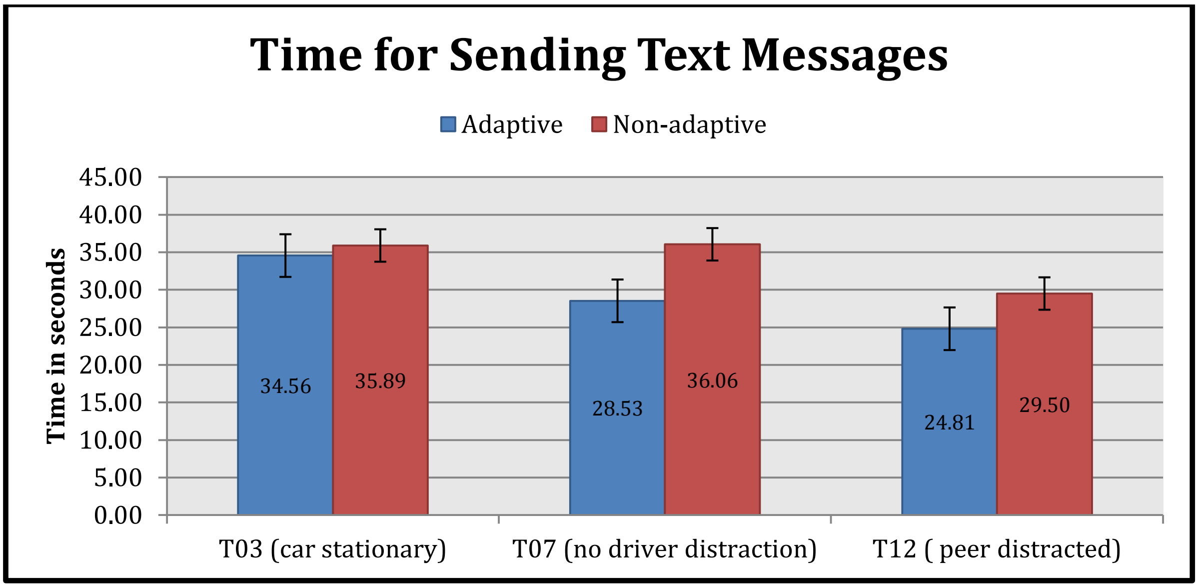


Figure 7.16: Time on Task for Sending a Text Message (n=20)

Figure 7.16 summarises the results obtained for the tasks in the category of sending a text message using the adaptive version and the non-adaptive version. While performing T03, the car was stationary. A t-test conducted on this task showed no significant difference (t (19), p = 0.54) between the adaptive version and the non-adaptive version. The duration of T03 was slightly lower for the adaptive version (mean = 34.56, median = 31.50) than for the non-adaptive version (mean = 35.89, median = 34.00).

T07 consisted of sending a text message to a peer while driving; though neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 0.89) between the adaptive version and the non-adaptive version. When using the adaptive version (mean = 28.53, median = 26.00), the average time for all participants was lower than the time that it took to perform the task using the non-adaptive version (mean = 36.06, median = 28.00).

T12 consisted of sending a text message while the driving conditions were unsafe for the peer. A t-test conducted on this task showed no significant difference (t (19), p = 0.59) between the adaptive version and the non-adaptive version. It took less time when using the adaptive version (mean = 24.81, median = 20.50), as compared to the time taken for the non-adaptive version (mean = 29.50, median = 27.50).

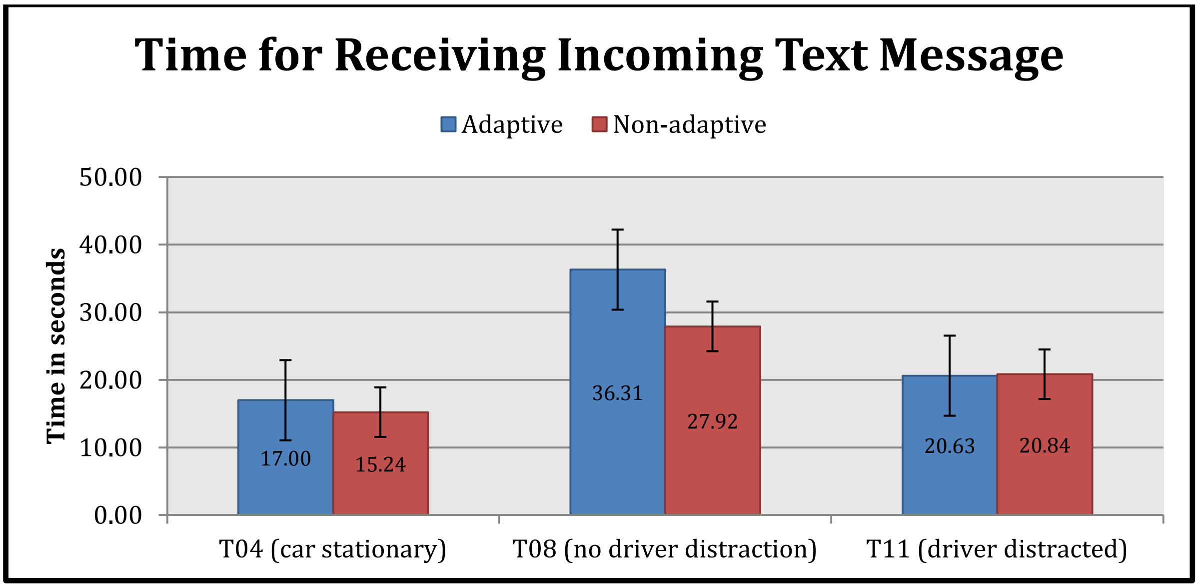


Figure 7.17: Time on Task for Receiving a Text Message (n=20)

Figure 7.17 summarises the results obtained for the tasks in the category of receiving a text message using the adaptive version and the non-adaptive version. While performing T04, the car was stationary. A t-test conducted on this task showed no significant difference (t (19), p = 0.54) between the adaptive version and the non-adaptive version. The duration of T04 was slightly higher for the adaptive version (mean = 17.00, median = 14.00) than for the non-adaptive version (mean = 15.24, median = 14.00). The adaptive version took additional time because the safety of the situation was checked once a second.

T08 consisted of receiving a text message from a peer while driving; however, neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 0.15) between the adaptive version and the non-adaptive version. When using the adaptive version (mean = 36.31, median = 31.50), the average time for all participants was higher than the time that it took to perform the task using the non-adaptive version (mean = 27.92, median = 15.00).

T11 consisted of receiving a text message while the driving conditions were unsafe for the driver. A t-test conducted on this task showed no significant difference (t (19), p = 0.70) between the adaptive version and the non-adaptive version. It took less time when using the adaptive version (mean = 20.63, median = 17.50), as compared to the time taken for the non-adaptive version (mean = 20.84, median = 19.00).

7.5.1.2 Completion Rate and Success Rate

Almost all tasks were successfully completed during this field study (Table 7.3). Some tasks were not completed because of poor speech recognition. The prototype did not allow the participant to attempt the same task more than three times.

Task ID	Completion rate (%)
T01 (making a call)	100
T02 (receiving a call)	100
T03 (sending a text message)	100
T04 (receiving a text message)	100
T05 (making a call)	95
T06 (receiving a call)	95
T07 (sending a text message)	90
T08 (receiving a text message)	100
T09 (making a call)	90
T10 (receiving a call)	100
T11(receiving a text message)	95
T12 (sending a text message)	90

Table 7.3: Mean Completion Rate (n=20)

Figure 7.18 summarises the results obtained for the tasks in the category of making a call using the adaptive version and the non-adaptive version. While performing T01, the car was stationary; the success rate was hundred per cent when using both versions. Despite a small difference in the duration of the tasks (Figure 7.15), all participants managed to make the call successfully.

T05 consisted of making a call to a peer while driving; however, neither the driver nor the peer was distracted while performing this task. This task was found to be more challenging for participants; the average success rate was 80% when using the adaptive version and 75% when using the non-adaptive version.

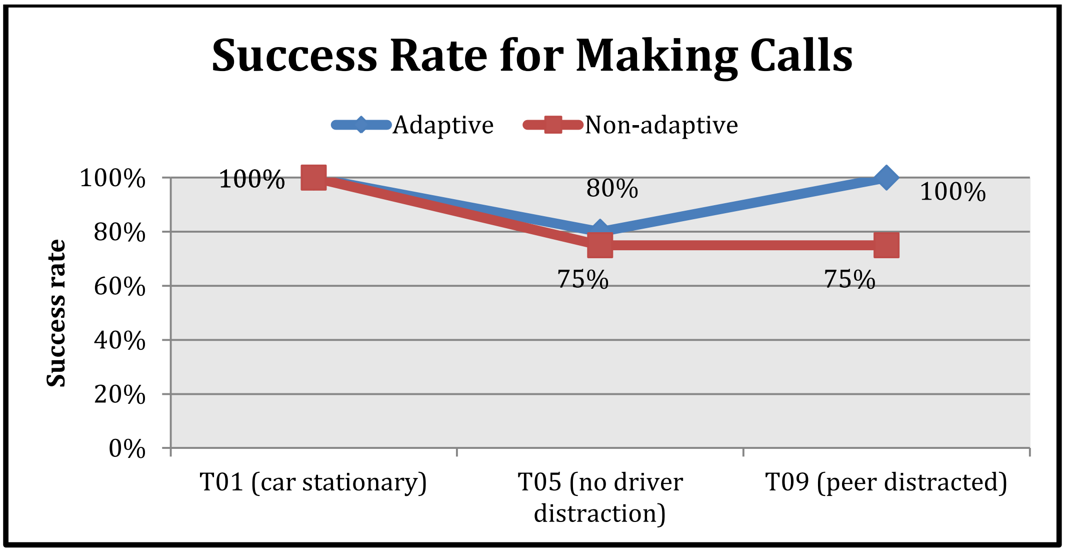


Figure 7.18: Success Rate for Making a Call (n=20)

T09 consisted of making a call while the driving conditions were unsafe for the peer. The t-test conducted on this task showed no significant difference (t (19), p = 0.16) between the adaptive version and the non-adaptive version. Participants managed to perform better when using the adaptive version (100%), as compared to the non-adaptive version (75%).

Figure 7.19 summarises the success rate obtained for the tasks in the category of receiving a call using the adaptive version and the non-adaptive version. Success in receiving a call was determined when the driver accepted to receive the call.

While performing T02, the car was stationary; the success rate was 100% when using both versions. Despite a small difference in the duration of the tasks (Figure 7.19), all participants managed to receive the call successfully.

T06 consisted of receiving a call from a peer while driving; however, neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 1.00) between the adaptive version and the non-adaptive version. The average success rate was 90% when using the adaptive version and 80% when using the non-adaptive version.

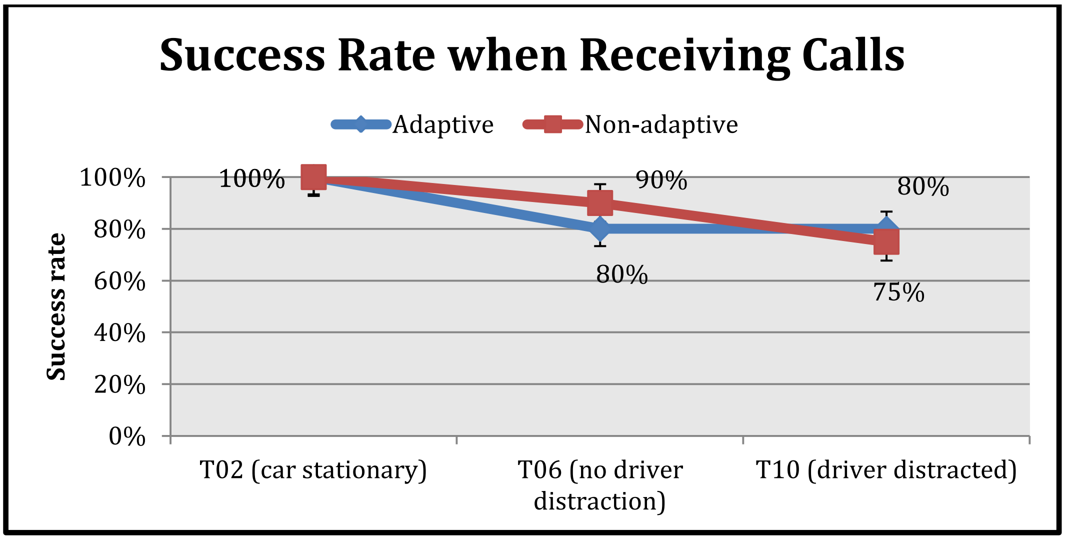


Figure 7.19: Success Rate for Receiving a Call (n=20)

T10 consisted of receiving a call while the driving conditions were unsafe for the driver. Participants managed to perform marginally better when using the adaptive version (80%), as compared to the non-adaptive version (75 %).

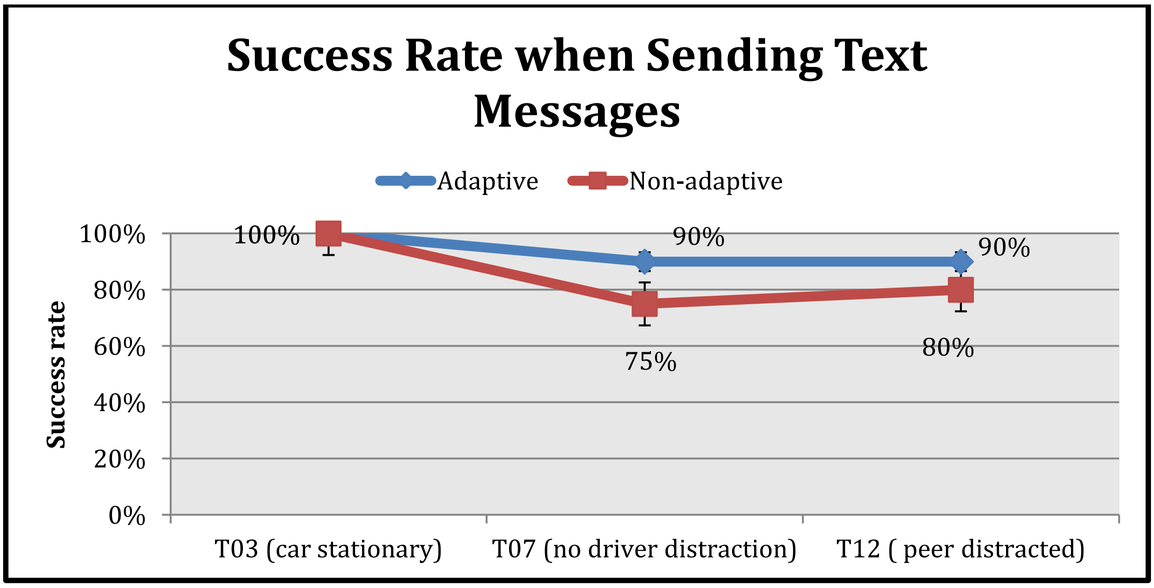


Figure 7.20: Success Rate for Sending a Text Message (n=20)

Figure 7.20 summarises the results obtained for the tasks in the category of sending a text message using the adaptive version and the non-adaptive version. While performing T03, the car was stationary; the success rate was hundred per cent when using both versions. Despite a small difference in the duration of the tasks (Figure 7.17) all participants managed to make the call successfully.

T07 consisted of sending a text message to a peer while driving; however, neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 0.08) between the adaptive version and the non-adaptive version. The average success rate was 90% when using the adaptive version and 75% when using the non-adaptive version.

T12 consisted of sending a text message while the driving conditions were unsafe for the peer. A t-test conducted on this task showed no significant difference (t (19), p = 0.16) between the adaptive version and the non-adaptive version, participants managed to perform marginally better when using the adaptive version (90%), as compared to using the non-adaptive version (80%).

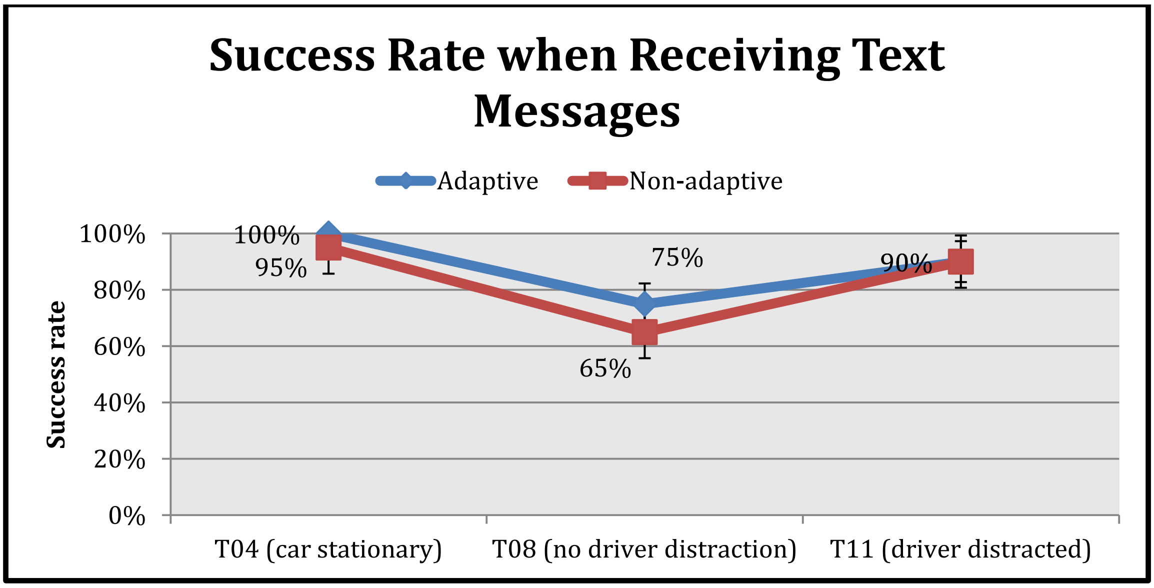


Figure 7.21: Success Rate for Receiving a Text Message (n=20)

Figure 7.21 summarises the success rate achieved for the tasks in the category of receiving a text message using the adaptive version and the non-adaptive version. While performing T04, the car was stationary; the success rate was 100% for the adaptive version and 95% for the non-adaptive version.

T08 consisted of receiving a text message from a peer while driving; however, neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 0.08) between the adaptive version and the non-adaptive version. The average of success rate was 75% when using the adaptive version and 65% when using the non-adaptive version.

T11 consisted of receiving a text message while the driving conditions were unsafe for the driver. Participants managed to achieve 90% success rate when using both versions.

7.5.1.3 Speech Recognition Errors

Errors reported in this section are speech recognition errors. Despite the speech training session, which was performed before the actual field study, participants experienced some speech recognition problems during the test.

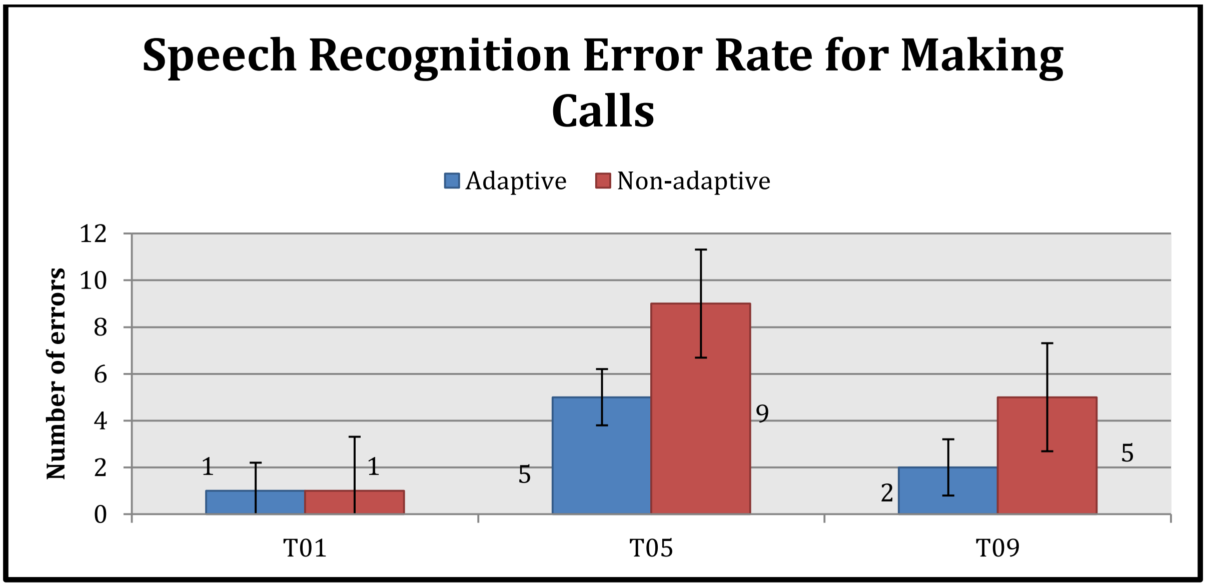


Figure 7.22: Error Rate for Making a Call (n=20)

Figure 7.22 summarises the error rate obtained for the tasks in the category of making a call using the adaptive version and the non-adaptive version. While performing T01, the car was stationary; on average, one speech recognition error was made by participants using both versions.

T05 consisted of making a call to a peer while driving; however, neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 0.51) between the adaptive version and the non-adaptive version. The average number of speech recognition errors was five when using the adaptive version and nine when using the non-adaptive version. This can be explained by the fact that, using the adaptive version, participants interacted with MIMIC when the driving situation was safe. In most cases, the speed was low and there was not much noise from the friction between the wheels and the road. Hence the response of the driver had more chance of being heard by the speech recognition engine.

T09 consisted of making a call while the driving conditions were unsafe for the driver. Participants managed to perform better when using the adaptive version. An average of only two errors was made as compared to an average of five errors when using the non-adaptive version. No significant differences were found after conducting a t-test on this task (t (19), p = 0.33).

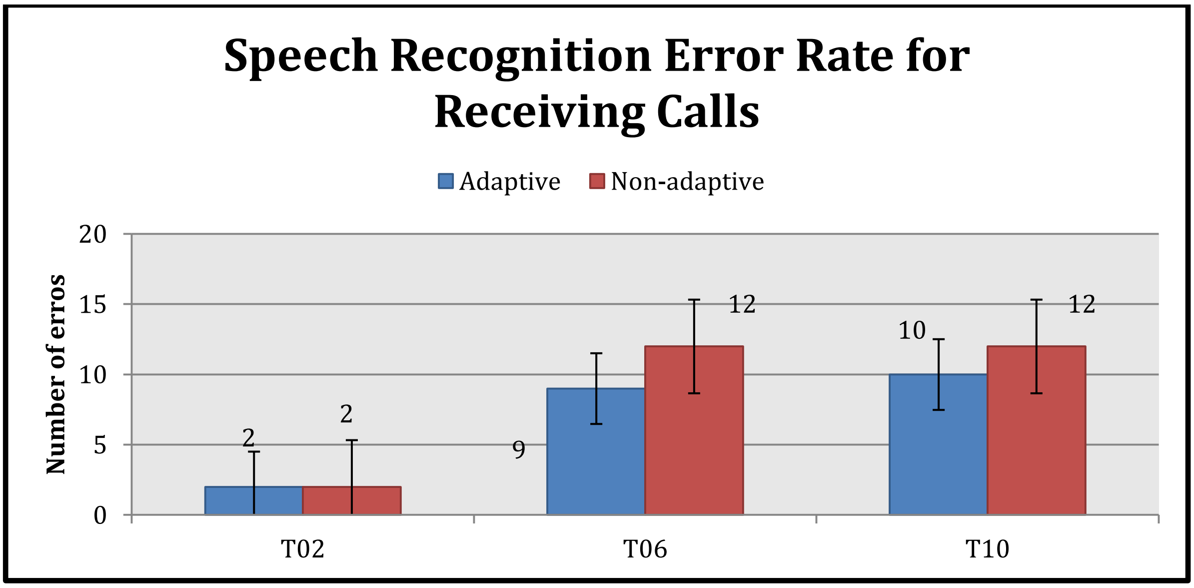


Figure 7.23: Error Rate for Receiving a Call (n=20)

Figure 7.23 summarises the error rate obtained for the tasks in the category of receiving a call using the adaptive version and the non-adaptive version. While performing T02, the car was stationary; on average, two errors were made by participants using both versions.

T06 consisted of receiving a call from a peer while driving; however, neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 0.70) between the adaptive version and the non-adaptive version. The average number of speech recognition errors was nine when using the adaptive version and 12 when using the non-adaptive version.

T10 consisted of receiving a call while the driving conditions were unsafe for the driver. Participants managed to perform better when using the adaptive version. An average of only ten speech recognition errors was made compared to an average of 12 errors when using the non-adaptive version. No significant differences were found after conducting a t-test on this task (t (19), p = 0.48).

Figure 7.24 summarises the error rate obtained for the tasks in the category of sending a text message using the adaptive version and the non-adaptive version. While performing T03, the car was stationary. On average, one error was made when using the adaptive version and two errors were made when using the non-adaptive version.

T07 consisted of sending a text message to a peer while driving; however, neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 1.00) between the adaptive version and the non-adaptive version. The average number of speech recognition errors was four when using the adaptive version and six when using the non-adaptive version.

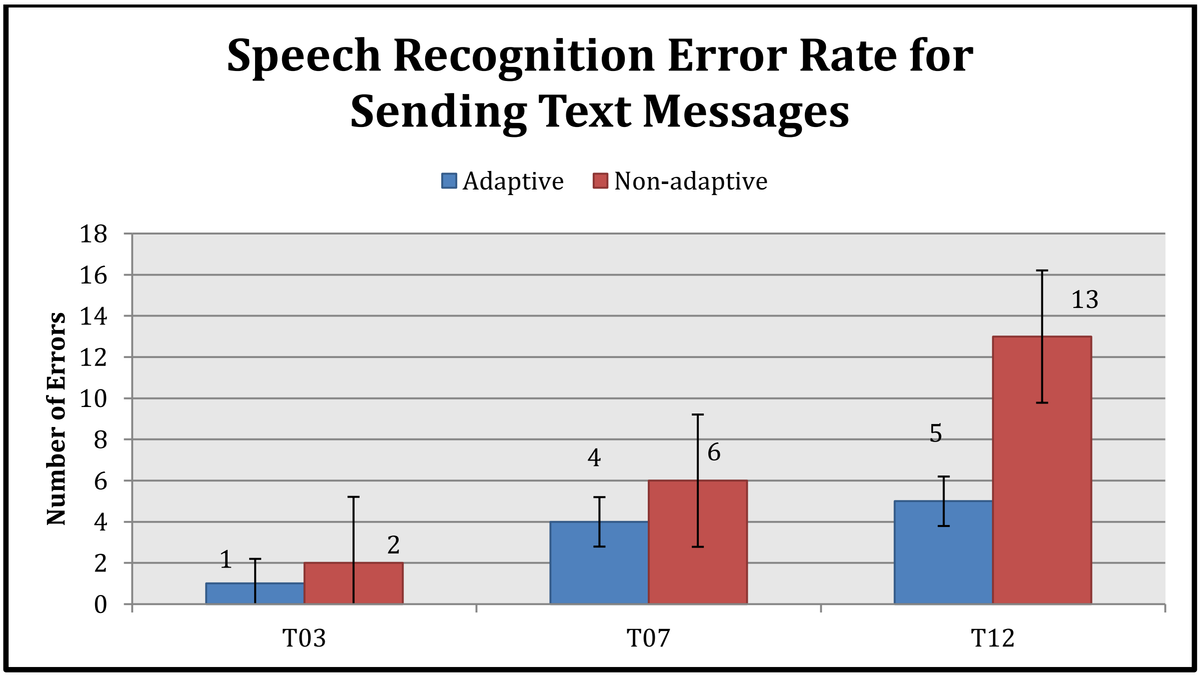


Figure 7.24: Error Rate for Sending a Text Message (n=20)

T12 consisted of sending a text message while the driving conditions were unsafe for the peer. Participants managed to perform better when using the adaptive version. An average of only five errors was made compared to an average of thirteen speech recognition errors when using the non-adaptive version. No significant difference was found after conducting a t-test on this task (t (19), p = 0.11).

Figure 7.25 summarises the error rate obtained for the tasks in the category of receiving a text message using the adaptive version and the non-adaptive version. While performing T04, the car was stationary. On average, one error was made when using the adaptive version and no errors were made when using the non-adaptive version.

T08 consisted of receiving a text message from a peer while driving; however, neither the driver nor the peer was distracted while performing this task. A t-test conducted on this task showed no significant difference (t (19), p = 0.61) between the adaptive version and the non-adaptive version. The average number of errors was seven when using the adaptive version and eight when using the non-adaptive version.

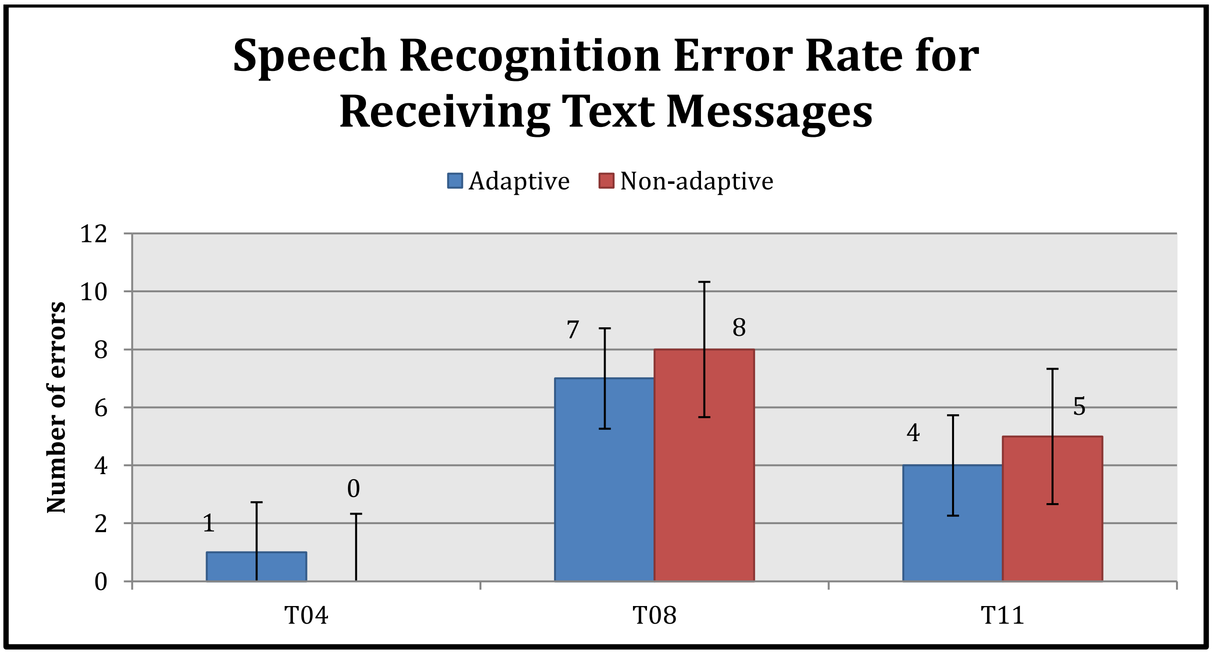


Figure 7.25: Error Rate for Receiving a Text Message (n=20)

T11 consisted of receiving a text message while the driving conditions were unsafe for the driver. Participants managed to perform slightly better when using the adaptive version. On average, only four errors were made compared to an average of five errors when using the non-adaptive version. No significant difference was found after conducting a t-test on this task (t (19), p = 0.77).

7.5.2. Perceived Workload Results

This section discusses workload results collected using the NASA TLX questionnaire. This questionnaire captures the following variables: mental workload, physical workload, temporal workload, perceived performance, effort and frustration.

The NASA TLX scores are given on a 5-point Likert scale. Except for the performance variable, all other variables below three (3.00) indicate a positive rating. The following paragraphs provide details on each variable.

As depicted in Figure 7.26, the mental demand when using the adaptive version (mean = 1.93, median = 2.00 and standard deviation = 0.81) and the non-adaptive version (mean = 2.07, median = 2.00 and standard deviation = 0.85) was well below 3.00. The adaptive version was perceived as slightly less mentally demanding than the non-adaptive version.

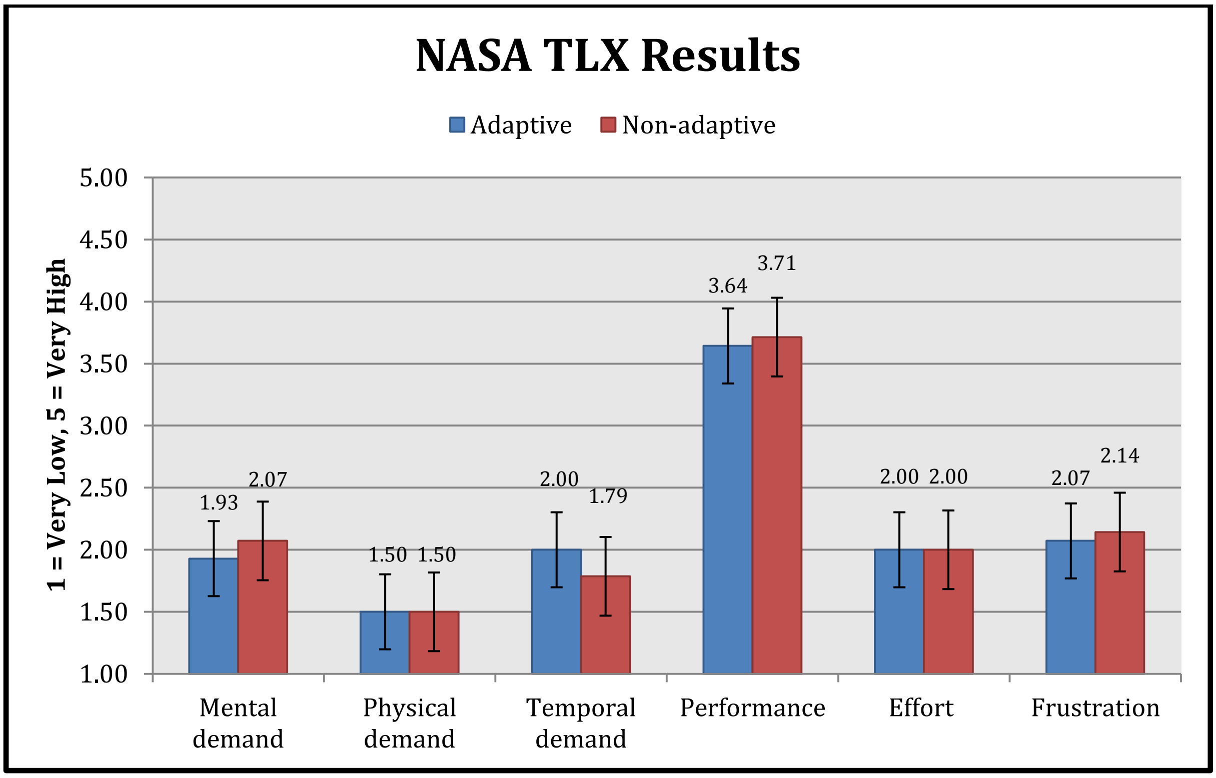


Figure 7.26: NASA TLX Results (n=20)

The physical demand required to perform the tasks was similar in the adaptive version (mean = 1.50, median = 1.00 and standard deviation = 0.77) and the non-adaptive version (mean = 1.50, median = 1.00 and standard deviation = 0.77). Participants did not find the tasks physically demanding despite the fact that some participants had to lean towards the mobile phone to speak close to the microphone.

The temporal demand was also well below 3.00. The non-adaptive version (mean = 2.00, median = 2.00 and standard deviation = 0.85) performed slightly better than the adaptive version (mean = 1.79, median = 1.00 and standard deviation = 0.83).

The self-reported performance, which is a subjective rating of the effectiveness of the system, was found to be higher than the threshold value (3.00), which means that it was positive. The adaptive version (mean = 3.64, median = 4.00 and standard deviation = 0.92) performed slightly worse than the non-adaptive version (mean = 3.71, median = 4.00 and standard deviation = 0.83).

Little effort was needed to perform the tasks. The adaptive version (mean = 2.00, median = 2.00 and standard deviation = 0.83) was the same, on average, as the non-adaptive version (mean = 2.00, median = 2.00 and standard deviation = 0.85).

Little frustration was caused while performing the tasks, even though some recognition errors seemed to frustrate participants. The adaptive version (mean = 2.00, median = 2.07 and standard deviation = 0.83) performed slightly worse than the non-adaptive version (mean = 2.14, median = 2.00 and standard deviation = 0.85).

7.5.3. User Satisfaction Results

The SUS questionnaire (Appendix C) was used to capture user satisfaction data after performing the tasks using the two versions. This questionnaire alternates positive and negative statements, but the scores were normalised to use the same scales.

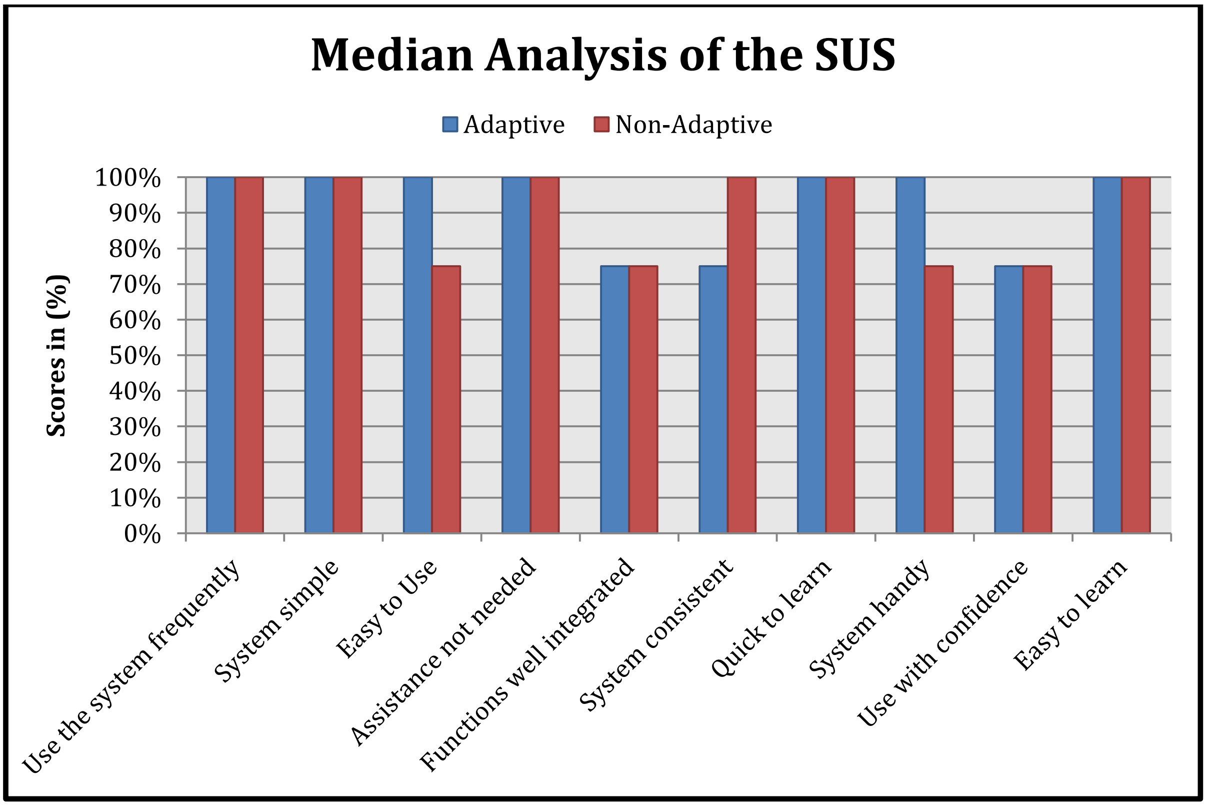


Figure 7.27: Median Analysis of the SUS (n=20)

Figure 7.27 summarises the usability results obtained. The following usability variables obtained an excellent score (100%) for both versions of the system: willingness to use the system frequently, system not unnecessarily complex, no assistance needed to use the system, the system can be learnt quickly and there were not many things to learn in order to use the system. This result was expected because the same design was used to develop both versions of MIMIC-Prototype. Seventy-five per cent of participants used both versions with confidence. This was also the case for the integration of the various functions of the system. The adaptive version of MIMIC-Prototype (100%) had a slight advantage over the non-adaptive version (75%) in terms of ease of use. Most participants found the non-adaptive version (75%) more cumbersome than the adaptive version (100%). The non-adaptive version (100%) had a slight advantage over the adaptive version (75%) in terms of the system consistency.

A t-test was conducted to compare each of the usability variables. No significant differences were found. This was expected because the interaction between participants and both versions was very similar.

Table 7.4 summarises the mean SUS scores for the adaptive and the non-adaptive versions of MIMIC-Prototype. It provides a general overview of the usability of both versions of MIMIC-Prototype. The mean scores of both versions were very high: 93% for the adaptive version and 90% for the non-adaptive version. This confirmed that the usability of both systems was satisfactory with a slight advantage for the adaptive version.

	Adaptive	**Non-Adaptive**
SUS score	93%	90%

Table 7.4: Mean SUS Score (n=20)

7.5.4. Self-reported Driver Distraction

Participants were asked how distracted they felt when they performed the tasks during the field study. Four variables captured the self-reported driver distraction for driver-initiated and peer-initiated tasks. These tasks belong to the following categories: making a call, receiving a call, sending and receiving a text message.

	Adaptive Version			**Non-Adaptive Version**			**t-test**
	Mean	Median	StdDev	Mean	Median	StdDev	p-value
Distracted making calls	2.05	2.00	1.08	2.11	2.00	0.88	0.84
Distracted sending Text messages	1.95	2.00	0.71	2.21	2.00	0.92	0.10
Distracted receiving Text messages	**1.95**	**2.00**	**0.85**	**2.42**	**2.00**	**1.17**	**0.05**
Distracted receiving calls	**1.7**	**2.00**	**0.98**	**2.74**	**3.00**	**1.33**	**0.01**

Table 7.5: Perceived Distraction while Performing Tasks (n=20)

A 5-point semantic differential scale, ranging from 1 (low distraction) to 5 (high distraction) was used to capture the data. A t-test analysis with a 95% confidence interval was conducted in order to compare the adaptive version and the non-adaptive version.

The results summarised in Table 7.5 showed that participants experienced little distraction making calls using the adaptive version. The t-test analysis (t (19), p = 0.84) did not find a significant difference between the adaptive version (mean = 2.05, median = 2.00, and standard deviation = 1.08) and the non-adaptive version (mean = 2.11, median = 2.00, and standard deviation = 0.88). This means that even the non-adaptive version was not distracting while making calls.

Participants also felt less distracted when sending text messages with the adaptive version. The t-test analysis (t (19), p = 0.10) did not find a significant difference between the adaptive version (mean = 1.95, median = 2.00 and standard deviation = 0.71) and the non-adaptive version (mean = 2.21, median = 2.00 and standard deviation = 0.92).

Participants were asked if they felt distracted when receiving text messages. A t-test analysis (t (19), p = 0.05) found a significant difference between the adaptive version (mean = 1.95, median = 2.00 and standard deviation = 0.85) and the non-adaptive version (mean = 2.42, median = 2.00 and standard deviation = 1.17). This means that participants did not experience driver distraction while receiving calls with the adaptive version. A mean rating of 2.42 was given for the non-adaptive version.

Participants were asked if they felt distracted when receiving calls. A t-test analysis (t (19), p = 0.01) found a significant difference between the adaptive version (mean = 1.70, median = 2.00 and standard deviation = 0.98) and the non-adaptive version (mean = 2.74, median = 3.00 and standard deviation = 1.33). This means that participants did not experience driver distraction while receiving text messages with the adaptive version.

7.5.5. Distraction Level

The distraction level provided by the Inference Engine was collected for each task. The distraction level was recorded when the participants started a dialogue with the system. The mean distraction level for each tack was calculated for all participants. Figure 7.28 summarises the results obtained for making a call using the adaptive version and the non-adaptive version. While performing T01, the car was stationary; the distraction level (mean = 1.00) was “Very Low” for both versions.

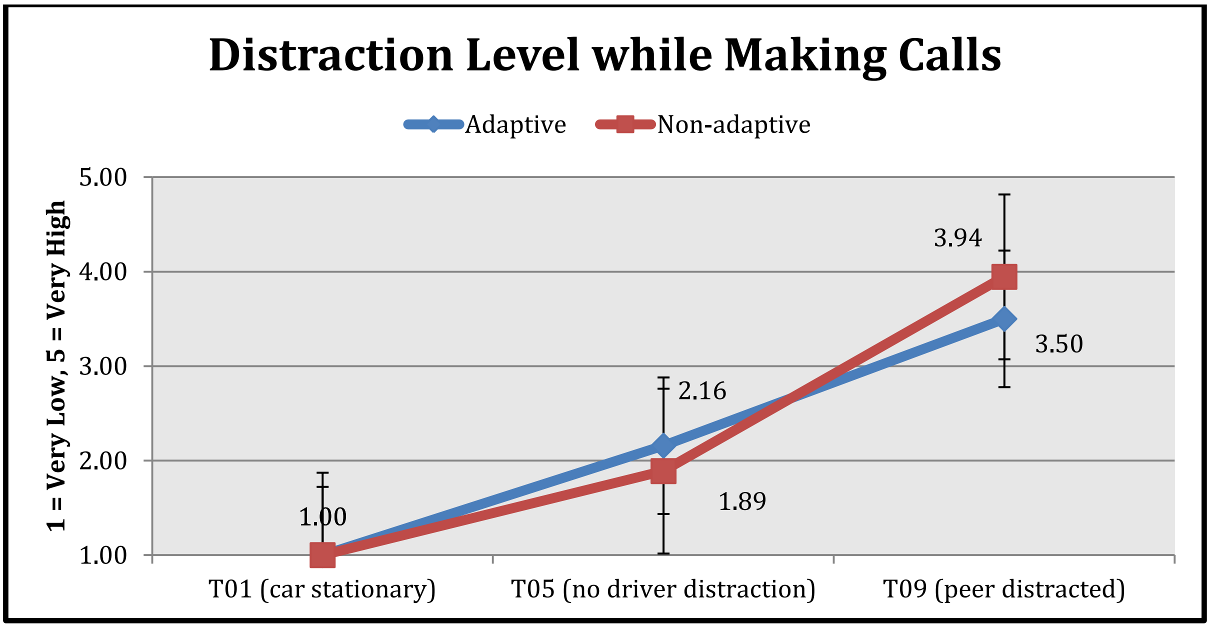


Figure 7.28: Mean Distraction Levels for Making Calls (n=20)

The mean distraction level was "Low" when performing T05, which occurred when the car was moving, but the driver and the callee were in a safe situation. Participants were slightly less distracted when using the non-adaptive version (mean = 1.89) as compared to the adaptive version (mean = 2.16). A t-test did not find any significant difference between both versions. The same results were found for T09, but the distraction level was higher because the task occurred when it was unsafe for the callee to receive a call. Both versions had a distraction level between "Medium" (3.00) and "High" (5.00) for T09.

Figure 7.29 summarises the results obtained when receiving a call using both versions of the system. The distraction level was "Very Low" (mean = 1.00) when receiving a call in the stationary car. The distraction level was still below "Medium" (mean = 3.00) for both versions when a call was received in a moving car (T06). A significant difference was found when performing T10; when participants had to receive a call in an unsafe situation. The adaptive version performed better (mean = 1.45) than the non-adaptive version (mean = 4.56). This can be explained by the fact that when using the adaptive version, the call notification was delayed until the driving situation returned to safe.

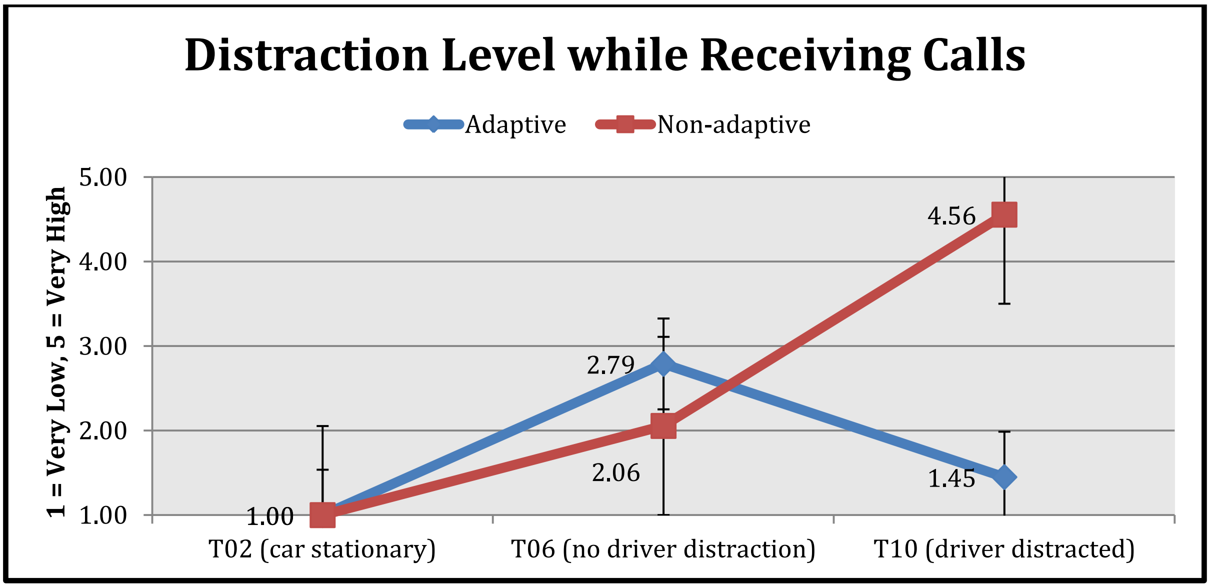


Figure 7.29: Mean of Distraction Level for Receiving Calls (n=20)

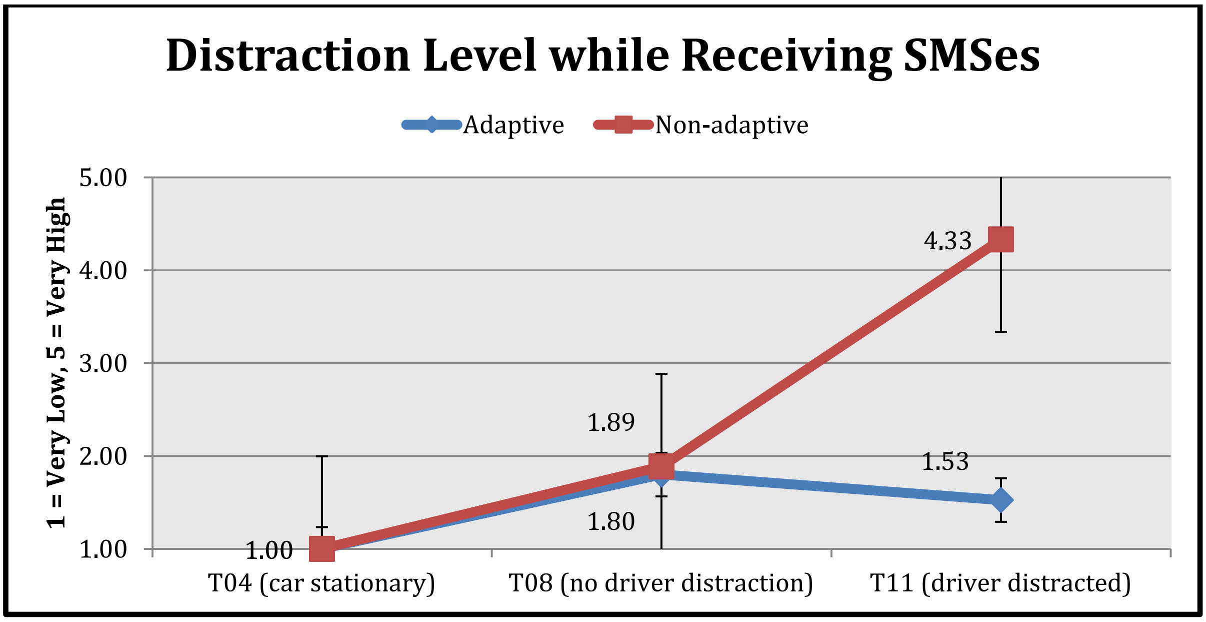


Figure 7.30: Mean of Distraction Level for Receiving Text Messages (n=20)

Figure 7.30 summarises the results obtained when receiving a text message using both versions of MIMIC-Prototype. The distraction level was “Very Low” (mean = 1.00) when receiving a call in the stationary car.

The distraction level was still below “Medium” (mean = 3.00) for both versions when a call was received in a moving car (T08). A significant difference was found when performing T11; participants had to receive a text message when it was unsafe to do so. The adaptive version performed better (mean = 1.53) than the non-adaptive version (mean = 4.33). This can be explained by the fact that when using the adaptive version, the call was delayed until the driving situation returned to safe.

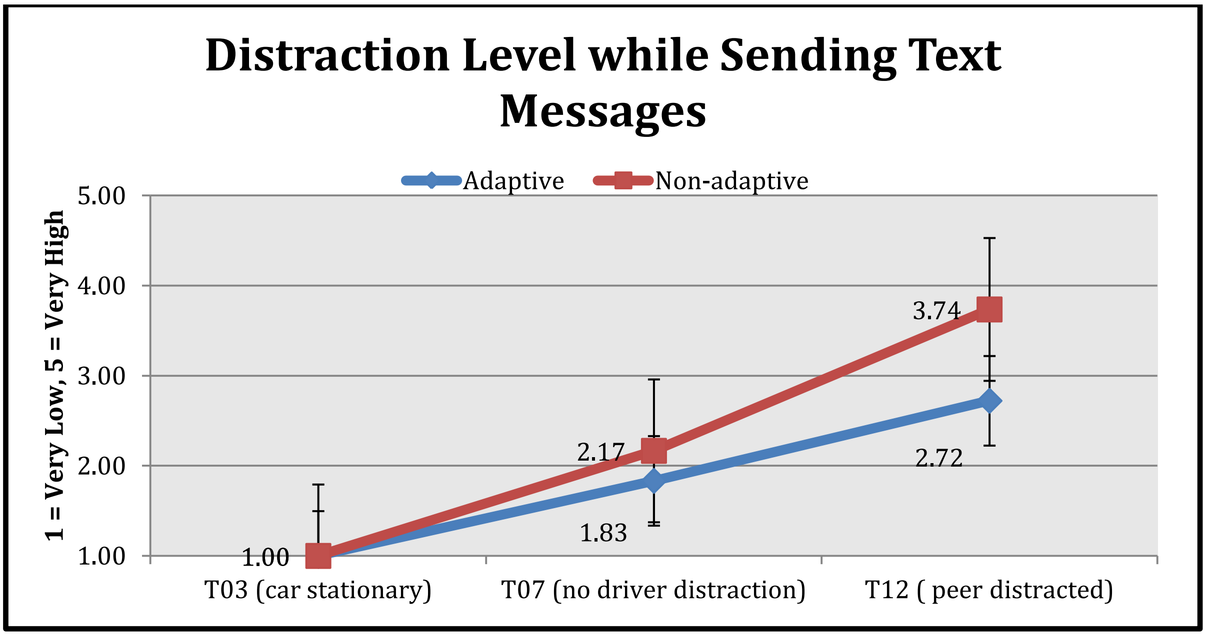


Figure 7.31: Mean of Distraction Level for Sending Text Messages (n=20)

Figure 7.31 summarises the results obtained for sending a text message using the adaptive version and the non-adaptive version of MIMIC-Prototype. While performing T03, the car was stationary; the distraction level (mean = 1.00) was equally “Very Low” for both versions. The mean distraction level was also “Low” when performing T07, which occurred when the car was moving, but the driver and the callee were in a safe situation. Participants were slightly less distracted when using the adaptive version (mean = 1.83), as compared to using the non-adaptive version (mean = 2.17). A t-test did not find any significant difference between both versions. The same results were found for T12, but the distraction level was higher because the task occurred when it was unsafe for the peer to receive a text message.

The adaptive version of MIMIC-Prototype contributed to reducing the distraction level. This was significant for peer-initiated tasks (incoming calls or text messages). MIMIC-Prototype only notified the driver when the driving situation was safe.

When considering all tasks (Figure 7.32) independently of the versions used, the distraction level measure by the MIMIC Inference Engine was generally lower than “Medium” (3.00). A t-test conducted on all tasks showed no significant difference (t (19), p = 0.13), but the adaptive version (mean = 1.81) was less distractive than the non-adaptive version (mean = 2.38).

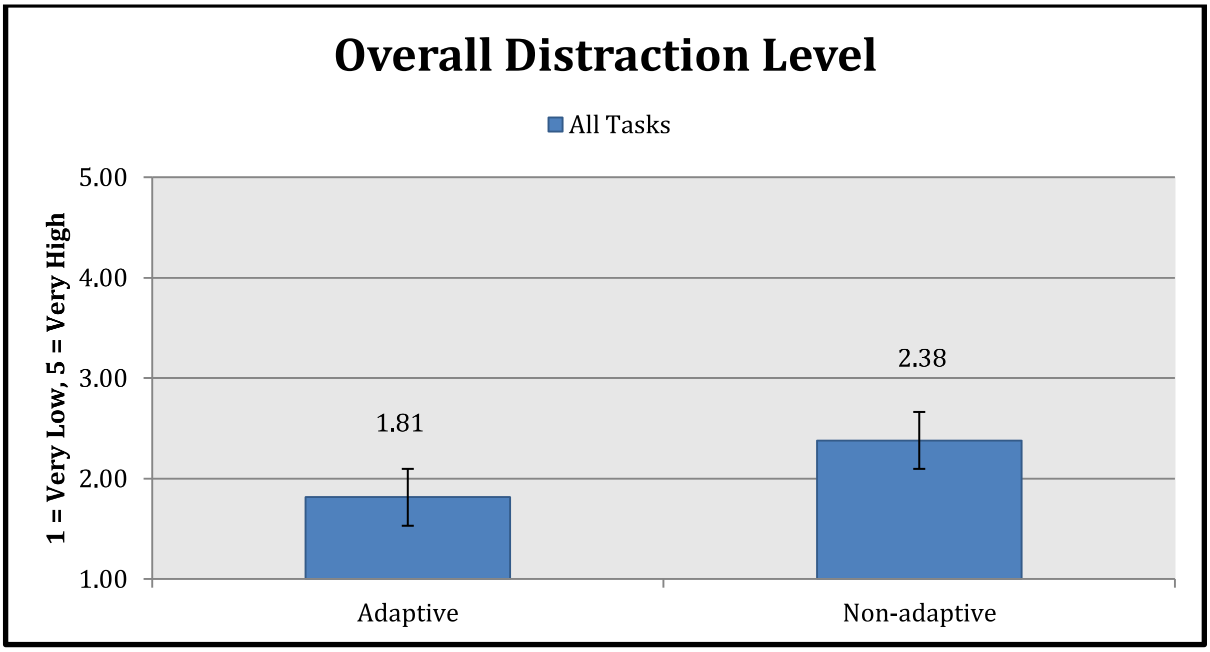


Figure 7.32: Overall Distraction Level for all Tasks (n=20)

7.5.6. Adaptability and Context-Awareness

The adaptive version used the Context-aware Module discussed in Chapter 5. Participants were asked to rate how they felt about the adaptation of the system with regard to the driving context. The ratings ranged from 1 to 5, where participants who gave a rating of 1 strongly disagreed that any adaptation took place, while participants who gave a rating of 5 strongly agreed that they noticed an adaptation mechanism while performing some tasks. The adaptation occurred mostly after peer-initiated tasks (incoming calls and text messages). The system also notified the driver when the peer could be distracted by a call or a text message.

	Adaptive Version			Non-Adaptive Version			T-test
	Mean	Median	Std Dev	Mean	Median	Std Dev	p-value
Adaptive when receiving calls	4.26	4.00	0.65	3.58	4.00	1.02	0.01
Adaptive when receiving Text messages	4.26	4.00	0.45	3.89	4.00	0.74	0.05
Aware of Peer context	4.00	4.00	0.97	3.74	4.00	0.65	0.02

Table 7.6: Noticeability of Adaptation Mechanisms (n=20)

The results summarised in Table 7.6 showed that participants noticed the fact that the adaptive version took their context into consideration. The following variables were investigated: adaptive when receiving calls, adaptive when receiving text messages and awareness of the peer context.

Regarding the noticeability of context adaptation when receiving calls, a t-test analysis (t (19), $p = 0.01$) found a significant difference between the adaptive version (mean = 4.26, median = 4.00 and standard deviation = 0.65) and the non-adaptive version (mean = 3.58, median = 4.00 and standard deviation = 1.02). This result was expected because the dialogue in the adaptive version was different from the dialogue in the non-adaptive version.

Regarding the noticeability of context adaptation when receiving text messages, a t-test analysis (t (19), $p = 0.05$) found a significant difference between the adaptive version (mean = 4.26, median = 4.00 and standard deviation = 0.45) and the non-adaptive version (mean = 3.89, median = 4.00 and standard deviation = 1.74). This result was also expected because the dialogue in the adaptive version was different from the dialogue in the non-adaptive version.

Finally, participants were more aware of the peer context when using the adaptive version. A t-test analysis (t (19), $p = 0.02$) found a significant difference between the adaptive version (mean = 4.00, median = 4.00 and standard deviation = 0.97) and the non-adaptive version (mean = 3.74, median = 4.00 and standard deviation = 0.65). When the driving context of the peer was found to be unsafe, the dialogue in the adaptive version was different from the dialogue in the non-adaptive version.

7.5.7. Effectiveness in Determining Safe Driving situations and in Implementing Adaptation Effects

The driving context was logged and the result of the safety assessment was also logged. Four tasks occurred when either the driver or the peer were in an unsafe driving situation (Table 7.2). This section discusses the effectiveness of MIMIC in determining whether a driving situation was safe or not. When performing the four tasks (T09, T10, T11 and T12) MIMIC always determined the driving situation as unsafe. This can be explained by the fact that these tasks occurred when participants were turning.

Two tasks occurred when the driver was in an unsafe driving situation (Table 7.2). This required an adaptation mechanism to reduce the driver distraction. This section discusses the effectiveness of MIMIC in implementing the correct adaptation effect. It was found that the expected adaptation effect occurred for all tasks that were completed successfully. The cause of errors was mostly poor speech recognition in certain locations.

7.5.8. Post-Trip Comments

Participants were asked three open-ended questions at the end of each trip. The first question asked them to give the most positive aspect about the system. The second question, asked them to give the most negative aspect about the system and the third question asked participants to provide general comments and suggestions for improvement.

Positive aspects of the adaptive version and the non-adaptive version were almost similar. Both systems were easy to use, simple and helpful. The most negative aspects were similar as well; voice recognition was the cause of most of the problems occurring when using both systems. Participants complained about having to speak clearly and loudly to improve the chances of having their commands recognised.

Suggestions for improvement related to improving the voice recognition. Some participants suggested implementing different beep sounds for positive and negative feedback. This could help the user to know what the system expected. It was also suggested to allow the user to change the text message selected in case a mistake was made. With the current design, the user must cancel the command and start again from scratch.

7.5.9. Post-Test Preference Ratings

Participants were asked to give a score using a 5-point semantic differential scale (Appendix D), ranging from 1 (strongly prefer the adaptive version) to 5 (strongly prefer the non-adaptive version).

Figure 7.33 summarises the responses given by the participants. When asked which system they preferred to use when receiving calls, they clearly chose the adaptive version (mean = 1.67). Participants felt that the adaptive version (mean = 1.56) was also better than the non-adaptive version for receiving text messages. Participants preferred the adaptive version for making calls (mean = 1.67) and sending text messages (mean = 1.50). Overall, when taking all tasks into consideration, participants preferred used the adaptive version (mean = 1.60) over the non-adaptive version.

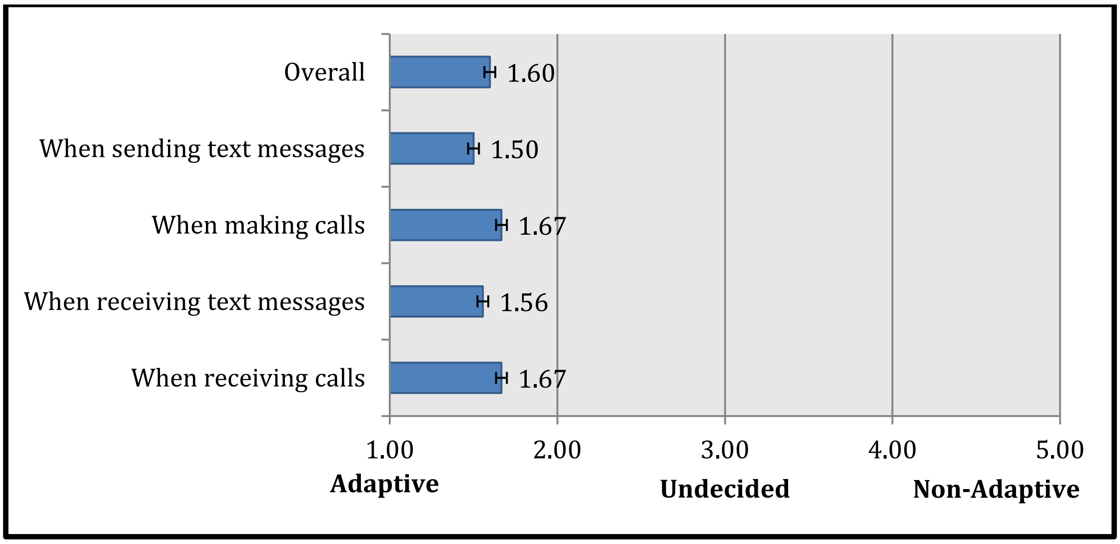


Figure 7.33: Preference for the Adaptive Version (n = 20)

The adaptive version was also preferred when it came to making calls and sending text messages. The difference between the adaptive version and the non-adaptive version was more pronounced on incoming tasks (receiving text messages and calls). This can be explained by the fact that the design of the adaptive version was clearly different from that of the non-adaptive version when the driving context was unsafe. The notification of the incoming events was postponed and participants felt more comfortable with that.

7.5.10. Post-Test Comments

Participants were asked four open-ended questions at the end of the test. These questions were about identifying the most positive aspect of the adaptive version, the most negative aspect of the adaptive version, the most positive aspect of the non-adaptive version and the most negative aspect of the non-adaptive version.

#	**Top 3 positive aspects of the adaptive version**	%
1	The system was easy to use	36
2	Felt more comfortable to use	28
3	Took context into consideration	26

Table 7.7: Positive Aspects of the Adaptive Version

Table 7.7 lists the three most common positive aspects of the adaptive version. Most participants found the system easy to use. The second most common positive aspect was that the system was comfortable to use. The third most common aspect was that the system took the driving context of the participants into consideration.

Table 7.8 lists the three most common negative aspects of the adaptive version. Most participants found that the system did not recognise their voice effectively. The second negative aspect was related to the first one, but was more specific. The recognition of the confirmation command “Yes” was poorly recognised. Participants had to repeat it several times before successful recognition was achieved. The third most common negative aspect was also related to speech recognition.

#	**Top 3 negative aspects of the adaptive version**	%
1	Voice recognition not always working	53
2	Bad recognition of “Yes”	22
3	Connection issues	9

Table 7.8: Negative Aspects of the Adaptive Version

The connectivity issue was pointed out because it was explained how the speech recognition is done by mobile phones. Participants knew that the poor recognition rate could be attributed to non-reliable Internet connectivity while moving.

#	**Top 3 positive aspects of the non-adaptive version**	%
1	The system was easy to use	62
2	Hands-free	14
3	Clear instructions were given	11

Table 7.9: Positive Aspect of the Non-Adaptive Version

Table 7.9 lists the three most common positive aspects of the non-adaptive version. Most participants found the system as easy to use as the adaptive version. The second positive aspect was that the system was helpful because it is hands-free, so the driver can drive while using the system. The third most common positive aspect was that the system gave clear instructions to the user, which made it easy to use and intuitive.

Table 7.10 lists the three most common negative aspects of the non-adaptive version. Most participants experienced issues with voice recognition. The second most common negative aspect was that the system allowed the driver to take calls while in a difficult driving situation. The third most common negative aspect also related to voice recognition issues caused by external and engine noises.

#	**Top 3 negative aspects of the non-adaptive version**	%
1	Voice recognition issues	65
2	Allowed me to take a call while distracted	21
3	External noise affects the system	11

Table 7.10: Negative Aspect of the Non-Adaptive Version

7.6. Discussion

The field study went well without any incident. The test moderator did not observe any apparent visual or manual distraction. The main issue that was highlighted from the open-ended questions was speech recognition for both versions. The tasks (T01, T02, T03 and T04) that were completed while the car was stationary did not suffer from poor speech recognition. The number of errors increased for all tasks completed while the car was moving. Despite the implementation of the Android native noise cancellation mechanism, short utterances such as, “Yes”, “No” or numbers for text message options were difficult to distinguish from the noise of the engine. This is why participants had to repeat these words several times before there was successful recognition by the speech engine. Most participants found this annoying.

The workload results obtained from the NASA TLX were generally satisfying. The frustration was fairly low despite the fact that speech recognition errors tended to annoy most participants. This can be explained by the fact that, according to the biographical details (Section 7.4.1); seventy percent of participants had never used an ICCS prior to this field study. They seemed amazed by the fact that most commands were recognised successfully and were therefore lenient when reporting on variables such as the frustration, as well as the physical and mental workload. There were no significant differences in terms of workload between the two versions of the MIMIC-Prototype. The performance high score was corroborated by the success rate, which was high for the majority of tasks. Some participants waited until they were stopped before providing a response to the system. The MIMIC-Prototype gave three periods of twenty seconds each in order to respond to the system. This was found sufficient for most participants to complete a task successfully. It could also explain the fact that some tasks took a lot of time. The fact that most participants (95%) had the feeling that a mobile ICCS can help in reducing driver distraction (Figure 7.9), might have influenced their positive post-test ratings of the system.

The usability rated with the SUS questionnaire was also generally high for both versions. Neither of the versions was significantly different from the other. It was reported that the system was simple, easy to use and that its different functions were well integrated. However, some suggestions were made in order to improve the feedback mechanism by using different "beep" sounds for negative and positive feedback. It was also suggested to improve the text message sending dialogue to enable the driver to correct a mistake without having to restart the dialogue from scratch.

The adaptation to the current driving context was noticed by the participants. A t-test analysis showed that the adaptive version was significantly better than the non-adaptive version in terms of taking the driving context into consideration. This was one of the most positive aspects of the adaptive version. Users liked the fact that the system only interacted with them when it was safe to do so.

The research question to be answered in this chapter was to determine whether the adaptive version reduced driver distraction (RQ9). The answer can be found in Section 7.5.4. The adaptive version was significantly better than the non-adaptive version in terms of causing the lowest perceived distraction when performing tasks. It is noteworthy that this result was confirmed by the actual distraction level as determined by the Inference Engine (Section 7.5.5). The distraction level when performing peer-initiated tasks was significantly lower when using the adaptive version of MIMIC-Prototype in unsafe driving situations.

7.7. Conclusion

This chapter described the field study that was conducted in order to measure to what extent MIMIC-Prototype reduces driver distraction. The study used a within-subjects approach to compare the adaptive version to the non-adaptive version. The two versions were compared in terms of performance, workload, driver distraction, adaptability and usability.

The adaptive version of the prototype was built based on the Inference Engine described in Chapter 5 and the Context Adapter discussed in Chapter 6. The experiment took place in the field using a real car and a mobile phone instead of a driving simulator.

The results obtained showed that the usability of both versions was good. It was easy to learn how to use both versions of the prototype. The workload caused by the prototype is important because it is used while multi-tasking. Satisfactory results were found for both versions regarding the workload. It was interesting to determine that, despite not being briefed about the adaptation effects, participants noticed the various adaptation mechanisms provided by the adaptive version. The adaptive version was significantly better than the non-adaptive version in terms of perceived driver distraction as well as the actual distraction level determined by the Inference Engine (Section 7.5.5). This provides a positive answer to the research question that was investigated in this chapter.

The next chapter will review the research objectives of this thesis and highlight the research achievements. Several suggestions for future research will also be given.

Chapter 8: Conclusions

8.1. Introduction

This chapter will present some conclusions which can be drawn from this research. Firstly, the results of this research will be summarised, highlighting the most noteworthy findings. Secondly, the contribution of this research will be summarised, highlighting theoretical and practical contributions. Limitations and problems encountered will also be discussed briefly. Finally, recommendations and opportunities for future work stemming from this research will be identified.

8.2. Summary of Findings

This research was based on the following thesis statement: *a mobile, context-aware model can be designed to reduce driver distraction caused by the use of in-car communication systems (ICCSs)*. In order to demonstrate this statement several research objectives were formulated to help in answering the research questions (Table 1.1). These objectives included the following:

- To identify the causes and effects of driver distraction (Chapter 2),
- To define and review existing models for context-aware applications (Chapter 3),
- To design a model for a speech-based, mobile ICCS (Chapter 4),
- To investigate the feasibility of a speech-based, mobile ICCS (Chapter 4),
- To design an inference engine that can be used to determine the driving context (Chapter 5),
- To select a machine learning technique that can determine the driving context (Chapter 5),

- To design a context adapter that can be used to reduce driver distraction (Chapter 6),

- To identify and implement adaptation effects that can help in reducing driver distraction (Chapter 6),

- To measure the extent to which the proposed model for a mobile, context-aware ICCS can help in reducing driver distraction (Chapter 7).

Causes and Effects of Driver Distraction (Chapter 2)

Objective**: To identify the causes and effects of driver distraction**

Output:

- Internal causes of driver distraction, External causes of driver distraction,
- Effects: decreasing driver attention, dangerous driving behaviour (speed, following distance), increasing risk of car accident.

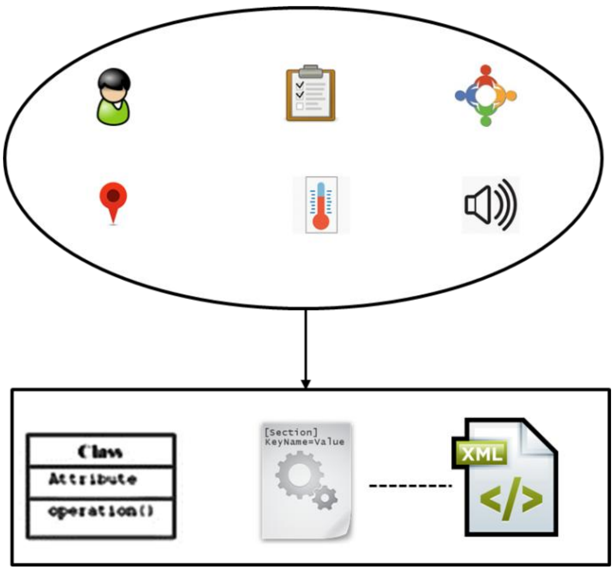
Figure 8.1: Summary of Chapter 2

Chapter 2 identified the causes and effects of driver distraction (Figure 8.1). Driver distraction occurs when the attention of the driver is diverted from the primary task, which is driving. This can be caused by internal and external events. Driver distraction is complex as it has many types, namely: visual, manual, auditory and cognitive distractions. Several efforts have been made to address manual and visual driver distraction, however, cognitive distraction remained difficult to address. This can be explained by the psychological aspects of driver distraction. The multiple resource theory is a four-dimensional model (modalities, processing code, stages and responses) that attempts to provide an explanation why driver distraction occurs when dealing with secondary tasks. It is believed that tasks that share the same pool of resources (visual, auditory, motor and cognitive) are likely to cause distraction. The workload also plays an important role in causing driver distraction. The workload tends to increase when the driver attention is divided. Awareness of the driver situation contributes towards minimising the workload. Amongst many causes of car accidents in the United States of America (USA), driver distraction is to blame for 18% of crashes. Sixteen per cent of drivers under the age of 20 who were involved in a car accident were reported to have been distracted.

Several solutions adopted so far were discussed, such as intelligent transportation systems, speech user interfaces and multimodal interfaces. It was shown that the design of ICCS following specific guidelines can help in reducing driver distraction. Today, several luxury vehicles are equipped with ICCS that aim to improve driver safety. The cost of such technology remains high and is therefore inaccessible for younger drivers. Many mobile application developers and car manufacturers are introducing mobile ICCS. Unfortunately, no empirical research exists on their safety and usability.

Context-Aware Applications (Chapter 3)

Objective**: To define and review existing models for context-aware applications**



Output:

- Context-Aware Applications are based on:
  - Human factors: Users, social environment, tasks,
  - Physical environment: Location, infrastructures, conditions (noise, temperature),
- Context Information Models
  - Key-value,
  - Object oriented,
  - Markup based,
  - Logic based,
  - Ontologies.

Figure 8.2: Summary of Chapter 3

Chapter 3 served to define and analyse context-aware applications (Figure 8.2). Context-aware applications are applications that adapt to the context of the user, the environment or the activity. As pervasive and ubiquitous computing is growing in popularity, context-aware applications are becoming increasingly developed for mobile users (drivers and pedestrians). Applications that are context-aware could possibly prevent drivers from engaging in secondary tasks when their full attention is required on the road.

A set of requirements were identified to successfully develop a context-aware application. The choice of the model to be used is critical for such applications. The key-value and object-oriented models were found to be suitable for a context-aware application to be run on a mobile phone. The simplicity of these models will guarantee an optimum use of resources. The accuracy of the determination of the driving context is also very important. Several techniques were reviewed, which can be used to determine the driving context. It was shown that techniques that can be trained offline are suitable options for mobile, context-aware applications. Nearest neighbour, decision tree, support vector machine, Bayesian networks, naïve Bayes and neural networks are good options for such applications (Section 3.6).

Proposed Model for a Speech-based ICCS (Chapter 4)

Objective 1**: To design a model for a speech-based mobile ICCS,**

Objective 2**: To investigate the feasibility of speech-based mobile ICCS.**

Output:

- A proposal of a model for a mobile, speech-based ICCS,
- Results of a usability evaluation of a speech-based mobile prototype:
    - Lab experiment,
    - Lane change task (LCT) Test.

Figure 8.3: Summary of Chapter 4

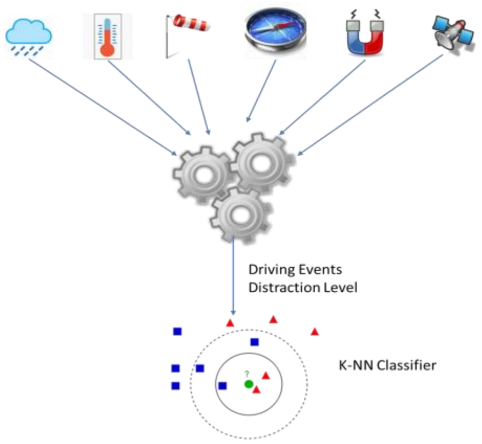
Chapter 4 proposed a model for mobile, context-aware ICCS, called MIMIC and investigated the feasibility of a speech-based, mobile ICCS. A prototype called MIMIC-Prototype was developed based on the proposed model (Section 4.2). A preliminary usability study was conducted in order to identify potential usability issues with the prototype. A low-fidelity driving simulator was set up in a laboratory using the lane change task (LCT) test software (Figure 8.3). A video projector and a board were used to simulate the windshield of the car. Participants had to perform several communication-related tasks using MIMIC-Prototype. The results showed that the system was highly effective in sending pre-recorded text messages and making calls. Overall, the usability results were encouraging (Section 4.4). The majority of the participants indicated that they were willing to use such an ICCS in future. However, the dictation of telephone numbers was a source of several errors that frustrated the participants. The system frequently failed to recognise some commands.

Figure 8.4 depicts the summary of Chapter 5, which had to answer two research questions. The first one was related to the update of MIMIC to enable the proposed model to collect sensor data from various sensors and web services available from a mobile device. In order to answer the second research question, several machine learning techniques that can determine the driving context effectively were reviewed. Two experiments were conducted to create a computing model that can predict the driving context. The driving context was divided into two variables, namely the driving event and the distraction level.

Inferring the Driving Context (Chapter 5)

Objective 1**: To design an inference engine that can be used to determine the driving context,**

Objective 2**: To select a machine learning technique that can determine the driving context.**



Output:

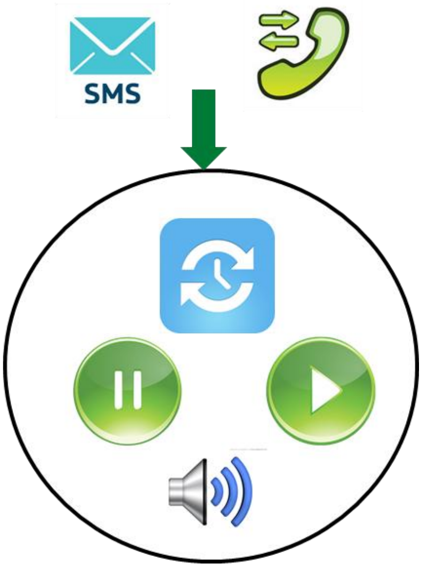
- A proposal of a model for mobile, speech-based context-aware ICCS, and
- Results of experiments and selection of algorithms for inferring the driving context (distraction level and driving events).

Figure 8.4: Summary of Chapter 5

An updated version of MIMIC-Prototype was implemented in order to help with the collection of data to be used as training and testing sets. Several machine learning algorithms were applied to determine which one provided the best accuracy. The nearest neighbour algorithms, IB1 and IB3 (Table 5.5) were found to have the highest accuracy in predicting the driving events. The distraction level was predicted accurately by a nearest neighbour algorithm (KStar), with an accuracy of 95.16%. These results show that both components of the driving context model can be predicted accurately using the techniques discussed in Chapter 5.

Chapter 6 proposed an updated version of MIMIC that takes into account the context adaptation and sharing (Figure 6.2). Adaptation effects that can help in reducing driver distraction were identified. This part was critical as the adaptation actions contribute directly to reducing driver distraction. It was shown that the collaborative context can be used to enable the driver to be aware of the driving context of the peer. MIMIC can determine the driving context of the caller, and the driving context of the peer. This can be used in the assessment of the risk of engaging in a conversation.

Adaptation Effect based on Driving Context (Chapter 6)
Objective**: To design a context adapter that can be used to reduce driver distraction**



Output:

- Adaptation variables:
  - Environment (Visibility, weather),
  - Driving context.
- Adaptation actions:
  1. Delaying call notification,
  2. Delaying SMS notification,
  3. Notifying the driver of a failed incoming call,
  4. Notifying the driver of a failed incoming SMS,
  5. Pausing/Resuming dialogue,
  6. Increasing/Decreasing volume.

Figure 8.5: Summary of Chapter 6

Four algorithms were discussed to assess the safety of a driving situation. These algorithms were the following: safety determination under normal circumstances, safety determination under severe weather conditions, safety determination in the dark and safety determination in the dark and under severe weather conditions.

Retroactive and proactive types of adaptation effects were discussed. The following adaptation mechanisms were proposed (Figure 8.5): *Adaptation 1* was used to postpone a call notification; *Adaptation 2* was used to postpone an incoming text notification; *Adaptation 3* was used to notify the driver that a contact had attempted to make a call; *Adaptation 4* was used to notify the driver of a pending incoming text message; *Adaptation 5* was used to pause the dialogue when the driving situation became unsafe; and *Adaptation 6* was used to adjust the dialogue volume.

These safety algorithms and adaptation effects were integrated into the Dialogue Module of MIMIC (Figure 5.2). The Context Adapter and the Dialogue Manager implementation took into account the assessment of a driving situation and the adaptation effects. The first version of MIMIC-Prototype was developed in Chapter 4 and was updated to improve the speech recognition aspect. The implementation of the final version of MIMIC-Prototype was done using a Galaxy S3 running Android 4.1.2. The native Android noise cancellation was used to try to reduce the effects of the ambient noise.

Evaluation (Chapter 7)

Objective**: To measure the extent to which the MIMIC-Prototype can help in reducing driver distraction**

Output:

- Comparing an adaptive version of MIMIC-Prototype with a non-adaptive one,
- Results of a usability evaluation of the prototype:
    - Usability,
    - Distraction level, and
    - Perceived driver distraction.
- Adaptive version was better in terms of user preferences, usability, perceived driver distraction.

Figure 8.6: Summary of Chapter 7

Chapter 7 evaluated the extent to which MIMIC-Prototype can help in reducing driver distraction (Figure 8.6). A field study was conducted in order to determine if that objective had been met. The prototype implemented in Chapter 6 was used in the field study. Two versions were evaluated: the adaptive version with the Context-Aware Module and the non-adaptive version without the Context Adapter. The field study aimed to compare the adaptive version and the non-adaptive version in terms of usability, perceived driver distraction and distraction level. A within-subjects counter-balanced study was designed on a nine kilometre urban road. Participants had to perform several communication-related tasks using both versions of MIMIC-Prototype.

The results showed a significant preference for the adaptive version in terms of reduced driver distraction and effective adaptability. Overall, the usability results were encouraging and confirmed the initial hypothesis. The majority of the participants indicated that they were willing to use such an ICCS in future. The distraction level recorded when performing tasks was lower for the adaptive version than for the non-adaptive version (Figure 7.32). This difference was also significant when receiving calls and text messages while the driving situation was unsafe (Figure 7.29 and Figure 7.30).

8.3. Contributions

This research project showed how a mobile, context-aware ICCS can be designed in order to address the issue of driver distraction. The contributions of this research are both theoretical and practical. The following sections will describe these contributions.

8.3.1. Theoretical Contributions

This research proposed a model for speech-based, mobile ICCS (Figure 4.1). The feasibility of the prototype speech-based, mobile ICCS was evaluated through a usability study. This is important theoretically as several mobile ICCS are available, but there is no research validating the usability of such applications.

Machine learning is a powerful technique that is used in various domains today. Little research has been conducted to determine the driving context. The existing applications of machine learning to determine driver distraction use data that can be difficult to obtain. This data includes the perspiration and the pupil diameter of the driver. The design of a model for a mobile, context-aware ICCS that can use sensor data to infer the current driving context was proposed. The field study investigated the accuracy of the determination of the driving context using data that can be obtained from a mobile phone. This research proposed the design of an algorithm that can be used to determine the driving context by using mobile phone sensors (accelerometer, gyroscope, compass, proximity, light and magnetometer), web services (weather conditions) and global positioning system (GPS) (speed, direction and altitude). The results obtained after the experiments showed that the most effective algorithms were the nearest neighbour (KStar and IB1) algorithms. It was also shown that the speed, the acceleration and the linear acceleration are the most important variables in determining the current driving context. This was an important result as it showed that some relatively simple machine learning techniques can be successfully used for determining the driving context. The selected algorithms were not resource intensive, which made it suitable for a mobile phone.

A new algorithm was proposed for assessing the driving situation (Figure 6.4). Adaptation effects were used to adapt the interaction of the system with the driver. Context-aware collaboration was also used to assess the safety. Time awareness was used to help in preventing driving context misclassification. Figure 8.7 depicts the final model for a mobile, context-aware ICCS that was proposed in Chapter 6.

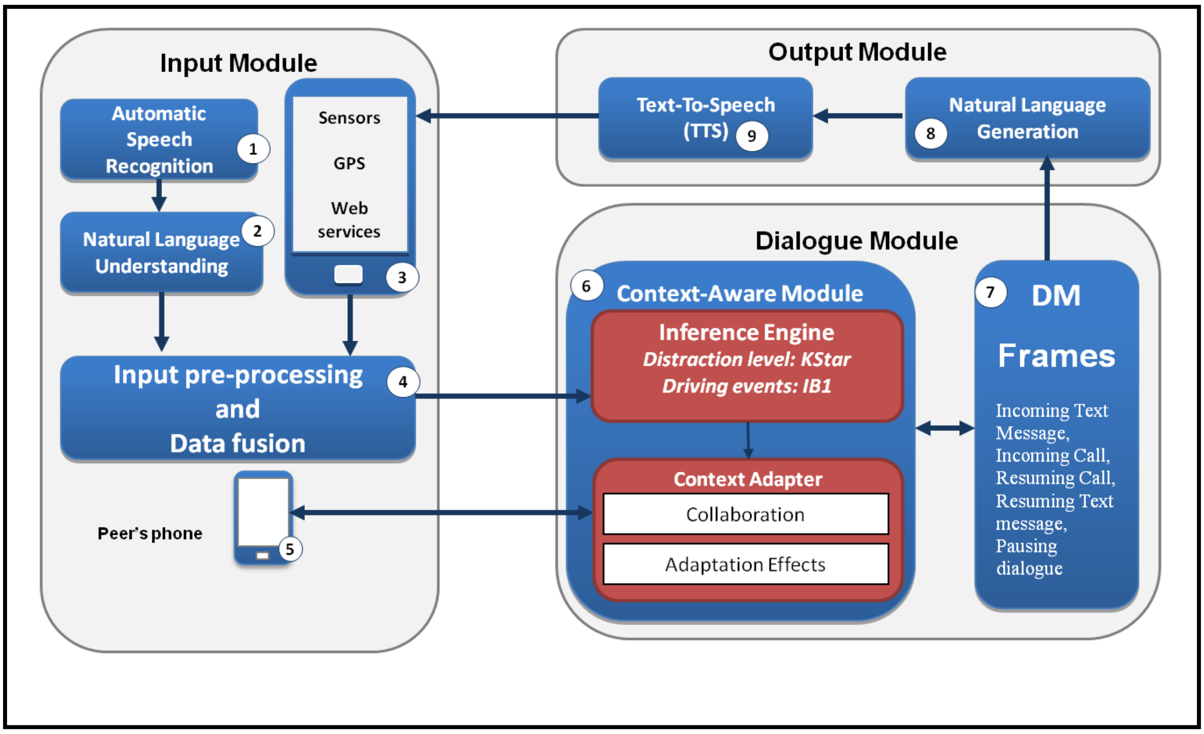


Figure 8.7: Final version of MIMIC

The field study confirmed the usability of the MIMIC-Prototype independent of the version used (adaptive or non-adaptive). MIMIC-Prototype reduced the perceived driver distraction significantly as participants found that the adaptive version of the prototype was less distracting than the non-adaptive version when receiving text messages and answering calls. Since there are no similar field studies that evaluated the benefit of a mobile, speech-based

ICCS, the research design described in Chapter 7 can be used in future for evaluating similar mobile ICCS.

MIMIC-Prototype is hands-free and eyes-free, which means that the visual and manual distraction experienced by the participants were non-existent or very low. As shown by the workload results of the field study, participants experienced little mental, physical and temporal demand. The effort (mean = 2.00) and frustration (mean = 2.07) were also low.

8.3.2. Practical Contributions

During the course of this research, several practical contributions were made by implementing prototypes that were used in the different user studies and experiments.

The proposed model for a mobile, speech-based ICCS was described in Chapter 4 (Figure 4.1). A prototype mobile application was implemented based on the proposed model (Section 2.7.1). The usability evaluation of this prototype produced encouraging results (Section 4.4).

A Context-Aware Module containing an Inference Engine was introduced (Figure 5.2) in order to enable the capture of sensor information. An updated version of MIMIC-Prototype was implemented in order to collect data that was used to train the Inference Engine. A Samsung Galaxy S3 was used and the machine learning training took place offline. The Waikato Environment for Knowledge Extraction (WEKA) package, that provides several implementations of well-known machine learning algorithms, was used for training to generate classifiers for determining the driving context. WEKA has been successfully used in different domains, but not in inferring the driving context. It was noteworthy that a good level of accuracy was obtained when using mobile phone data to infer the driving context.

The Context-Aware Module was updated by adding the Context-Adapter (Figure 6.2). This was done to enable the MIMIC-Prototype to react to the current driving context in a way to minimise the distraction caused by using the mobile phone. This prototype was later used to conduct the field study (Chapter 7).

8.4. Limitations

While conducting the field study some issues occurred that possibly increased the level of frustration of the participants. The speech recognition rate of the mobile phone was low especially when it came to short utterances such as “Yes” or “No”. This low recognition rate can be attributed to the poor quality of the noise cancellation for short utterances. Noise cancellation worked correctly with longer utterances, but often failed when the driver tried to give short responses to the system.

Another issue that was discovered during the field study was the poor quality of the mobile network signal in some locations. Mobile network signal strength fluctuated in some areas where tasks were performed. This affected the speech recognition rate, which depends on the network availability.

8.5. Recommendations and Future Work

The problem of poor speech recognition was highlighted as an issueby the results of the preliminary usability evaluation (Chapter 4) and the field study (Chapter 7). Online and offline speech recognition exists for mobile devices (Chapter 2). Most mobile phones use online speech recognition by default. The unavailability of the Internet connection in certain locations in South Africa makes online speech recognition ineffective. Therefore, when designing a speech-based, mobile ICCS, offline speech recognition is recommended in order to address the issue described above. Pocket Sphinx is an offline speech recognition engine that could be used (Huggins-Daines, Kumar, Chan *et al.*, 2006). Unfortunately, at the moment it requires acoustic and language models to be built by the developer. Generating acoustic and language models is a difficult and lengthy task, which will be part of the future work.

Drivers became used to the MIMIC-Prototype very quickly and could sometimes anticipate the options presented by the system. It is therefore important to enable the system to be interrupted (barge-in). This will help in decreasing the time on task for experienced users. This is the case especially for time-consuming commands such as sending a text message. Barge-in capabilities that are available in speech software development toolkits (SDK) such as Microsoft Speech, are not yet a common feature in mobile speech recognition application programming interfaces (API). As the results showed (Section 7.5), this can affect the mental workload of drivers. It is therefore very important, when designing a mobile ICCS, to use barge-in as a dialogue strategy. This will also be part of future work.

When designing the final version of the prototype, the average distraction level (DL) and driving event (DE) over ten seconds was considered. This choice provided good results, but could be improved by using a weighted function that gives more weight to the most recently captured DL and DE.

The possibility of expanding the range of sensors used can be explored in future research. Wearable devices such as intelligent watches (iWatch from Apple and Galaxy Gear from Samsung) are becoming popular. Blood pressure, heart rate and hydration are some of the context information that can be provided by intelligent watches. This could be another option to be investigated as part of future work.

List of References

Abowd, G. D., A. K. Dey, P. J. Brown, N. Davies, M. Smith and P. Steggles (1999). Towards a Better Understanding of Context and Context-Awareness. Proceedings of the First International Symposium on Handheld and Ubiquitous Computing - HUC99, Karlsruhe, Germany.

AdelaVoice (2010). StartTalking.

Agrawal, S., I. Constandache, S. Gaonkar, R. R. Choudhury, K. Caves and F. DeRuyter (2011). Using Mobile Phone to Write in the Air. Proceedings of the 9th International Conference on Mobile Systems, Applications and Services, Bethesda, Maryland, USA, ACM.

Akin, D. and B. Akbas (2010). "A Neural Network (NN) Model to Predict Intersection Crashes Based Upon Driver, Vehicle and Roadway Surface Characteristics." Scientific Research and Essays **5**(19): 2837-2847.

Amditis, A., L. Andreone, K. Pagle, G. Markkula, E. Deregibus, M. R. Rue, F. Belloti, A. Engelsberg, R. Brouwer, B. Peters and A. De Gloria (2010). "Towards the Automotive HMI of the Future: Overview of the AIDE-Integrated Project Results." Intelligent Transportation Systems **11**(3): 567 - 578.

Anagnostopoulos, C. B., A. Tsounis and S. Hadjiefthymiades (2007). "Context Awareness in Mobile Computing Environments." Wireless Personal Communications **42**(3): 445-464.

Andreone, L., A. Amditis, E. Deregibus, S. Damiani, D. Morreale and F. Bellotti (2005). Beyond Context-Awareness: Driver-Vehicle-Environment Adaptability. From the COMMUNICAR Project to the AIDE Concept. IFAC 16th World Congress, Prague, Czech Republic.

Android Developers. (2012). "Android SDK." Retrieved June 2012, from http://developer.android.com/.

Android Developers. (2013). "Recognizing the User's Current Activity." Retrieved June 2013, from http://developer.android.com/training/location/activity-recognition.html.

Android Studio. (2013). Retrieved July, 2013, from http://developer.android.com/sdk/installing/studio.html

Angelbrecht, A. P. (2007). Computational Intelligence: An Introduction. West Sussex, England, John Wiley & Sons.

Apple Inc. (2014). "CarPlay." from http://www.apple.com/ios/carplay/.

Bailey, T. J. (1994). The Road Safety Implications of Mobile Phone Use while Driving: A Short Literature Review, Office of Road Safety Office of Road Safety, South Australian Department of Transport**:** 14.

Basacik, D. and A. Stevens (2008). Scoping Study of Driver Distraction. London, Transport Research Laboratory, Department of Transport**:** 72.

Bayly, M., K. L. Young and M. A. Regan (2009). Sources of Distraction Inside the Vehicle and their Effects on Driving Performance. Driver Distraction: Theory, Effects, and Mitigation. M. A. Regan; J. D. Lee and K. L. Young. Boca Raton, FL, CRC Press**:** 191-213.

Becker, T., N. Blaylock, C. Gerstenberger and I. Kruijff-Korbayová (2006). The SAMMIE System: Multimodal In-Car Dialogue. Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions, Sydney, Australia.

Bedogni, L., M. Di Felice and L. Bononi (2012). By Train or by Car? Detecting the User's Motion type through Smartphone Sensors Data. Proceedings of the Wireless Days Conference, Ireland, IEEE

Bettini, C., O. Brdiczka, K. Henricksen, J. Indulska and D. Nicklas (2010). "A Survey of Context Modelling and Reasoning Techniques." Pervasive and Mobile Computing **6**(2): 161-180.

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. New York, Springer.

BMW. (2009). "BMW iDrive." Retrieved 20/10/2008, from http://www.bmw.com/com/en/insights/technology/technology_guide/articles/idrive.html.

BMW. (2013). "BMW ConnectedDrive." Retrieved November, 2014, from http://www.bmw.co.za/com/en/insights/technology/connecteddrive/2013/index.asp.

Bo, C., X. Jian, X.-Y. Li, X. Mao, Y. Wang and F. Li (2013). You're Driving and Texting: Detecting Drivers Using Personal Smart Phones by Leveraging Inertial Sensors. Proceedings of the 19th Annual International Conference on Mobile Computing and Networking, MobiCom 2013, Miami, FL, USA.

Brooke, J. (1996). SUS: A "Quick and Dirty" Usability Scale. Usability Evaluation in Industry. P. W. Jordan; B. Thomas; B. A. Weerdmeester and I. L. McClelland. London, Taylor and Francis.

Brooks, C. A. and A. Rakotonirainy (2005). In-Vehicle Technologies, Advanced Driver Assistance Systems and Driver Distraction: Research Challenges. Proceedings of the International Conference on Driver Distraction, Sydney, Australia.

Bui, T. H. (2006). Multimodal Dialogue Management - State of the Art. Enschede, The Netherlands, Centre for Telematics and Information Technology, University of Twente**:** 24.

Chang, C.-C. and C.-J. Lin (2011). "LIBSVM: A Library for Support Vector Machines." ACM Transactions on Intelligent Systems and Technology (TIST) **2**(3): 27.

Chang, Y.-S., S.-H. Hung, N. Wang and B.-S. Lin (2011). CSR: A Cloud-assisted Speech Recognition Service for Personal Mobile Devices. Proceedings of the International Conference on Parallel Processing, Taipei City.

Chen, F. (2004). Speech Interaction System-How to Improve the usability? Proceedings of the Eighth International Conference on Spoken Language Processing, Jeju Island, Korea.

Chen, G. and D. Kotz (2000). A Survey of Context-Aware Mobile Computing Research. Hanover, NH, United States, Department of Computer Science, Dartmouth College**:** 16.

Chen, H., T. Finin and A. Joshi (2003). An Intelligent Broker for Context-Aware Systems. Proceeding of the UbiCom, Nottingham, United Kingdom.

Chen, H., T. Finin and A. Joshi (2004). "An Ontology for Context-Aware Pervasive Pervasive Computing Environments." Knowledge Engineering Review **18**(3): 197–207.

Chen-, G. and D. Kotz (2000). A Survey of Context-Aware Mobile Computing Research. Hanover, NH, United States, Department of Computer Science, Dartmouth College**:** 16.

Cheverst, K., K. Mitchell and N. Davies (1999). "Design of an Object Model for a Context Sensitive Tourist GUIDE." Computer & Graphics Journal **23**(6): 883-891.

Chu, H., V. Raman, J. Shen, A. Kansal, V. Bahl and R. R. Choudhury (2014). I am a Smartphone and I Know My User is Driving. Proceedings of the 6th International Conference on Communication Systems and Networks (COMSNETS), Bangalore, India.

Cohen, M. H., J. P. Giangola and J. Balogh (2004). Voice User Interface Design. Boston, MA, Addison-Wesley.

Collis, J. and R. Hussey (2009). Business Research: A Practical Guide for Undergraduate and Postgraduate Students. Basingstoke, Hampshire, UK, Palgrave Macmillan.

Constantinos, K. C. and D. Kim (2006). A Qualitative Review of Empirical Mobile Usability Studies. Proceeding of the 12th Americas Conference on Information Systems (AMCIS), Acapulco, Mexico.

Cooper, G. (1998). "Research into Cognitive Load Theory and Instructional Design at UNSW." Retrieved on 30/6/2009, from http://education.arts.unsw.edu.au/staff/sweller/clt/.

Cowan, J. P. (1993). Handbook of Environmental Acoustics. New York, John Wiley & Sons.

Curbera, F., M. Duftler, R. Khalaf, W. Nagy, N. Mukhi and S. Weerawarana (2002). "Unraveling the Web Services Web: An introduction to SOAP, WSDL, and UDDI." IEEE Internet Computing **6**(2): 86-93.

Da Costa, P. C., K. B. Laskey and K. J. Laskey (2008). PR-OWL: A Bayesian Ontology Language for Semantic Web. Proceeding of the International Workshops on Uncertainty Reasoning for the Semantic Web (ISWC), Berlin Heidelberg, Springer.

Dey, A. (2001). "Understanding and Using Context." Personal and Ubiquitous Computing **5**(1): 4-7.

Dey, A. K., G. Abowd and D. Salber (2001). "A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications." **16**(2): 97-166.

Dingus, T. A. and S. G. Klauer (2008). The Relative Risks of Secondary Task Induced Driver Distraction (Paper No. 2008-21-0001). Proceeding of Convergence 2008, Detroit, MI, Warrendale, PA: Society of Automotive Engineers.

Direct Line Motor Insurance (2002). The Mobile Phone Report: A report on the Effects of Using a Hand-Held and a Hands-Free Mobile Phone on Road Safety. United Kingdom, Direct Line Insurance Croydon.

Doubleday, J., D. McLaren, S. Chien and Y. Lou (2011). Using Support Vector Machine Learning to Automatically Interpret MODIS, ALI, and L-band SAR Remotely Sensed Imagery for Hydrology, Land Cover, and Cryosphere Applications. Workshop on AI and Space, International Joint Conference on Artificial Intelligence, Barcelona, Spain.

Drews, F. A., P. Monisha and D. L. Strayer (2008). "Passenger and Cell Phone Conversations in Simulated Driving." Experimental Psychology: Applied **14**(4): 392-400.

Efstratiou, C. (2004). Coordinated Adaptation for Adaptive Context-Aware Applications. Doctor of Philosophy, Lancaster University.

Eriksson, J., L. Girod, B. Hull, R. Newton, S. Madden and Balakrishnan (2008). The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring. Proceeding of the 6th International Conference on Mobile Systems, Applications, and Services (MobiSys), Breckenridge, CO, USA, ACM.

European Commission (2000). Safe and Efficient In-Vehicle Information and Communication Systems: A European Statement of Principles on Human Machine Interface. Brussels, Belgium, European Commission.

Feld, M. and C. Muller (2011). The Automotive Ontology: Managing Knowledge Inside the Vehicle and Sharing it Between Cars. Automotive UI, Salzburg, Austria.

Ford. (2008). "Ford Sync." Retrieved 11/09/2009, 2009, from http://www.fordvehicles.com/technology/sync/.

Frigui, H., L. Zhang, P. Gader, J. N. Wilson, K. Ho and A. Mendez-Vazquez (2012). "An Evaluation of Several Fusion Algorithms for Anti-tank Landmine Detection and Discrimination." Information Fusion **13**(2): 161-174.

Gellatly, A. (1997). The Use of Speech Recognition Technology in Automotive Applications. PhD.

General Motors. (2003). "On Star." Retrieved 12/12/2009, from http://www.onstar.com/web/portal/handsfreecalling.

Gilbert, M. and J. Feng (2008). "Speech and Language Processing over the Web." IEEE Signal Processing **25**(3): 42-60.

Google Inc. (2008). "Android Developer." Retrieved June 2008, from http://developer.android.com.

Google Inc. (2009). "Android Speech API." 2011, from://developer.android.com/reference/android/speech/package-summary.html.

Google Inc (2010). Android Froyo.

Google Inc (2011). Android Gingerbread.

Gordon, C. P. (2008). Crash Studies of Driver Distraction. Driver Distraction: Theory, Effects and Mitigation. M. A. Regan; J. D. Lee and K. L. Young. Boca Raton, Florida, CRC Press**:** 281-304.

Haigney, D. (1997). Mobile Phones and Driving: A Literature Review. Birmingham, The Royal Society for the Prevention of Accidents (RosPa),.

Haigney, D. and S. Westerman (2001). "Mobile (Cellular) Phone Use and Driving: A Critical Review of Research Methodology." Ergonomics **44**(2): 132-143.

Hall, M., E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten (2009). "The WEKA Data Mining Software: An Update." SIGKDD Explorations **11**(1): 10-18.

Halpin, T. (1998). Object-Role Modeling (ORM/NIAM). Handbook on Architectures of Information Systems. P. Bernus; K. Mertins and G. Schmidt. Berlin, Springer. **2:** 81-101.

Hart, S. G. and L. E. Staveland (1988). "Development of NASA-TLX: Results of Empirial and Theoretical Research." Advances in Psychology **52**: 139-183.

Hartson, R. H. (1998). "Human-Computer Interaction: Interdisciplinary Roots and Trends." Journal of Systems and Software **43**(2): 103-118.

Healey, J. A. and R. W. Picard (2004). "Detecting Stress During Real-World Tasks Using Physiological Sensors." Transactions on Intelligent Transportation Systems **6**(2): 156-166.

Henricksen, K. and J. Indulska (2006). "Developing Context-Aware Pervasive Computing : Models and Approaches." Pervasive and Mobile Computing **2**(1): 37-64.

Ho, J. and S. S. Intille (2005). Using Context-Aware computing to Reduce the Perceived Burden of Interruptions from Mobile Devices. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI.05), Porland, Oregon, USA, ACM.

Hofer, T., W. Schwinger, M. Pichler, G. Leonhartsberger, J. Altmann and W. Retschitzegger (2003). Context-Awareness on Mobile Devices-the Hydrogen Approach. Proceedings of the 36th Annual Hawaii International Conference on System Sciences (HICSS), Big Island, HI, USA, IEEE.

Hoff, J., J. Grell, N. Lohrman, c. Stehly, J. Stoltzfus, G. Wainwright and W. S. Hoff (2013). "Distracted Driving and Implications for Injury Prevention in Adults." Journal of trauma nursing **20**(1): 31-34.

Hofstee, E. (2006). Constructing a Good Dissertation: A Practical Guide to Finishing a Masters, MBA or PHD on Schedule. Johannesburg, South Africa, EPE.

Horrock, I. (2011). OWL: A Descriptive Logic Based Ontology Language. International Conference on Principles and Practice of Constraint Programming (CP 2005), Lecture Notes of Computer Sciences.

Horvitz, E., J. Breese, D. Heckerman, D. Hovel and K. Rommelse (1998). The Lumiere Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users. Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence (UAI), Madison, Wisconsin, USA, Morgan Kaufmann Publishers Inc.

Hosking, S. G., K. L. Young and M. A. Regan (2009). "The Effects of Text Messaging on Young Drivers." Human Factors **54**(4): 582-592.

Huggins-Daines, D., M. Kumar, A. Chan, A. W. Black, M. Ravishankar and A. I. Rudnicky (2006). PocketSphinx: A Free, Real-time Continuous Speech Recognition System for Handheld Devices. Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP), Toulouse, France, IEEE.

Iqbal, S. T., E. Horvitz, Y.-C. Ju and E. Mathews (2011). Hang on a Sec! Effects of Proactive Mediation of Phone Conversations while Driving. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11), Vancouver, BC, ACM.

Iqbal, S. T., Y.-C. Ju and E. Horvitz (2010). Cars, Calls, and Cognition: Investigating Driving and Divided Attention. Proceedings of the SIGCHI on Human Factors in Computing Systems (CHI 2010), Atlanta, GA, USA, ACM.

Ishigami, Y. and R. M. Klein (2009). "Is a Hand-free Phone Safer than a Handheld Phone?" Journal of Safety Research **40**(2): 157-164.

Ito, H., H. Uno, B. Atsumi and M. Akamatsu (2001). "Visual Distraction While Driving: Trends in Research and Standardization." International Association of Traffic and Safety Sciences (IATSS) Research **25**(2): 20-28.

Jain, A. K. and J. Mao (1996). "Artificial Neural Networks: A Tutorial." IEEE Computer **29**(3): 31-44.

Jameson, A. (2003). Adaptive Interfaces and Agents. The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications. J. A. Jacko and A. Sears. Mahwah, NJ, USA, Erlbaum Associates Inc. **:** 305–330.

Jenness, J. W., R. J. Lattanzio, M. O'Toole and N. Taylor (2002). Voice-Activated Dialling or Eating a Cheeseburger: Which is More Distracting during Simulated Driving? Proceeding os the Human Factors and Ergonomics Society Annual Meeting, Santa Monica, CA, SAGA Publications.

Joachims, T. (1999). "SVMLight: Support Vector Machines." Retrieved March 2012, from http://svmlight.joachims.org/.

Kampichler, C., R. Wieland, S. Calmé, H. Weissenberger and S. Arriaga-Weiss (2010). "Classification in Conservation Biology: A Comparison of Five Machine-Learning Methods." Ecological Informatics **5**(6): 441-450.

Kaplan, E. D. and C. J. Hegarty (2005). Understanding GPS: Principles and Applications Norwood, MA, Artech House.

Khalil, R. T., A. F. Khalifeh and K. A. Darabkh (2012). Mobile-Free Driving with Android Phones: System Design and Performance Evaluation. Proceeding of the 9th International Multi-Conference on Systems, Signals and Devices, Chemnitz, Germany, IEEE.

KIA. (2010). "Kia Motors and Microsoft Usher in New Era of In-Car Technology." Retrieved 5/5/2010, from http://www.microsoft.com/presspass/features/2010/jan10/01-05kiauvo.mspx.

KIA. (2014). "Kia Launching UVO 2.0 in 2014 Sorento." Retrieved December, 2013, from http://www.autoblog.com/2012/12/01/kia-launching-uvo-2-0-in-2014-sorento.

Kim, S., H. Zhang, R. Wu and L. Gong (2011). Dealing with Noise in Defect Prediction. Proceedings of the 33rd international Conference on Software Engineering - ICSE '11, Waikiki, Honolulu, Hawaii, ACM Press.

Kjeldskov, J. and C. Graham (2003). A Review of Mobile HCI Research Methods. Mobile HCI 2003, Udine, Italy, Springer-Verlag Berling Heidelberg.

Klauer, S. G., T. A. Dingus, V. L. Neale, J. D. Sudweeks and D. J. Ramsey (2006). The Impact of Driver Inattention On Near-Crash/Crash Risk: An Analysis Using the 100-Car Naturalistic Driving Study Data. Blacksburg, Virginia 24061, National Highway Traffic Safety Administration**:** 224.

Klein, H. K. and M. D. Myers (1999). "A Set of Principles for Conducting and Evaluating Interpretive Field Studies in Information Systems." MIS Quarterly - Special Issue on Intensive Research in Information Systems **23**(1): 67-94.

Klein, L. A. (2004). Sensor and Data Fusion: A Tool for Information Assessment and Decision Making. Bellingham, Washington, USA, SPIE-The International Society for Optical Engineering

Konstantopoulos, P., P. Chapman and D. Crundall (2010). "Driver's Visual Attention as a Function of Driving Experience and Visibility. Using a Driving Simulator to Explore Drivers' Eye Movements in Day, Night and Rain Driving." Accident Analysis and Prevention **42**(3): 827-834.

Lam, T. C., J. J. Ding and J.-C. Liu (2008). "XML Document Parsing: Operational and Performance Characteristics." Computing Practices **41**(9): 30-37.

Lane, N. D., E. Miluzzo, H. Lu, D. Peebles, T. Choudhury and A. T. Campbell (2010). "A Survey of Mobile Sensing." Ad Hoc and Sensor Networks **48**(9): 140-150.

Lavie, T. and J. Meyer (2010). "Benefits and Costs of Adaptive User interfaces." International Journal of Human Computer Studies **68**(8): 508-524.

Lee, J. D. (2007). "Technology and Teen Drivers." Journal of Safety Research **38**(2): 203-213.

Lee, J. D., K. L. Young and M. A. Regan (2008a). Defining Driver Distraction. Driver Distraction: Theory, Effects and Mitigation. M. Regan; J. Lee and K. Young. Boca Raton, Florida, CRC Press.

Lee, J. D., K. L. Young and M. A. Regan (2008b). Driver Distraction: Theory, Effects, and Mitigation. Boca Raton, FL, USA, CRC Press.

Lei, X., A. Senior, A. Gruenstein and J. Sorensen (2013). Accurate and Compact Large Vocabulary Seech Recognition on mobile Devices. Proceedings of the 14th Annual

Conference of the International Speech Communication Association - Interspeech 2013, Lyon, France.

Lenhart, A., R. Ling, S. Campbell and K. Purcell (2010). Teens and Mobile Phones - Text Messaging Explodes as Teens Embrace it as the Centerpiece of their Communication Strategies with Friends. Washington, DC, Pew Research Center**:** 20.

Liang, Y. (2009). Detecting Driver Distraction. Philosophy Degree, University of Iowa.

Lin, T. W., S. L. Hwang and P. A. Green (2009). "Effects of Time-gap Settings of Adaptive Cruise Control (ACC) on Driving Performance and Subjective Acceptance in a Bus Driving Simulator." Safety science **47**(5): 620-625.

Lindqvist, J. and J. Hong (2011). Undistracted Driving: A Mobile Phone that Doesn't Distract. Proceedings of HotMobile'11, Phoenix, Arizona USA.

Mattes, S. and A. Hallen (2009). Surrogate Distraction Measurement Techniques: The Lane Change Test. Driver Distraction: Theory, Effects, and Mitigation M. A. Regan; J. D. Lee and K. L. Young. Boca Raton, FL, CRC Pres**:** 107-121.

Maybury, M. T. (2002). Language Technology: A Survey of the State of the Art Language Resource. Multimodal Language Resources. J. Mariani, Springer Verlag.

McEvoy, S. P., M. R. Stevenson and M. Woodward (2006). "The Impact of Driver Distraction on Road Safety." Injury Prevention **12**(4): 242-247.

Mednis, A., G. Strazdins, R. Zviedris, G. Kanonirs and L. Selavo (2011). Real Time Pothole Detection using Android Smartphones with Accelerometers. Proceeding of the International Conference on Distributed Computing in Sensor Systems and Workshops (MobiSensor 2011), Barcelona, Spain, IEEE.

Michon, J. A. (1985). A Critical View on Driver Behavior Models: What Do We Know, What Should We Do? Proceedings of the General Motors Symposium on Human Behavior and Traffic Safety, Warren, Michigan, Plenum Press.

Milloy, S. L. and J. K. Caird (2011). External Driver Distractions: The Effects of Video Billboards and Wind Farms on Driver Performance. Handbook of Driving Simulation for Engineering, Medicine and Psychology. D. L. Fisher; M. Rizzo; J. K. Caird and J. D. Lee. Boca Raton, FL, USA, CRC Press.

Mitchell, T. M. (1997). Machine Learning. Burr Ridge, IL, McGraw-Hill.

Moore, P., B. Hu, X. Zhu, W. Campbell and M. Ratcliffe (2007). "A Survey of Context Modeling for Pervasive Cooperative Learning."

Muller, C. and G. Weinberg (2011). "Multimodal Input in the Car, Today and Tomorrow." IEEE Multimedia: 98-103.

Nasoz, F., C. L. Lisetti and A. V. Vasilakos (2010). "Affectively Intelligent and Adaptive Car Interfaces." Information Sciences **180**(20): 3817-3836.

National Highway Traffic Safety Administration (2010). Distracted Driving 2009. Traffic Safety Notes DOT, Department of Transportation.

National Highway Traffic Safety Administration (2012a). Visual-Manual NHTSA Driver Distraction Guidelines for In-Vehicle Electronic Devices, US Department of Transport**:** 281.

National Highway Traffic Safety Administration (2012b). Young Drivers Report the Highest Level of Phone Involvement in Crash or Near-Crash Incidences. Washington, DC, U.S. Department of Transport**:** 5.

Nelson, E., P. Atchley and T. Little (2009). "The Effects of Perception of Risk and Importance of Answering and Initiating a Cellular Phone Call while Driving." Accident Analysis and Prevention **41**(3): 438-444.

Ng, A. Y., A. Coates, M. Diel, V. Ganapathi, J. Schulte, B. Tse, E. Berger and E. Liang (2006). "Autonomous Inverted Helicopter Flight via Reinforcement Learning." Experimental Robotics IX **21**: 363-372.

Nielsen, J. and T. K. Landauer (1993). A Mathematical Model of the Finding of Usability Problems. Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems, Amsterdam, The Netherlands, ACM.

Niklas, D. and B. Mitschang (2004). "On Building Location Aware Applications using an Open Platform Based on the Nexus Augmented World Model." Software and System Modeling **3**(4): 303-313.

Nissan Technology Magazine. (2007). "Drunk-driving Prevention Concept Car." Retrieved 2011, 2007, from http://www.nissan-global.com/EN/TECHNOLOGY/OVERVIEW/dpcc.html.

Nuance Inc. (2013). "Dragon Mobile Assistant." Retrieved July, 2014, from http://www.nuance.com/for-individuals/mobile-applications/dragon-mobile-assistant/index.htm.

NVIDIA. (2014). "NVIDIA Powers New Honda Infotainment System." Retrieved November, 2014, from nvidianews.nvidia.com/NEWS/NVIDIA-Powers-News-Honda-Infotainment-System-bd6.aspx.

Oh, Y., A. Schmidt and W. Woo (2007). Designing, Developing, and Evaluating Context-Aware Systems. International Conference on Multimedia and Ubiquitous Engineering.

Olson, R. L., R. J. Hanowski, J. Hickman and J. Bocanegra (2009). Driver Distraction in Commercial Vehicle Operations. Blacksburg, VA, Center for Truck and Bus Safety, Virginia Tech Transportation Institute**:** 285.

Oxford Dictionaries. (2011). Retrieved May 2011, from http://www.oxforddictionaries.com/definition/english/context.

Phyu, T. N. (2009). Survey of Classification Techniques in Data Mining. Proceedings of the International MultiConference of Engineers and Computer Scientists (IMECS), Hong Kong.

Pisano, P. A., L. C. Goodwin and M. A. Rossetti (2008). US Highway Crashes in Adverse Road Weather Conditions. Proceeding of the 24th Conference on International Interactive Information and Processing Systems for Meteorology, Oceanography and Hydrology (IIPS), New Orleans, LA.

Poslad, S. (2009). Ubiquitous Computing: Smart Devices, Environments and Interactions. West Sussex, PO19 8SQ, United Kingdom, John Wiley & Sons.

Quinlan, J. R. (1993). C4.5: Programs for Machine Learning. San Francisco, Morgan Kaufmann.

Ranney, T. (2008). Driver Distraction: A Review of the Current State-of-Knowledge. Blacksburg, Virginia, National Highway Traffic Safety Administration**:** 32.

Richard, C., J. Campbell and J. Brown (2006). Task Analysis of Intersection Driving Scenarios: Information Processing Bottlenecks Seattle, WA 98109-3598, Federal Highway Administration**:** 222.

Road Traffic Management Corporation (2008). Road Traffic Report Pretoria, South Africa, Department of Transport**:** 82.

Rothermel, K., D. Dudkowski, F. Durr, M. Bauer and C. Becker (2003). Ubiquitous Computing: More than Computing Anytime Anyplace? Proceedings of the 49th Photogrammetric Week (PhoWo), Stuttgart, Germany.

SAAB. (2011). "SAAB-IQon: Open Innovation in Car Infotainment." Retrieved April 2011, from http://www.androidcentral.com/saab-turns-android-their-iqon-car-infotainment-system.

Salkham, A., R. Cunningham, A. Senart and V. Cahill (2006). A Taxonomy of Collaborative Context-Aware Systems. UMICS'06, Luxemburg.

Saunders, M., P. Lewis and A. Thornhill (2003). Research Methods for Business Students. London, Financial Times Prentice Hall Inc.

Saunders, M., P. Lewis and A. Thornhill (2009). Research Methods for Business Students. Essex, England, Pearson Education Limited.

Schilit, B., N. Adams and R. Want (1994). Context-Aware Computing Applications. Proceedings of IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), Santa Cruz, California, IEEE Computer Society Press.

Schmidt, A., M. Beigl and H.-W. Gellersen (1999). "There is More Context than Location." Computer & Graphics Journal **23**(6): 893-902.

Schölkopf, B. and A. J. Smola (2002). Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond. Cambridge, MA, The MIT Press.

Schölkopf, B., A. J. Smola and K.-R. Müller (1998). "Nonlinear Component Analysis as a Kernel Eigenvalue Problem." Neural Computation **10**(5): 1299-1319.

Seneff, S., R. Lau and J. Polifroni (1999). Organization, Communication, and Control in the GALAXY-II Conversational System. Proceedings of EUROSPEECH'99, Budapest, Hungary.

Shah, A. A., J. D. Lee, N. P. Mahalik and J. Namkoog (2006). Intelligent Transportation-Deployment and Development Process in Korea. Urban Transport XII. Urban Transport and the Environment in the 21st Century. C. A. Brebbia and V. Dolezel. Prague , Czech Republic, Wessex Institute of Technology**:** 10.

Shutko, J., K. Mayer, E. Laansoo and L. Tirejina. (2009). "Driver Workload Effects of Cell Phone, Music Player, and Text Messaging Tasks with the Sync-Voice Interface versus the Devices' Handheld Visual-Manual Interfaces." Retrieved 1/6/2009, from https://shop.sae.org/technical/papers/2009-01-0786.

Staubach, M. (2009). "Factors Correlated with Traffic Accidents as a Basis for Evaluating Advanced Driver Assistance Systems." Accident Analysis and Prevention **41**(5): 1025-1033.

Strang, T. and C. Linnhoff-Popien (2004). A Context Modeling Survey. Proceedings of the International Workshop on Advanced Context Modelling, Reasoning And Management, in Conjunction with UbiComp 2004, Nottingham, England.

Strayer, D. L., J. M. Watson and F. A. Drews (2011). Cognitive Distraction While Multitasking in the Automobile. The Psychology of Learning and Motivation. B. Ross. Burlington, Academic Press. **54:** 29-58.

Stuster, J., Z. Coffman and D. Warren (1998). Synthesis of Safety Research Related to Speed and Speed Management. McLean, VA, Federal Highway Administration.

Stutts, J. C., D. W. Reinfurt, L. Staplin and E. A. Rodgman (2001). The Role of Driver Distraction in Traffic Crashes. Washington, DC, AAA Foundation for Traffic Safety**:** 70.

Subramanian, N. and L. Chung (2000). Architecture-driven Embedded Systems Adaptation for Supporting Vocabulary Evolution. Proceedings of the International Symposium on Principles of Software Evolution, Kanazawa, Japan, IEEE.

Suhm, B. (2003). Towards Best Practices for Speech User Interface Design. Proceedigns of the 8th European Conference on Speech Communication and Technology (Eurospeech), Geneva, Switzerland.

Tango, F. and M. Botta (2013). "Real-Time Detection System of Driver Distraction Using Machine Learning." Transactions on Intelligent Transportation Systems **14**(2): 894-905.

Tashev, I., M. Seltzer, Y.-C. Ju, Y.-Y. Wang and A. Acero (2009). Commute UX: Voice Enabled In-car Infotainment System. Proceedings of Mobile HCI '09, Workshop on Speech in Mobile and Pervasive Environments SiMPE, Bonn, Germany, ACM.

Tchankue, P. (2011). An Intelligent Multimodal Interface for In-Car Communication Systems. Master, Nelson Mandela Metropolitan University.

Tchankue, P., D. Vogts and J. Wesson (2010a). Design and Usability Evaluation of a Multimodal Interface for In-car Communication Systems. Proceedings of SAICSIT'10: Annual Research Conference of the South African Institute of Computer Scientist and Information Technologists, Bela Bela, South Africa, ACM.

Tchankue, P., J. Wesson and D. Vogts (2010b). MIMI: A Multimodal and Adaptive Interface for an In-Car Communication System. Proceedings of SATNAC 2010, Stellenbosch, South Africa.

Tchankue, P., J. Wesson and D. Vogts (2011). The Impact of an Adaptive User Interface on Reducing Driver Distraction. Proceedings of the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications, Salzburg, Austria.

Tchankue, P., J. Wesson and D. Vogts (2012). Are Mobile In-Car Communication Systems Feasible? A Usability Study. Proceedings of the South African Institute for Computer Scientist and Information Technologists Conference (SAICSIT), Centurion, Tshwane, South Africa, ACM.

Tchankue, P., J. Wesson and D. Vogts (2013). Using Machine Learning to Predict the Driving Context whilst Driving. Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference, East London, South Africa, ACM.

The Automotive Association of South Africa. (2011). "Why Distracted Driving is Becoming a Fatal Problem." Retrieved December 2011, from http://www.aa.co.za/about/press-room/press-releases/why-distracted-driving-is-becoming-a-fatal-problem.html.

Theodoridis, S. and K. Koutroumbas (2009). Pattern Recognition. Burlington, MA, Academic Press.

Tison, J., N. Chaudhary and L. Cosgrove (2011). National Phone Survey on Distracted Driving Attitudes and Behaviors. Washington, DC, National Highway Traffic Safety Administration**:** 44.

Toledo, T., H. N. Koutsopoulos and M. E. Ben-Akiva (2003). "Modeling Integrated Lane-Changing Behavior." Transportation Research Record: Journal of the Transportation Research Board **1857**(1): 30-38.

Toyota. (2012). "Toyota Entune." Retrieved June 2012, from http://www.toyota.com/entune/.

Tullis, T. and B. Albert (2008). Measuring the User Experience: Collecting, Analizing and Presenting Usability Metrics. San Francisco, CA, USA, Morgan Kaufmann Publishers.

van Tonder, B. and J. Wesson (2010). Is Tilt Interaction Better than Keypad Interaction for Mobile Map-based Applications? Proceedings of SAICSIT '10 South African Institute of Computer Scientists and Information Technologists, Bela Bela, South Africa, ACM.

Veeraraghavan, H., N. Bird and S. Atev (2007). "Classifiers for Driver Activity Monitoring." Transportation Research Part C: Emerging Technologies **15**(1): 51-67.

Verster, J., J. Taillard, P. Sagaspe, B. Olivier and P. Philip (2011). "Prolonged Nocturnal Driving can be as Dangerous as Severe Alcohol-impaired Driving." Sleep Research **20**(4): 585-588.

Verwey, W. B. (2000). "On-line Driver Workload Estimation: Effects of Road Situation and Age on Secondary Task Measures." Ergonomics **43**(2): 187-209.

Villing, J. (2009). In-vehicle Dialogue Management: Towards Distinguishing between Different Types of Workload. Proceeding of the 10th Annual Meeting of the Special Interest Group in Discourse and Dialogue, Queen Mary University of London.

Vlingo Corporation. (2010). "Vlingo Personal Assistant." Retrieved March 2011, from http://www.vlingo.com/.

Wang, X. H., T. Gu, D. Q. Zhang and H. K. Pung (2004). Ontology Based Context Modeling and Reasoning using OWL. Proceeding of the IEEE Conference on Pervasive Computing and Communications Workshops (PerCom 2004 Workshops), Orlando, FL, IEEE Computer Society.

Wang, Y., J. Yang and H. Liu (2013). Sensing Vehicle Dynamics for Determining Driver Phone Use. Proceeding of the 11th International Conference on Mobile Systems, Applications, and Services (MobiSys), Tapei, Taiwan, ACM.

Weißenberg, N., R. Gartmann and A. Voisard (2006). "An Ontology-Based Approach to Personalize Situation-Aware Mobile Service Supply." GeoInformatica **10**(1): 55-90.

Welch, G. and G. Bishop (2000). An Introduction to the Kalman Filter. Chapel Hill, North Carolina, Department of Computer Sciences, University of North Carolina.

Weston, J., A. Elisseeff, G. BakIr and F. Sinz. (2005). "The Spider Machine Learning Toolbox." Retrieved January 2014, 2014, from http://people.kyb.tuebingen.mpg.de/spider/.

Wickens, C. D. (1984). Engineering Psychology and Human Performance. Columbus, Merrill.

Wickens, C. D. (1992). Engineering Psychology and Human Performance, HarperCollins Publishers.

Wickens, C. D. (2008). "Multiple resource and mental workload." Human Factors: 449-455.

World Wide Web Consortium. (2009). "OWL Web Ontology Language Overview - W3C Recommendation 10 February 2004." Retrieved December 2011, from http://www.w3.org/TR/owl-features/.

Wu, H., M. Siegel, R. Stiefelhagen and J. Yang (2002). Sensor Fusion Using Dempster-Shaffer Theory. Proceedings of The 19th IEEE Instrumentation and Measurement Technology (IMTC), Anchorage, AK, USA, IEEE.

Yang, J., S. Sidhom, G. Chandrasekaran, T. Vu, H. Liu, N. Cecan, Y. Chen, M. Gruteser and R. P. Martin (2012). "Sensing Driver Phone Use with Acoustic Ranging through Car Speakers." IEEE Transactions on Mobile Computing **11**(9): 1426-1439.

Yannis, G., E. Papadimitriou, P. Papantoniou and C. Voulgari (2013). "A Statistical Analysis of the Impact of Advertising Signs on Road Safety." International Journal of Injury Control and Safety Promotion **20**(2): 111-120.

Yao, A., J. Gall and L. Van Gool (2010). "A Hough Transform-Based Voting Framework for Action Recognition." Computer Vision and Pattern Recognition (CVPR): 2061-2068.

Yekhshatyan, L. (2010). Detecting Distraction and Degraded Driver Performance with Visual Behavior Metrics. PhD (Doctor of Philosophy), University of Iowa.

Yokouchi, K., H. Ideno and M. Ota (2000). Car-navigation Systems. Intelligent Transport Systems. Tokyo, Japan, Mitsubishi Electric Advance **91:** 10-14.

You, C.-W., N. D. Lane, F. Chen, R. Wang, Z. Chen, T. J. Bao, M. Montes-de-Oca, Y. Cheng, M. Lin, L. Torresani and A. T. Campbell (2013). CarSafe App: Alerting Drowsy and Distracted Drivers using Dual Cameras on Smartphones. Proceedings of the 11th International Conference on Mobile Systems, Applications and Services (Mobisys), Tapei, Taiwan, ACM

Young, K. L., M. A. Regan and M. Hammer (2007). Driver Distraction: A Review of the Literature. Distracted Driving. Sydney, Autralian College of Road Safety**:** 379-405.

Zhang, D., J. Cao, J. Zhou and M. Guo (2009). Extended Dempster-shafer Theory in Context Reasoning for Ubiquitous Computing Environments. International Conference on Computational Science and Engineering (CSE), Vancouver, BC, Canada, IEEE.

Zhao, N., B. Reimer, B. Mehler, L. A. D'Ambrosio and J. F. Coughlin (2013). "Self-reported and Observed Risky Driving Behaviors among Frequent and Infrequent Cell Phone Users." Accident Analysis and Prevention **61**: 71-77.

ZoomSafer. (2011). "Zoomsafer: Texting While Driving." Retrieved 2 March, 2011, from http://zoomsafer.com/01c/.

Appendix A: Post-Test Questionnaire for the Usability Evaluation

PLEASE COMPLETE THE POST TASK QUESTIONNAIRE

A. Cognitive Load (NASA Task Load index)							
1. **Mental demand:** How mentally demanding were the tasks?							
	Very Low	1	2	3	4	5	Very High
2. **Physical demand:** How physically demanding were the tasks?							
	Very Low	1	2	3	4	5	Very High
3. **Temporal demand:** How hurried or rushed was the pace of the tasks?							
	Very Low	1	2	3	4	5	Very High
4. **Performance:** How successful were you in accomplishing what you were asked to do?							
	Very Low	1	2	3	4	5	Very High
5. **Effort:** How hard did you have to work to accomplish your level of performance?							
	Very Low	1	2	3	4	5	Very High
6. **Frustration:** How insecure, discouraged, irritated, stressed, and annoyed were you?							
	Very Low	1	2	3	4	5	Very High

B. Usability (System Usability Scale)							
1. I think that I would like to use this system frequently.							
	Strongly disagree	1	2	3	4	5	Strongly agree
2. I found this system unnecessarily complex							
	Strongly disagree	1	2	3	4	5	Strongly agree

3. I thought this system was easy to use							
	Strongly disagree	1	2	3	4	5	Strongly agree
4. I think that I would need assistance to be able to use this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
5. I found the various functions in this system were well integrated							
	Strongly disagree	1	2	3	4	5	Strongly agree
6. I thought there was too much inconsistency in this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
7. I would imagine that most people would learn to use this system very quickly							
	Strongly disagree	1	2	3	4	5	Strongly agree
8. I found this system very cumbersome/awkward to use							
	Strongly disagree	1	2	3	4	5	Strongly agree
9. I felt very confident using this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
10. I needed to learn a lot of things before I could get going with this system							
	Strongly disagree	1	2	3	4	5	Strongly agree

C. Speech UI							
1. I felt confused owing to the poor recognition rate							
	Strongly disagree	1	2	3	4	5	Strongly agree

2. I felt that the system did not recognize my voice							
	Strongly disagree	1	2	3	4	5	Strongly agree
3. The utterances issued by the system were clear enough							
	Strongly disagree	1	2	3	4	5	Strongly agree
4. I knew when the system wanted a response from me							
	Strongly disagree	1	2	3	4	5	Strongly agree

D. General
1. Identify the **most positive** aspect of the system
2. Identify the **most negative** aspect of the system
3. Please provide any **general comments or suggestions** for improvement

Appendix B: Biographical Details for the Field Study

Biographical Details (mark **X** when appropriate**)**											
Age	18 - 20		21 – 29		30 – 39		40 – 49		50 and +		
Gender	Male		Female								
Student?		Year of study:		or	Staff						
Mobile Phone Experience (Years)											
< 1		1 – 2		3 – 5		> 5					
First Language											
Afrikaans		English		Xhosa							
Driving Experience (year)											
< 1		1 – 2		3 – 5		> 5					
How often do you read or write text messages while driving? *(provide a number)*											
In Motion											
In the past 24h		Past 2 days		Past week		Past month					
Whilst stopped											
In the past 24h		Past 2 days		Past week		Past month					
How safe do you feel when receiving or sending a text message while driving?											
Very safe	1	2	3	4	5	Very unsafe					

How often do you call or answer calls while driving? *(provide a number)*								
In Motion								
Past 2 days		Past week		Past month				
Whilst stopped								
Past 2 days		Past week		Past month				
How safe do you feel when receiving or making a call while driving?								
Very Safe	1	2	3	4	5	Very unsafe		
Have you ever used a speech-based in-car communication system?	Yes		No					
Do you think that a speech-based in-car communication system can help in reducing driver distraction?	Yes		No					

Appendix C: Post-Test Questionnaires for Trip 1 and Trip 2

Post-Task Questionnaire (Trip 1)

A. Cognitive Load (NASA Task Load Index)							
1. Mental demand: How mentally demanding were the tasks?							
	Very Low	1	2	3	4	5	Very High
2. Physical demand: How physically demanding were the tasks?							
	Very Low	1	2	3	4	5	Very High
3. **Temporal demand:** How hurried or rushed was the pace of the tasks?							
	Very Low	1	2	3	4	5	Very High
4. **Performance:** How successful were you in accomplishing what you were asked to do?							
	Very Low	1	2	3	4	5	Very High
5. **Effort: How hard did you have to work to accomplish your level of performance?**							
	Very Low	1	2	3	4	5	Very High
6. **Frustration:** How insecure, discouraged, irritated, stressed and annoyed were you?							
	Very Low	1	2	3	4	5	Very High

B. Usability (System Usability Scale)							
1. I think that I would like to use this system frequently.							
	Strongly disagree	1	2	3	4	5	Strongly agree
2. I found this system unnecessarily complex							
	Strongly disagree	1	2	3	4	5	Strongly agree
3. I thought this system was easy to use							
	Strongly disagree	1	2	3	4	5	Strongly agree
4. I think that I would need assistance to be able to use this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
5. I found the various functions in this system were well integrated							
	Strongly disagree	1	2	3	4	5	Strongly agree
6. I thought there was too much inconsistency in this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
7. I would imagine that most people would learn to use this system very quickly							
	Strongly disagree	1	2	3	4	5	Strongly agree
8. I found this system very cumbersome/awkward to use							
	Strongly disagree	1	2	3	4	5	Strongly agree
9. I felt very confident using this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
10. I needed to learn a lot of things before I could get going with this system							
	Strongly disagree	1	2	3	4	5	Strongly agree

C. DISTRACTION							
1. I did not feel distracted while making calls with the system							
	Strongly disagree	1	2	3	4	5	Strongly agree
2. I felt not distracted while sending text messages with the system							

	Strongly disagree	1	2	3	4	5	Strongly agree
3. I did not feel distracted while receiving text messages with the system							
	Strongly disagree	1	2	3	4	5	Strongly agree
4. I did not feel distracted while receiving calls with the system							
	Strongly disagree	1	2	3	4	5	Strongly agree

D. ADAPTATIVITY							
1. I felt that this system took my context into consideration when receiving calls							
	Strongly disagree	1	2	3	4	5	Strongly agree
2. I felt that this system took my context into consideration when receiving text messages							
	Strongly disagree	1	2	3	4	5	Strongly agree
3. I felt that this system took the Peer's context into consideration when sending text messages or making calls							
	Strongly disagree	1	2	3	4	5	Strongly agree

E. General
1. Identify the **most positive** aspect of the system
2. Identify the **most negative** aspect of the system

3. Please provide any **general comments or suggestions** for improvement

Post-Task Questionnaire (Trip 2)

A. Cognitive Load (NASA Task Load Index)							
1. **Mental demand:** How mentally demanding were the tasks?							
	Very Low	1	2	3	4	5	Very High
2. **Physical demand:** How physically demanding were the tasks?							
	Very Low	1	2	3	4	5	Very High
3. **Temporal demand:** How hurried or rushed was the pace of the tasks?							
	Very Low	1	2	3	4	5	Very High
4. **Performance:** How successful were you in accomplishing what you were asked to do?							
	Very Low	1	2	3	4	5	Very High
5. **Effort:** How hard did you have to work to accomplish your level of performance?							
	Very Low	1	2	3	4	5	Very High
6. **Frustration:** How insecure, discouraged, irritated, stressed and annoyed were you?							
	Very Low	1	2	3	4	5	Very High

B. Usability (System Usability Scale)							
1. I think that I would like to use this system frequently.							
	Strongly disagree	1	2	3	4	5	Strongly agree
2. I found this system unnecessarily complex							
	Strongly disagree	1	2	3	4	5	Strongly agree
3. I thought this system was easy to use							
	Strongly disagree	1	2	3	4	5	Strongly agree
4. I think that I would need assistance to be able to use this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
5. I found the various functions in this system were well integrated							
	Strongly disagree	1	2	3	4	5	Strongly agree
6. I thought there was too much inconsistency in this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
7. I would imagine that most people would learn to use this system very quickly							
	Strongly disagree	1	2	3	4	5	Strongly agree
8. I found this system very cumbersome/awkward to use							
	Strongly disagree	1	2	3	4	5	Strongly agree
9. I felt very confident using this system							
	Strongly disagree	1	2	3	4	5	Strongly agree
10. I needed to learn a lot of things before I could get going with this system							
	Strongly disagree	1	2	3	4	5	Strongly agree

C. DISTRACTION							
1. I did not feel distracted while making calls with the system							
	Strongly disagree	1	2	3	4	5	Strongly agree
2. I did not feel distracted while sending text messages with the system							

	Strongly disagree	1	2	3	4	5	Strongly agree
3. I did not feel distracted while receiving text messages with the system							
	Strongly disagree	1	2	3	4	5	Strongly agree
4. I did not feel distracted while receiving calls with the system							
	Strongly disagree	1	2	3	4	5	Strongly agree

D. ADAPTATIVITY							
1. I felt that this system took my context into consideration when receiving calls							
	Strongly disagree	1	2	3	4	5	Strongly agree
2. I felt that this system took my context into consideration when receiving text messages							
	Strongly disagree	1	2	3	4	5	Strongly agree
3. I felt that this system took the Peer's context into consideration when sending text messages or making calls							
	Strongly disagree	1	2	3	4	5	Strongly agree

E. General
1. Identify the **most positive** aspect of the system
2. Identify the **most negative** aspect of the system

3. Please provide any **general comments or suggestions** for improvement

Appendix D: Comparing the Adaptive version and the Non-Adaptive version

Comparing Trip 1 with Trip 2

1	In which trip did you feel less distracted while receiving a call?					
Trip 1	1	2	3	4	5	**Trip 2**
2	In which trip did you feel less distracted while receiving a text message?					
Trip 1	1	2	3	4	5	**Trip 2**
3	In which trip did you feel less distracted while making a call?					
Trip 1	1	2	3	4	5	**Trip 2**
4	In which trip did you feel less distracted while sending a text message?					
Trip 1	1	2	3	4	5	**Trip 2**
General						
State one **positive** aspect of the application during **Trip 1**						
State one **negative** aspect of the application during **Trip 1**						

State one **positive** aspect of the application during **Trip 2**
State one **negative** aspect of the application during **Trip 2**

Appendix E: Ethics Clearance

Nelson Mandela Metropolitan University

for tomorrow

• PO Box 77000 • Nelson Mandela Metropolitan University
• Port Elizabeth • 6031 • South Africa • www.nmmu.ac.za

Chairperson: Research Ethics Committee (Human)
Tel: +27 (0)41 504-2235

Ref: [H12-SCI-CS-001/Approval]

RECH Secretariat: Mrs U Spies

16 September 2013

Prof J Wesson
Faculty of Science
09-02-13
South Campus

Dear Prof Wesson

A MODEL FRO MOBILE CONTEXT-AWARE IN CAR COMMUNICATION SYSTEMS TO REDUCE DRIVER DISTRACTION

PRP: Prof J Wesson
PI: Mr T Sielinou

Your above-entitled application for ethics approval served at the Research Ethics Committee (Human).

We take pleasure in informing you that the application was approved by the Committee.

The ethics clearance reference number is **H12-SCI-CS-001**, and is valid for three years. Please inform the REC-H, via your faculty representative, if any changes (particularly in the methodology) occur during this time. An annual affirmation to the effect that the protocols in use are still those for which approval was granted, will be required from you. You will be reminded timeously of this responsibility, and will receive the necessary documentation well in advance of any deadline.

We wish you well with the project. Please inform your co-investigators of the outcome, and convey our best wishes.

Yours sincerely

Prof CB Cilliers
Chairperson: Research Ethics Committee (Human)

cc: Department of Research Capacity Development
Faculty Officer: Science