Android In-Vehicle Infotainment System (AIVI)

Tarik Al-Ani

A thesis submitted for the degree

Master of Applied Science
in Software and Knowledge Engineering

University of Otago
Dunedin, New Zealand

June 2011

Supervised by:
Tony Savarimuthu
Maryam Purvis
Information Science Department
Abstract

Travelling long distances by cars in the modern world has become bearable with the availability of in-vehicle infotainment systems that can entertain and inform both drivers and other occupants. One of the problems faced by application developers for in-vehicle infotainment systems is the lack of an open and standard platform across different car manufacturers. This research investigates the suitability of an Android operating-system to be used as a standard platform for developing an in-vehicle infotainment system. This integrated system will be used as both a source of information and entertainment.

Our objective in this thesis is to build an integrated system that provides several infotainment services to the user such as playing music, GPS-based navigation services, road and traffic information and many more services available from third parties. This research thus aims to demonstrate how the Android system can be used as a stepping stone towards establishing an open-standard in the automotive in-vehicle infotainment industry. We believe a system such as ours will help reduce the closed, proprietary and non-extensible systems being released today and bolster the design and development of open, more complete, feature-full systems in vehicles that will not only assist but also entertain drivers and passengers.

This research consisted of three main phases. Firstly, the requirements of the system were obtained through a user survey. Secondly, an Android-based system was integrated based on the requirements specification. The integration effort required several software components to be developed and tested. Thirdly, a user opinion survey was conducted based on the features developed in our Android-based In-vehicle Infotainment (AIVI) system.
Acknowledgements

I would like to thank all who made this thesis possible, thank you for all the support and encouragement (from all my family and friends) that kept me going in the toughest times. My appreciations go to my supervisors, Tony Savarimuthu and Maryam Purvis, who have supported and guided me throughout this research. Thank you to all the attendees of the research presentations and survey respondents. My love goes out to my family and friends who kept me going during the late nights of work.

I would like to dedicate this thesis to my parents, Thamir and Iman Alani for supporting me throughout the toughest years of my academic life. Thank you for believing in me, and for always having faith in my studies.
# Table of Contents

Abstract ........................................................................................................................................... 2
Acknowledgements ....................................................................................................................... 3
Table of Contents .......................................................................................................................... 4
Table of Figures .............................................................................................................................. 7

Chapter 1: Introduction .................................................................................................................. 10
  1.1. Overview ............................................................................................................................. 10
  1.2. Background ......................................................................................................................... 10
  1.3. Problem Areas .................................................................................................................... 12
  1.4. Proposed Solution ............................................................................................................... 13
  1.5. Summary ............................................................................................................................ 15

Chapter 2: Current Research ......................................................................................................... 16
  2.1. Introduction ......................................................................................................................... 16
  2.2. Proprietary Systems ........................................................................................................... 17
    2.2.1. Microsoft Embedded Automotive Platform (Microsoft Auto) .................................. 17
    2.2.2. Ford SYNC................................................................................................................. 18
    2.2.3. Kia UVO and Blue&Me ........................................................................................... 19
    2.2.4. Issues ......................................................................................................................... 20
  2.3. Aftermarket IVI systems ...................................................................................................... 21
    2.3.1. Pioneer Audio Visual (AVIC-Z110BT) ..................................................................... 21
    2.3.2. Generic in-vehicle audio and infotainment systems ................................................. 22
    2.3.3. Issues concerning aftermarket systems .................................................................... 22
  2.4. Open-source Platforms ....................................................................................................... 23
    2.4.1. MeeGo - In-Vehicle Infotainment (IVI) ................................................................. 23
  2.5. Overview of the requirements of modern IVI systems ...................................................... 25
    2.5.1. Applications and Functionalities .............................................................................. 25
    2.5.2. Standard open-source platform ............................................................................... 26
    2.5.3. Connectivity in a vehicle ......................................................................................... 27
    2.5.4. Current demands for IVI systems ............................................................................ 27
  2.6. Summary ............................................................................................................................. 28

Chapter 3: Requirements of the Android-based IVI System .......................................................... 29
  3.1. Overview ............................................................................................................................. 29
  3.2. Android - a Proposed Solution .......................................................................................... 30
    3.2.1. Android Architecture .............................................................................................. 30
    3.2.2. Android Features (and their importance for IVI systems) ....................................... 32
  3.3. AIVI: comparison to current IVI systems .......................................................................... 35
    3.3.1. Multimedia ................................................................................................................. 35
    3.3.2. Navigation .................................................................................................................. 37
    3.3.3. Voice .......................................................................................................................... 39
    3.3.4. Display ....................................................................................................................... 40
6.5.4. Device Synchronisation

6.6. Summary

Appendix A: IVI System Features
Appendix B: Opinion Survey for Gathering Requirements
Appendix C: Comparison Matrix
Appendix D: Embedded Development
Appendix E: Opinion Survey Results based on Gathered Requirements
Appendix F: GPS Development
Appendix G: 3G Module Development
Appendix H: System Evaluation Survey
Appendix I: System Evaluation Results
Appendix J: Code Repository
Bibliography
Table of Figures

Figure 1 - An example of an IVI system in a car. ................................................................. 11
Figure 2 - Voice Recognition and Turn-By-Turn Navigation (Google Inc. 2010). .................. 14
Figure 3 - Pioneer Z110BT running MSN Direct ................................................................. 21
Figure 4 - Android O.S Architecture (Xuguang, 2009) ......................................................... 30
Figure 5 - Opinion Survey Chart: Desirability of ‘Audio Player’ ........................................... 44
Figure 6 - Opinion Survey Chart: Desirability of ‘Video Player’ .......................................... 45
Figure 7 - Opinion Survey Chart: Desirability of ‘GPS Navigation’ ...................................... 46
Figure 8 - Opinion Survey Chart: Desirability of ‘Reverse Camera’ .................................... 47
Figure 9 - Opinion Survey Chart: Desirability of ‘Voice Actions’ ......................................... 48
Figure 10 - Opinion Survey Chart: Desirability of ‘Email’ ..................................................... 49
Figure 11 - Opinion Survey Chart: Desirability of ‘Internet Browser’ ................................... 50
Figure 12 - Opinion Survey Chart: Desirability of ‘Installing Apps’ .................................... 51
Figure 13 - Opinion Survey Chart: Desirability of ‘Wi-Fi Hotspot’ ...................................... 52
Table 14 - Response Weighting ................. ................................................................. 53
Table 15 - Desirability Score of IVI Features .......................................................... 53
Figure 16 - Desirability Ranking of IVI Features ............................................................... 54
Figure 17 - User Interface for In-Vehicle Use ............................................................... 54
Figure 18 – Opinion Survey Chart: Mobile Operating-System Experience ....................... 55
Figure 19 – Opinion Survey Chart: Downloading vs. Streaming ....................................... 56
Figure 20 - Opinion Survey Chart: Mobile Data Usage ..................................................... 57
Figure 21 - Waze on Android (Waze Mobile, 2010) ......................................................... 59
Figure 22 - AIVI Components Architecture Diagram......................................................... 66
Figure 23 - Bootstrap chain ................................................................................. 69
Figure 24 - Android GPS flow diagram ................................................................. 73
Figure 25 - 3G Modem: Vodem Stick Pro ....................................................................... 74
Figure 26: Logitech C120 .................................................................................. 76
Figure 27: Android Camera Sub-System (Android Open-Source Project 2010) ................... 76
Figure 28 - Reverse Camera: Sequence Diagram ......................................................... 78
Figure 29 - Proximity Sensor: Flow Diagram ................................................................. 80
Figure 30 - AIVI CarHome Application ................................................................. 81
Figure 31 - AIVI CarHome: Navigation Slider ............................................................ 82
Figure 32 - AIVI CarHome: Weather ........................................................................ 82
Figure 33 - Sample Code: Text to Speech used for weather information ....................... 83
Figure 34 - Road Information ........................................................................ 83
Figure 35 - Speed Limit Exceeded .................................................................. 84
Figure 36 - System Evaluation: Satisfaction of Multimedia Features ............................. 89
Figure 37 - System Evaluation: Satisfaction of Connectivity Features ............................. 90
Figure 38 - System Evaluation: Satisfaction of the 'Navigation' features ....................... 91
Figure 39 - System Evaluation: Satisfaction of Third-party Applications ....................... 92
Figure 40 - Vehicle rear view (head-turn) ................................................................... 93
Figure 41 – Reverse camera view ........................................................................ 93
Figure 42 - System Evaluation: Satisfaction of the ‘Reverse Camera’ ......................... 93
Figure 43 - System Evaluation: Overall satisfaction of the AIVI system .................... 94
Table 44 - Summary of poll results ............................................................ 95
Figure 45 - Extended AIVI Architecture Diagram ................................................... 104
Figure 46 - Bootstrap Order ........................................................................................................D-1
Figure 47 - 3G Modem: PPPD Script..........................................................................................G-2
Figure 48 - 3G Modem: Connect Script.....................................................................................G-3
Figure 49 - 3G Modem: Init command to start 3G services.......................................................G-3
Chapter 1

Introduction

1.1. Overview

This chapter will give an introduction to In-Vehicle Infotainment systems, identify the problems associated with current systems and propose a solution to resolve these issues.

This chapter will cover the following:

1. **Background** – A background overview of in-vehicle infotainment systems.
2. **Problem Area** – Current problems associated with systems presently available.
3. **Proposed solution** – A solution that will redress the problems covered.

This will provide a basic overview and understanding of the research being conducted and provide insightful information concerning the current issues and future direction of in-vehicle infotainment.

1.2. Background

1.2.1. An introduction to In-Vehicle Infotainment (IVI) systems

Infotainment, by definition, is the idea of broadcasting media which is intended to both entertain and inform. This originates from the idea of ‘blending’ entertainment and information (Oxford Dictionaries, 2010). An Infotainment System is a system which can provide a blend of information
and entertainment, usually using a screen to convey this information allowing users to input certain commands and retrieve informative information in different formats of media.

The first example of infotainment was the television, as it provided informative media for the viewers. But it is not only limited to that as more sophisticated infotainment systems are now much more interactive than the Television. Interactivity allows users to input action commands and get the necessary information required from the system. These devices can commonly be found in vehicles including motor-vehicles and aeroplanes to keep passengers informed as well as entertained.

Features of in-vehicle infotainment systems typically include a multimedia player, navigation system, the ability to read news, internet and as such provide necessary infotainment for the user. There is no exact standard that can define an infotainment system, or what it must implement. However, there are recurring features that appear throughout the current devices, labelled as in-car entertainment or infotainment systems on the market.

Some of the features present in in-vehicle infotainment systems are usually considered to be standard features that are expected in every in-vehicle infotainment system. These include multimedia music, video and radio players and also a GPS assisted navigation system. As technology progresses less important features are becoming normalised in in-vehicle infotainment systems. One of these emerging features is Bluetooth connectivity, where a cellular phone is connected to the system to provide hands free communication via Bluetooth interface (Gaoxing, 2007). Another important feature is the ability to mount content from external devices via a USB interface (or other external storage slot). Currently, with more demand for natural interaction and the mass-production of touch-screens the use of a touch capable screen is also becoming a more standard theme within the existing in-vehicle infotainment industry. Samsung leads the way in the mass-production of integrated AMOLED touch-screens panels from March 2010 bringing down costs to mobile touchscreens (Davies, 2010).

A common term used to refer to infotainment systems in automobile environments is ‘in-car computer’ or less formally ‘Carputer’. Although this project can be referred to as a Carputer, it is not exactly the case. A Carputer, according to definition, generally refers to a desktop computer (or laptop) that has been physically designed or modified to fit into an automobile (Datzov, 2006). This is typically interpreted as a homebrew system that is designed by an enthusiast. These are frequently used with standard operating systems that run on 32-bit processors, thus they differs from widespread infotainment systems currently available. In-vehicle infotainment systems at present use a mobile processor such as the ARM processor (Advanced RISC Machine). This is a reduced instruction set from the 32-bit processor that is normally available on your common personal computer. The ARM processor is also cost-effective and powerful enough to run all mobile application requirements today (Jaggar, 1997; Levy & Many, 2006).

See Chapter 4 for more information regarding ARM processors and hardware associated with in-vehicle infotainment systems.
1.2.2. Automotive World

Current infotainment systems are being used in wide range of areas, but more specifically we will look at how they are used in an automobile environment. Infotainment systems are either fitted with the vehicle during manufacturing, or fitted via an aftermarket supplier usually in the field of car stereos.

1.3. Problem Areas

There are certain problems associated with current IVI systems. These include:

1. Lack of a standardised system.
2. Lack of an open-source system.
3. Lack of extensibility features of current systems.
4. Lack of community and third-party support.

Lack of a standardised and open-source system:

One of the main problems for mobile software development was the pressure of mobile manufacturers pushing out products (hardware) without having the software ready for them. When a mobile-phone manufacturer such as Samsung, needs to release a device, a proprietary system is usually developed in-house for the device (i.e. hardware). A Nokia phone would need a Symbian operating-system to be used; when a phone’s hardware is complete, it is not necessary that the software will be ready as it is a proprietary product that could take a long time for an external company to develop and risk requirements not completely met. This in turn led to the development of the Android operating-system (Open Handset Alliance, 2008b).

This problem can once more occur when it comes to in-vehicle infotainment systems. Manufacturers can release a specific design with its own unique hardware and features although the software is not necessarily designed by the manufacturer. This leads again to issues regarding the ability to straightforwardly change the software and update when required.

Another important factor concerning an open-system to be used in car-stereos is the ability to use the same software across multiple lines of products produced by the same manufacturer. With this ability, the firmware can then easily be updated across all products, thereby removing the need to update each product separately.

These issues not only affect the manufacturer, but can also affect the end-user by way of standardisation of in-dash systems. An end-user is not guaranteed that required features by the user will be available from system to system. Problems can occur which include playing the correct format for music or videos, as well as device support for any external gadgets to be used. A further example is the system’s ability to give the end-user the choice to use an application of preference that they are familiar with for their specific requirements. This allows external organisations to release applications to consumers in their field of expertise, thus consumers may pick what applications they want. This is in contrast to manufacturers releasing predefined applications that the consumer is ‘stuck’ with for the duration of the device’s life-cycle.
Extensibility features and third-party support:

We see some of these issues arise when looking at the current status of this research into the field of In-Vehicle Infotainment. In following section, we consider in-vehicle infotainment systems such as the Microsoft Auto Platform (Ford SYNC, Kia UVO and Blue&Me devices), and the Pioneer infotainment system (FIAT Group Automobiles, 2007; Ford Motor Company, 2009; Pioneer, 2010). These devices lack a standard and extensible operating-system that can provide a rich framework with API’s for third party developers (Tutor, 2009).

Having an extensible, yet a non-established system might not be enough. It is important to have a well-established platform that provides an extensive application-base and an interface that the consumers are familiar with. The MeeGo IVI platform is the closest to achieving such requirements but it lacks the established nature of a standard operating-system (MeeGo, 2010b). With the additional lack of an application-base and support from vendors and affiliates the MeeGo does not seem mature enough to be used as a standard system for In-Vehicle Infotainment. This is discussed in detail in Chapter 2.

1.4. Proposed Solution

This research aims to demonstrate the feasibility of building an In Vehicle Infotainment system based on the well-known Android operating system.

Standard and Open-source solution:

The Android platform provides an open-source based solution that we use to aid the development of infotainment systems. This research will provide a proof-of-concept as it looks into developing an Android-based In-Vehicle-Infotainment System (AIVI) that is built on a Cortex-A8 ARM processor with a Texas Instrument OMAP3530 board. The system’s graphical interface will be portrayed via a 7-inch touch-capable screen.

The Android system will help provide all IVI capabilities that are deemed to be relevant to today’s requirements of an In-Vehicle Infotainment System. This will in turn, help scrutinize and potentially replace the requirements of the standard infotainment systems and stereo systems that are built-in to vehicle dashboards.

The Android operating-system (Android O.S.) is a mobile operating system which can be used in multiple devices, not only mobile phones. Using the Android O.S. for in-vehicle entertainment provides all the entertainment features offered by a top-of-the-range in-dash infotainment system with the addition of informative, driver-assisting content including; hands-free calling, multimedia centre, a navigation system and a range of applications which offer features required by the user. These are just some examples, for an open-source expandable system (whereby the framework can be extended and applications can be developed for it). The system can include features such as a reverse camera and other options which can be further developed.

Extensible System:

Currently, there is a need for developing an infotainment system which runs on an open-source platform and can be customised by users. For example, a taxi company using this system can design and develop their own custom-made application that can be deployed in all their vehicles to monitor their locations in real-time. Some examples include GPS tracking of vehicles in time-critical jobs where tracking is necessary. Others include development of applications to interact with the vehicle’s internal system such as; vehicle alarm systems, remote entry and vehicle start, as well as data analysis of the data provided by the vehicle’s computer. These are all possible with the ability to develop
applications using Android’s SDK (Java) and NDK (Lower-level Native development kit using C/C++ Programming), and the assistance of Android’s large online development community (Allen, Graupera, & Lundrigan, 2010).

Using an Android system can overcome the problems previously addressed, as it is an open-source platform. This means that the manufacturer can take the operating system and adapt it to suit specific hardware and software requirements. Additionally, new applications can be added and modified by the user of the system.

Overcoming these problems is important as it greatly reduces costs when it comes to maintaining the system and suitability updating it. The large community base of an open-source system means that the system is actively maintained and new features are constantly being developed (Open Handset Alliance, 2008a).

**Third-party support:**

The ability to use the Android Market place means that applications can be developed and used across different products from different manufacturers. This is important for end users as it brings a larger application base. This encourages manufacturers’ focus on building better hardware without worrying about the software (Open Handset Alliance, 2008b).

**Design and Interaction:**

The system will be placed into the standard dash area. This will take two standard decks, known as the double-DIN standard, which will be enough to fit a 7-inch touch screen display unit (Quilling, 2005). The user will interact predominantly with the screen in a natural manner, like how a user interacts with an Android touch-screen device. In addition to the touch interface, users can also take advantage of Android's intelligent speech recognition capabilities provided by Google services¹; this will allow hands-free interaction for enhanced safety while on the road.

Figure 2 demonstrates an android feature loaded with Android 2.0 known as Turn-By-Turn Google Maps which is a free Navigation service provided by Google (M Arrington, 2009).

---

¹ Note that users are not restricted to use any Google services, but these are the standard services and can use any alternative services by third parties.
1.5. Summary

This chapter covered all the introductory components to current in-vehicle infotainment systems and the issues being raised in these systems. This chapter also looked at the proposed solution to resolve such issues.

Current IVI Systems

Current In-Vehicle Infotainment systems all feature components that provide entertainment as well as informative information for vehicle occupants. This includes the ability to play music, radio and general audio from multiple formats. Infotainment systems also include a GPS Navigation system that enhances the driver’s experience. These are the basic components of IVI systems; most IVI systems further expand to provide unique features from other manufacturers.

Current Problems

Problems with existing systems include lack of standardisation and extensibility:

- **Standardisation**
  Standardisation is an issue at present in the current IVI market. There is no single standard platform that manufacturers are using. Consumers have to readapt to each system made by various manufacturers and there are no set standards as to what should be implemented in an IVI system. This also includes safety regulations.

- **Open-source**
  This issue relates to manufacturers having to redevelop systems and technologies that have been widely used in existing developed systems. Using the Android open-source platform means that hardware vendors need not put too much emphasis on creating the perfect software, but rather concentrate on perfecting already existing, widely used, software. This gives manufacturers more room to improve the hardware components.

- **Extensibility**
  Each manufacturer provides their systems which are proprietary, closed, and offer no support for extensibility. Using the Android operating-system offers API’s for developers to create custom applications that further expand the life-cycle of the product. This will grant devices with newer applications and features that can be installed on demand by users, and thus satisfy each user’s personal requirements.

Proposed Solution

Developing an Android-based IVI system will tackle all the problems covered above as it provides a standard, open-source, expandable system to users. Furthermore, it is widely used in the mobile industry and very common for consumers to recognise. This reduces the adoption rate for end-users and allows them to install the readily available 120,000+ applications provided by the Android Market.
Chapter 2

Current Research

2.1. Introduction

In-vehicle infotainment devices today are usually produced by vehicle manufacturers to be used in their own vehicles or by aftermarket device manufacturers, usually within the in-car entertainment industry. These infotainment systems are pre-installed before the purchase of the vehicle or can be offered as an ‘add-on’ for consumers to purchase and install at any time.

The software for these systems are developed by one of two methods; in-house using proprietary custom built software, or built on-top of an existing operating-system and modified accordingly to match requirements within the limits of the adopted operating-system.

In-house systems use custom software built according to the manufactures themselves; these systems are generally fixed and rarely updated. The system is fixed throughout its life-cycle and cannot be modified by end users apart from basic customisation that is permitted by the software developers. These systems are quite standard and can be seen in most in-vehicle infotainment at the moment in major vehicle brands such as BMW, Mercedes Benz, Audi and many other auto-makers that have infotainment systems pre-equipped in their vehicles (Koenigseger, Hock, & Kirk, 2003; Sulowski & Bock, 2003). These in-house built systems are also developed by major car-audio producers such as Pioneer, Sony, JVC, and many others which include the generic, low-priced, systems that appear principally on online e-commerce websites which are becoming more popular as demand is increasing.

Furthermore, the need for open-source systems has emerged over the recent months with systems such as MeeGo IVI arising. This comes from an increased demand for open-source systems within the IVI market where manufacturers feel that there is a need for an open-platform to be used, and accordingly customised to their requirements.
This research aims to build an integrated system that provides a centralised point of infotainment by incorporating several entertainment and informative services to enhance driver and passenger experience in the vehicle. Services include; music, GPS navigation services, road and traffic information and many more services provided by third-party vendors. This research aims to demonstrate how Android systems can be used as a standard platform for modern in-vehicle infotainment to allow the use of an open-source and extensible systems in vehicles that will not only assist but also entertain vehicle occupants.

This chapter will describe these approaches in detail and include some of the available platforms used in in-vehicle infotainment; these are broken up into the following:

- **Proprietary Systems**
  - Pre-installed Systems:
    - Microsoft Embedded Automotive Platform
    - Ford SYNC, Kia UVO, FIAT Blue&Me
  - Aftermarket Systems:
    - Pioneer in-vehicle audio system (Pioneer AVIC-Z110BT)
    - Generic IVI systems
- **Open-source Systems**
  - MeeGo IVI
  - Current Issues
- **Overview of the requirements of modern IVI systems**
- **Summary**

The research gathered from these systems will help us distinguish the current issues, and understand the requirements of building the proposed Android-based IVI system.

### 2.2. Proprietary Systems

Proprietary systems are closed-source and do not allow public access to the source-code. The system is built within a single manufacturer, and in the IVI industry, most systems present no real standard platform. This includes aftermarket systems that are developed by in-car entertainment companies (such as Pioneer). Microsoft Auto platform is the leading standardised platform in the infotainment industry, but this platform is not without its limitations.

#### 2.2.1. Microsoft Embedded Automotive Platform (Microsoft Auto)

Microsoft Embedded Automotive Platform, or more commonly known as Microsoft Auto, is a closed-source platform designed specifically for the purpose of In-Vehicle Infotainment. Microsoft Auto provides a platform for automakers to take, and build ‘on-top’, to design an infotainment system that is specifically tailored to their needs (Ghangurde & Business, 2010).

Microsoft Auto provides all the necessary backbone for the platform including multimedia synchronisation capabilities, speech recognition, communication and multimedia libraries which offer the basic infrastructure for an infotainment system. Automotive manufacturers take this platform and
redesign the look and feel while adding specific functionalities relevant to the automakers direction and strategy.

Microsoft Auto implementations differ from vehicle to vehicle even within the same vehicle manufacturer. Some of the outlined features of the Microsoft Auto platform include:

**Multimedia and Radio**

This is, as expected, a standard feature of an infotainment system today. This includes the support for multiple multimedia formats including audio and video. This is further broken down to multiple supports for different encoding formats as well as support for AM/FM radio.

**Synchronisation**

Microsoft Auto allows synchronisation with many portable devices via various interfaces including USB and Bluetooth. These include popular portable devices such as the iPod, Zune, and other assorted MP3 players and cellular phones. This synchronisation process allows the sharing of media from various devices and provides the automobile's sound system as a central source for the media to be used with. This also allows the integration of the cellular phone’s address book which can be utilised for hands free voice calling.

**Voice Recognition**

Microsoft Auto features applications that keep driver focus on the road by minimizing the distractions of operating the device itself which could lead to accidents.

These features include;
- Instant voice recognition
- Voice activated calling
- Voice activated music player
- SMS messages are read back to you

The voice activation features are usually activated by a button, which is appropriately placed on the steering wheel in order to minimise driver distractions and keep focus on road safety. These buttons are implemented accordingly to the car manufacturer based on their decisions; each product varies from different vehicles to different manufacturers as mentioned previously.

Current implementations of the Microsoft Auto are currently used in the following products:

- Ford SYNC - *Ford Motor Company*
- Kia UVO - *Kia Motors*
- Blue&Me - *Fiat Group*

### 2.2.2. Ford SYNC

Ford SYNC, powered by Microsoft Auto, is a proprietary closed-source system designed by Ford and intended to be used in Ford vehicles produced from 2008 onwards (Piotrowski, 24 January 2007). Ford Sync is powered by an ARM11 mobile processor which includes 256MB RAM and 2GB Flash Memory. The Ford Sync has an extensive feature-base which enables the synchronisation of portable
devices with the infotainment system to allow a central entertainment source as well as an informative system that provides driver enhancements.

The system also has support for GPS navigation, including a Traffic Alert service as well as Turn by Turn. These services are only free for the first three years of the vehicle, and then they can be purchased by a monthly or yearly subscription from the service provider.

Since this system is designed by Ford, its design is compatible with the vehicle’s Engine Control Unit (ECU) in retrieving certain data to display for the user. This ability enables the provision of two important features; Vehicle Health Reports and the 911 Assist (Ford Motor Company, 2009).

- **Vehicle Health Reports**

  This feature allows the user to view vehicle diagnostics in an appropriate and humanly comprehensible manner enabling luxury car features to be readily provisioned in SYNC-enabled vehicles. This is in comparison to the vague blinking check engine light shown in most cars available today. This will allow car owners to better look after their vehicles as they are always aware of vehicle diagnostics (counting; malfunctioning car parts and devices, abnormal readings etc.).

- **911 Assist**

  Having access to the vehicle ECU allows the system to be aware of certain situations that have been triggered by the vehicle itself. The vehicle airbags, a safety feature, are deployed once a vehicle is involved in an accident exceeding a specified force to help protect the driver and passengers of the vehicle. These airbags inflate rapidly to stop a person from making physical contact with the vehicle’s interiors that may result in injury or death. Ford Sync can detect when an airbag has been deployed, then places an emergency call which will play a pre-recorded message, supply GPS coordinates, and allow the driver to speak to the emergency correspondents in order to arrange further assistance if required.

  Apart from the latest releases of the Ford SYNC system (MyFord Touch), there is typically no full-colour screen display but rather a regular, stereo style, small sized LCD screen. Features vary from vehicle to vehicle; not all Ford SYNC capable vehicles will have navigation or video capabilities as they lack the visual display that is necessary (characteristically a guided map overlaid with indicators for direction and points of interest).

### 2.2.3. **Kia UVO and Blue&Me**

Ford SYNC is not the only product to have been produced from the Microsoft Auto platform. Other products include Kia UVO and Blue&Me. These products also implement the Microsoft Auto platform for in-vehicle infotainment systems.

**Kia UVO – IVI system for Kia vehicles.**

The Kia UVO system is developed by Kia Motors and also powered by the Microsoft Auto platform. Kia UVO is equipped with a large display touch-screen, usually 4.3 inches. The Kia UVO delivers no real difference in regards to functionality offered by Microsoft Auto, except the ‘Reverse Camera’ feature. The reversing camera, also known as a back-up camera, is a camera that is placed around the rear bumper area (Microsoft, 2010). This provides the users with a view of the rear of the
vehicle providing vision to any obscurity that is limited by the rear-view mirror. The images captured by the reverse camera are displayed on the LCD screen for the user in real-time.

**Blue&Me – IVI System for FIAT vehicles.**

Blue&Me is FIAT’s and Microsoft joint implementation of the Microsoft Auto platform. This product provides all the features of the Microsoft Auto platform with the ability to add different types of mobile devices.

The EcoDrive feature, which is FIAT’s unique enhancement to the infotainment system, offers users the ability to determine the driving efficiency of the user. This collects any necessary data from the vehicle’s ECU in regards to vehicle emissions. This data is then analysed and presented as a report for the user to easily understand (FIAT Group Automobiles, 2007).

### 2.2.4. Issues

The Microsoft Embedded Automotive Platform provides an effective, but not complete, platform for In-Vehicle Infotainment. Although it delivers the majority of the necessary features and requirements of an infotainment system, it brings up a few concerns;

- **Closed-source proprietary system.**

  Being a standard platform means that you can take the system, make minimum changes necessary and be able to deliver it out with minimal costs, as the system should be ready and the software is thoroughly tested. This is in-comparison to *in-house* built systems that are developed within the company, or where there is complete outsourcing of a system where a third party will build your system from scratch tailored to the vendor’s specifications. With the Microsoft Auto platform, car manufacturers have no access to the ‘source-code’ or the ability to modify the system beyond the limited framework API’s available. Car manufacturers cannot make any changes to the framework or to the kernel. Developers will have to work with the limitations of the Platform Development Kit (PDK) and cannot make any changes to drivers if any hardware incompatibilities arise (ECU connection, Bluetooth module, Camera type etc.). This also might limit any framework changes, for example if there is a need to change the frequency ranges of FM radio for different parts of the world.

  This raises the issue where if there are any changes, which cannot be achieved in the PDK, the manufacturer must work back with Microsoft to accomplish them. This becomes a major issue as it increases the production time and does not give any advantages of using a standard platform, as is the case with Microsoft Auto. Every implementation of the Microsoft Auto platform is a joint operation of Microsoft and the car manufacturer, which is not conducive for either party.

- **No third-party Applications**

  Another concern is that there is a lack of API (Application programming interface). No API means that there is no capability of installing external applications acquired from third party. Any new functionality must be released by the car manufacturer themselves as the system varies from

---

2 In computing, the kernel is the central component of most computer operating systems; it is a bridge between applications and the actual data processing done at the hardware level.

3 Microsoft Auto PDK references are not available publicly, only to their customers and partners.
manufacturer to manufacturer, and it is nearly impossible to obtain an API for the Microsoft Auto platform otherwise. This means there is no interoperability between Microsoft Auto products (according to Microsoft Auto 4.1). This proves to be problematic as it doesn’t allow third parties to produce their product of expertise. For example, TomTom or Garmin cannot release its own product for GPS navigation which brings a much larger feature base in-comparison to the standard navigation application that is pushed out by the car-manufacturer. These third party products are also constantly updated as each party concentrates on its own field of expertise to improve the device as a whole.

These issues all relate to the fact that the Microsoft Auto platform is proprietary and no changes can be made internally and externally. Furthermore, the Microsoft Auto platform is not extensible by any other party apart from the manufacturers themselves (Microsoft and the car manufacturer) and the device cannot be installed as an after-market product off-the-shelf. It must be sold for that specific make, and generally for that specific car model\(^4\).

### 2.3. Aftermarket IVI systems

After-market products are items which are bought off-the-shelf and can often be installed by a customer of minimal installation experience. In the aftermarket In-Vehicle Infotainment (IVI) industry there are a few products which have certainly met the standard of a proper In-Vehicle Infotainment system.

#### 2.3.1. Pioneer Audio Visual (AVIC-Z110BT)

Pioneer is one of the leading innovators of after-market stereo systems. Their latest products are equipped with all of the features necessary for infotainment systems. The model analysed here is the Pioneer AVIC-Z110BT. This is a high-end Pioneer stereo system which included the following features; Voice Control, Bluetooth connectivity, Reverse Camera (capable\(^5\)), USB Interface, GPS Navigation and a multimedia system for video, audio and radio (Pioneer, 2010).

The pioneer audio-visual system is providing the MSN direct service built-in to the devices to provide a more complete infotainment solution. MSN Direct\(^6\) is a FM-based digital service that gives traffic notifications, weather information, news headlines, gas prices and other services. This is a subscription based service and with the Pioneer stereo system, you receive a two-year subscription upon activation (Price, 2008). This service is to be discontinued as of January 2012 (announced by Microsoft) (MSN Direct, 2010).

---

\(^4\) Limited to Ford, Kia, FIAT (as of December 2010).

\(^5\) Pioneer Z110BT is capable of the reverse camera functionality, but is not included.

\(^6\) MSN Direct service is only available in the United States of America.
2.3.2. **Generic in-vehicle audio and infotainment systems**

These systems are ordinarily generic brand products designed by independent manufacturers for small businesses to resell to the general market (Jiangmen Xinhui Goodbee Technology Co., 2003). These systems are mass-produced for sale and branded according to the reseller’s specifications. Most of these systems are sold with the same software included which has been slightly modified to contain, or omit, any components that have been specified according to the reseller’s specified requirements. These also consists of the redesign of the logos and basic aesthetics of the software and hardware components, but the underlying software system is usually the same. Examples of these systems can be seen on e-commerce sites like TradeMe⁷.

2.3.3. **Issues concerning aftermarket systems**

These aftermarket solutions are commonly feature-packed and provide users with the most necessary features required for in-vehicle infotainment. This is noteworthy because these infotainment systems are an after-market installable solution, rather than being pre-installed by the vehicle manufacturer. These can be set up in most vehicles as long as the appropriate fitting requirements are met. The *fitting* requirements is customarily a double-DIN standard, compatible with most vehicles on the road today (Quilling, 2009).

The main issues concerning Pioneer’s approach for In-Vehicle infotainment in the Pioneer Z110BT is that the system is custom; a non-standard platform proprietarily designed by Pioneer. This means all applications are built by Pioneer themselves with exceptions, such as the integration of MSN Direct. This makes the product in no way extensible as it does not reside on a standard platform or operating system.

This is a recurring issue with the generic after-market systems as they use which ever software has been developed by the device manufacturers themselves. This software is closed-source and has no real customisation from lack of any extensibility support.

These systems do not provide any API’s or developer support for external and third-party applications. The issue that arises here is relevant to Microsoft Auto where third-party developers cannot provide their products to the public, and Pioneer re-inventing the wheel, so to speak, with aspects such as navigation capabilities.

Lastly, in cases concerning integration, it is not dealt with well. This applies to the MSN Direct feature. This feature is extraneous as it is and not integrated into the device’s navigation application as it only provides generic feeds with no real understanding of where a traffic issue is, or if it is relevant to your location (See Figure 3). This shows a degree of separation for features that are implemented by different developers.

2.4. Open-source Platforms

Open-source systems provide developers with an open-platform, publicly listed, source-code to be used, modified and customised for device vendors to use on the desired hardware. At this time, the MeeGo system is the only emerging system in the IVI field that is considered to be open-source.

2.4.1. MeeGo - In-Vehicle Infotainment (IVI)

MeeGo, previously known as Maemo, is a Linux-based platform that is open-source. This is a first real attempt at an open-source platform that is made for multiple devices. This means that the MeeGo platform can be taken by device vendors and used on their own hardware so as long as it is running an ARMv7 processor or an Intel Atom processor (Schroeder, 2010).

MeeGo will be able to run on multiple devices such as In-Vehicle Infotainment, tablets, media phones, mobile phones, connected TVs, and netbooks. MeeGo provides framework and open-source libraries that are readily available and interfaced via MeeGo’s comprehensive API.

MeeGo IVI is a subproject by the MeeGo group for development of in-vehicle infotainment. MeeGo IVI is led by the GENIVI Alliance, an affiliation of Nokia and Intel with support from multiple partners such as BMW, Mini Cooper, and Alpine (GENIVI Alliance, 2010; Meyer & Ouazzani, 2010).

The following are some important features that the MeeGo platform makes available:

Open-source Mobile Operating System

Essentially, MeeGo is a light-weight mobile operating system that is capable of running on mobile processors. Like the Microsoft Auto platform, it is a complete platform with standard libraries for multimedia and informative aspects. The advantage of MeeGo using standard libraries, such as; BlueZ for the Linux Bluetooth stack, and WebKit (layout engine for web-content) is that they are well supported, updated and developers are familiar with the API provided for these libraries.

One major difference with MeeGo as an open-source platform is that any device vendor can take this platform, change and modify it to suit the hardware or software requirements for a specific manufactured device, and release it. This was the major concern in regards to the Microsoft Auto platform as it did not allow any modification to the source-code, being proprietary to Microsoft as the platform developer. Changes required by device vendors would have to go back to the platform developers themselves as previously discussed.

Qt framework

Qt (pronounced ‘cute’) is a cross-platform application and UI framework designed by Nokia to allow developers to develop one application for multiple platforms (Qt community & Nokia Corp, 1992). Using the Qt framework, developers can create one application that is compatible with any Qt supported platforms, including; MeeGo, Windows Mobile/CE, Symbian and Linux Embedded. This also has support for desktop platforms such as; Windows, Linux/X11, Mac OSX. There are also external ports of the Qt framework that allows support for Android phones as well as iPhones (iOS).

The Qt framework also allows developers to use their programming language of choice from the supported language list. Some languages include; PHP, Java, Ruby, JavaScript, C++, C#, Ruby,
Pascal, Python, Perl, among many others. More languages are being supported as the framework improves.

This platform proves to be vital for software development as it allows software developers to implement only one version of their software and run it on multiple platforms without the need to recode the application for each platform, according to their own framework and programming language preference.

**Issues**

Currently, MeeGo is in the early phases of development. This means that there are multiple issues concerning the practicality of using the MeeGo system today. The issues are discussed below.

- **Lack of Application-base and Application market/store**

  At present, there are no applications developed for MeeGo that are released to the community (apart from the sample applications developed by MeeGo developers for demoing purposes). Consequently, there are only a handful of applications that are used to demonstrate the basic functionality of the MeeGo framework and API. There is a lack of voice recognition capabilities or powerful applications to give it any competitive advantages over current infotainment systems like the Microsoft Auto platform and Pioneer system.

- **Lack of Hardware and Limited Hardware Requirements**

  There are presently no vendor releases of the MeeGo platform. Consequently, there is no hardware that is commercially available, tested, and being used today on any devices (including handsets and other supported MeeGo devices). Nokia is the leading developer of the MeeGo platform, yet there seems to be no devices that have been released from any manufacturer with MeeGo software.

  The development kits for developers are costly, but these kits are necessary to build any MeeGo based device and provide the basic prototype for a MeeGo system. Support for MeeGo seems to be limited to Intel boards (as they are a main affiliate)\(^8\) (MeeGo, 2010a).

  These issues do relate to the early phase of MeeGo development but it is considered mature enough to be labelled with a version post beta (MeeGo IVI 1.1) and seen as a “solid baseline for device vendors and developers” as quoted in the official MeeGo website (MeeGo, 2010b). MeeGo, in our opinion is not ready for release on the public market due to still being unstable and only limited support.

---

\(^8\) MeeGo IVI platform only has support for SSSE3 enabled CPUs. These are mostly Intel processors (Core 2, Atom and other similar Intel processors). Currently there is no ARM Support for MeeGo IVI.
2.5. Overview of the requirements of modern IVI systems

All in-vehicle infotainment systems have some basic feature requirements which become more apparent when comparing different devices. These features include;

1. Multimedia support:
   - Basic audio support; music, radio, and video support.
   - Synchronisation with previously existing media playing devices.

2. Navigation support:
   - Ordinarily through a graphical map.
   - Can include updated en-route traffic information.

3. Basic safety features:
   - Can include options to deactivate features that could potentially distract drivers, such as video. These are deactivated when the vehicle is in motion.
   - Can include reverse camera support to view obstructed areas behind the vehicle.
   - In more advanced systems; Voice activated commands or speech feedback to reduce the driver’s physical interactions and improve road safety.

Many in-vehicle infotainment systems provide a wide range of functionalities and features; we identify three types of stakeholders in an in-vehicle infotainment system. They are given below.

1. **Device manufacturers and developers** – These are the manufacturers and the developers of the IVI system, including auto-makers who provide IVI systems preinstalled into their vehicles, or aftermarket manufacturers that offer IVI systems as installable add-ons for vehicles.

2. **Application developers** – Usually external, third-party developers that provide additional applications to already existing systems. Applications developed by third-party developers are customarily accessible via an application-base for that specific platform.

3. **End-users** – The user-level consumers who will be utilising IVI devices for their personal or business interests.

2.5.1. Applications and Functionalities

Applications are actually the core-functionalities of an IVI system. This is because applications provide the necessary features for an IVI system. For example; a multimedia application will provide the Music feature, in addition to the video and audio. The case is the same case with GPS Navigation; a Navigation application will provide the navigation features of the device. This is why it is important to have an extensible application-base, as it in fact provides a large feature-base for the IVI system (assuming the required hardware is available such as a GPS receiver for a navigation application).

An application-base refers to a software-distribution mechanism (usually known as a software store) that allows end-users to browse through a vast database and install external applications on their devices. It also allows developers to submit their own applications to be sold, or distributed for free. Some examples of an application-base include; Apple’s AppStore (for iOS platform; iPhone, iPod
Touch, iPad), or the Android Market (for Android O.S. devices) (Chu, 2008; Zdziarski, 2009). Generally these software-stores let users submit reviews and ratings for applications to identify which ones are less useful or non-functional.

Issues, as described in the previous sections, include the lack of an extensible application-base that allows consumers to use existing applications readily available on their mobile device in their IVI system. Also the ability to use the same applications across multiple IVI devices removes the isolation factor, allowing end-users accessibility to all features from different IVI devices in different vehicles.

It is evident from the research gathered (see section; 2.2.1), a few infotainment systems provide unique features with some sort of competitive advantage that uniquely improve the user experience. Some of these features include Ford SYNC’s 911 assist, which contacts local emergency services when the deployment of airbags have been detected (as previously described). Another feature is Blue&Me’s EcoDrive tool (by FIAT); this is a driving behaviour analysis tool for drivers to visually see their driving behaviour according to vehicle emissions and other data recorded and analysed by the software. Both of these products use the same Microsoft Auto platform but each application is unique to the auto-maker’s own device and not available to the other.

Application API’s and an application-base give device manufacturers the assurance that there will be a large application-base with developers constantly providing applications for their platform. This will shorten the manufacturing life-cycle of the device as manufacturers will have reliable applications that have been well tested (some providing appropriate feedback). This will reduce the time for development, enabling manufacturers to focus only on the hardware components of the system (screen, input devices etc.).

Application developers will provide the same applications for multiple devices, normally without modifications.9 Likewise, most applications will be able to run on the newly developed IVI. Furthermore, newer applications more relevant to the IVI context will likely emerge.

End-users will be able to take advantage of an extensible platform as they can add or remove any relevant application. All users have different requirements; they search through the application-base to find the appropriate application for their specific purpose. Examples are; keeping track of milage for company purposes (Nygren, 2010), and finding friends by ‘social’ navigation (Waze Mobile, 2010) These applications are readily available and can be downloaded when using a platform such as the Android operating-system.

2.5.2. Standard open-source platform

Using a standard open-source platform is important for manufacturers, application developers, and end-users for the following reasons:

An open-source platform allows device manufacturers to take the platform and modify it appropriately to suit the hardware prerequisites required for the development of that specific device (this is more commonly known as ‘porting’). Porting an open-source platform allows the platform to operate on multiple types of devices, but keep the same framework to allow applications to function accordingly across all types of platforms.

It also gives application developers the opportunity to create one application across multiple devices, providing they use the same platform. When using an existing platform, the same applications can be used, which are readily available, on the new devices without any change to the

---

9 Some applications might require different ‘views’ to be implemented that allow support for a larger (or smaller) screen. This modification typically does not involve changes to the model (core-code) of the application.
source-code. This gives the newly developed in-vehicle infotainment system an existing application base which can be further extended. This is made possible by standard API’s that are provided by the platform.

A standard platform gives end-users a comfortable, easy to use system that they are familiar with. This minimises time to familiarise with the system’s graphical user interface (GUI) and basic navigation and interaction with the system. Users will be able to utilise an already well supported platform that has been released previously, anticipating the strong possibility that the user would have experienced that platform on a different device.

2.5.3. Connectivity in a vehicle

At the moment, a more commonly demanded feature is device connectivity. A connected device will be able to communicate with the internet downloading as well as uploading relevant data to provide users with necessary information that can help assist users accomplishing everyday tasks efficiently.

None of the presently available commercial IVI systems provide us with a connected device that can further expand the functionalities of the system. A connected device will be able to communicate with web applications and services. This includes retrieving live weather information, traffic reports and other data from existing web-services (including business dashboards).

According to a market research study from Invensity (Wiesbaden, Germany) it is expected that from 2013, every new car built in Europe will be equipped with an internet connection from the factory (Hammerschmidt, 2008). This study also shows that there is more demand for internet connectivity in everyday vehicles, rather than being restricted to high-end luxury vehicles (as suggested by 50% of the experts surveyed).

2.5.4. Current demands for IVI systems

The market research study from Invensity (Wiesbaden, Germany) notes that the strongest need for development in infotainment system is the ‘need for standardisation’. 74% of experts polled believe that the industry has failed to sufficiently drive and develop the infotainment standards. Furthermore in car-to-car communication Standards are not developed far enough (63% believed that this is the case) (Hammerschmidt, 2008).

It is also believed (by 90% of the experts polled) that the interior electronic development will be most impacted by the development of in-vehicle infotainment, so it is expected that as infotainment systems mature more of the vehicle electronics will be controlled by a central in-vehicle infotainment system. These interior electronics include the lightning system, sound system, air-conditioning, and vehicle alarm system.

These numbers show that a standard, open-source and connected system is a must for the development of in-vehicle infotainment to meet the increased demands for improved technologies and better standards for in-vehicle infotainment.
2.6. Summary

In summary, current in-vehicle infotainment systems are developed by two methods, custom built systems by manufacturers or by extending on a standard mobile\textsuperscript{10} based operating-system; such as an open-source Linux platform or a closed-source Microsoft Auto platform. These are further broken up into aftermarket devices or ones pre-installed by the vehicle manufacturer.

Current IVI systems have a common feature-base which is regarded as standard for current IVI systems, these include; navigation, entertainment and basic safety support. The safety support complements the IVI features to provide safer interaction with minimal distractions to drivers.

There is currently a higher demand from IVI systems for, connectivity, third-party applications and open-platforms. These components let consumers customise their system to their desired needs and requirements, meanwhile an open-platform will also assist device manufacturers in releasing systems which are tested, readily available and have an established application-base for consumers to use (by downloading and installing these applications).

\textsuperscript{10} Mobile operating-systems refer to component reduced light-weight operating systems optimised to run on a mobile-platform such as ARM or Intel Atom architectures.
Chapter 3

Requirements of the Android-based IVI System

3.1. Overview

This chapter will look into the use of Android operating-system as a proposed solution for in-vehicle infotainment (IVI). Android O.S. will be used and compared with a current IVI system available for IVIs based on certain criteria. These systems include; Ford SYNC, Kia UVO, Blue&Me, MeeGo system, and a top of the range Pioneer in-car audio system (AVIC-Z110BT). These systems are briefly described in Chapter 2 and a proper analysis will be conducted in comparison to the Android system.

Furthermore, based on an opinion survey that was conducted, we present what actual end-users and consumers want out of infotainment systems. In particular we will identify what features they would like to see “in-vehicle” and also the relative ranking of the features (i.e. the importance of the feature) with respect to their personal requirements. Then we also present our survey results on how people are using mobile gadgets, their experience with web-applications and current mobile platforms and what features they expect to see. This data is collected from an opinion survey that was distributed with the resulting data effectively analysed.

This chapter also investigates the practicality, availability and costs of present mobile internet connectivity. Finally, an analysis of how the results from the survey impact the proposed system is presented.

The survey thus has provided us with all the necessary knowledge and understanding into building an effective in-vehicle infotainment effectively, enabling us to prioritise the requirements that we should focus on and understand what consumers want for their infotainment systems.
3.2. Android - a Proposed Solution

This section will cover an overview of Android O.S. and outline why it has been chosen for in-vehicle infotainment over current IVI systems on the market.

3.2.1. Android Architecture

Android is a complete mobile operating-system. It provides all the features necessary of a mobile operating system including more demanding features such as multi-tasking, speech recognition and synthesis, the connectivity features and a 3D graphics engine.

Android also provides a wide range of libraries which are accessible via an extensive framework. This framework is used for the API’s needed to develop third-party/external applications by any developer interested in Android development. This is made easier as Android provides all the tools necessary via a software development kit (SDK).

The Android architecture can be seen in the following diagram (Figure 4):

![Android O.S Architecture](image)

Figure 4 - Android O.S Architecture (Xuguang, 2009).

We will discuss each component separately to more thoroughly understand how the platform works. We will start from the core Linux Kernel and move up to the Hardware Abstraction Layer (HAL), the Libraries and runtime engine (Dalvik). Further on we will also discuss the Application Framework and how applications use these components.
**Linux Kernel**

A kernel is a central component in an operating-system. This provides the lowest level of abstraction for communication with hardware. In addition, the kernel manages system resources such as process, memory and device management.

The Linux kernel is a well-established kernel that is actively developed and is used for range of different distributions. For example GTK+, GNOME, KDE, Qt, and Android are all based on the Linux kernel (Bovet, Cesati, & Oram, 2002).

Android runs on a Linux kernel, therefore it is a Linux-based system that allows it to run native Linux applications. Linux is a widely used kernel with a lot of hardware compatibilities; this provides support for hardware as hardware manufacturers (and the open-source community developers) build device drivers for Linux platforms to allow specific hardware to interact with the Linux software stack (Vaughan-Nichols, 2010). Once the Linux has detected the hardware, usually the Android software-stack will be able to interact with the hardware, except in some instances where the hardware abstraction is required for Android. This case applies to GPS hardware, Camera support etc. (see Figure 4).

**Hardware Abstraction Layer (HAL)**

This layer is an abstraction layer for hardware communication between the Android framework and the Linux libraries (drivers). The HAL layer will provide the standard interface for libraries and the Android framework to communicate in a standard way. This means that using the HAL, the Android Framework should not need to know the internal details of what specific hardware it is running, but only require the implemented methods of the HAL interface to retrieve or send the data necessary to the hardware.

Examples of uses of the HAL in Android are apparent when using GPS hardware. There are multiple GPS receivers that use slightly different methods to communicate with the software; HAL ensures that the appropriate protocol understood by the Android software-stack is used. The HAL also makes sure that the correct baud-rate\(^\text{11}\) is used for communication which can be device dependant. HAL implementations are written in C/C++.

More details on the GPS and Reverse Camera HAL implementation for the proposed Android-based IVI solution are provided in Chapter 4.

**Libraries**

Libraries, in computer science, are reusable program components that applications can use. They are generally standard sets of code that are ready to use by applications and the application framework. Android libraries are predominantly open-source, standard libraries, and are used in many platforms (including desktop systems such as Windows, MAC O.S.). Some libraries include:

- WebKit web-engine used for Chrome and Safari browsers.
- SQLite database library; a lightweight *server-less* relational database.
- Open GL ES, a graphics library used for 2D and 3D graphics in mobile devices.

\(^{11}\) Baud-rate is symbols per seconds, a method used to determine speed of communication. The sender (hardware) and receiver (software) must use the same baud-rate to correctly read the data. This also applies to software sending data to the hardware.
It is important that the libraries are standard and open-source as it allows native developers (using Android NDK) to be able to use the libraries with the standard set of API’s widely used. These provide developers with the ability to port existing applications to the Android platform with minimal changes.

**Android Runtime**

Android uses a runtime engine known as the Dalvik virtual machine (VM). The Dalvik VM converts Android applications to Dalvik Executable that is optimised for memory and CPU powered constraints on mobile devices. This also allows applications to run in their own processes making multi-tasking possible (Bornstein, 2008).

**Application Framework**

The Application Framework, written in Java, provides all the core framework operations, and a set of API’s, for Android application developers to use for developing applications. This provides all the necessary components such as graphical user-interface elements as well as exposing application components and functionalities to be used by other applications (these are known as intents).

The framework is open, like other components of the system, and can be further extended to provide applications with further capabilities. Framework updates, will generally include new functionalities for applications to use. See Figure 4 for examples of some of the framework components used in the Android operating-system.

**Applications**

Applications are what the end-users will be interacting with; these will provide the users with accessibility to the functionalities of the system. Android includes applications such as the alarm clock, calculator, and more sophisticated applications such as Google Maps Navigation or Last.fm cloud music application¹² (CBS Interactive, 2002). Applications are written in standard Java using Android packages and libraries.

**3.2.2. Android Features (and their importance for IVI systems)**

**Open-source**

The Android operating-system is an open-source platform. Open-source code includes; Linux Kernel, Android framework, libraries¹³ and the basic android applications (such as the launcher, lock-screen, and calculator). Open-source systems allow device vendors to customise and use any hardware components while making the essential changes to the code without rewriting any core components. End-users will be able to use the same Android applications on the IVI system that they would use for their phone.

Furthermore, IVI system device manufacturers can offer more intelligent approaches to exposing some vehicle properties to the operating-system’s framework for applications to use. This could lead to a new set of framework capabilities that allow applications to read vehicle data and even write to

---

¹² Last.fm for Android allows users to stream online music using various techniques such as generating a playlist from a single artist and have the music continuously play while using other applications.

¹³ Most libraries are available within the public source-code, but proprietary closed-source libraries can still be found for certain vendors.
the vehicle computer (i.e. send the vehicle’s computer commands). Research based in Italy looks into this vehicle framework that would use the android security architecture allowing and prohibiting applications from using certain functionalities of the vehicle (Macario, Torchiano, & Violante, 2009). This shows the extensibility of using such a system; once a system like Android becomes a standard de-facto then there would be a standard framework for which vehicle manufacturers can provide an interface to. Android devices can effectively communicate with the vehicle computer to provide us integrated features such as the FIAT’s EcoDrive system, and Ford SYNC’s 911 assist feature (see section 2.2.2).

**Updatable**

The Android operating-system is completely updatable. When a new version of Android released, the source-code is made available to the public and manufacturers. At that time, one needs to take the standard clean source code, apply the necessary changes\(^\text{14}\) and push it out to one’s Android-based products. This enables IVI manufacturers to simply take the update and allow users to manually download it via the vendors website and apply by USB connection or SD-card, or simply retrieve the update through the OTA method (Over The Air) by downloading it directly from the servers.

This is important as it keeps the system well updated, bug-free, and enabled with newer more sophisticated features to provide a longer life for your device. For example; updates from version 2.1 to 2.2 include features such as Wi-Fi Hotspot, Flash integration, Live Wallpapers, and the Just-In-Time (JIT) compilation to increase runtime performance.

**Lightweight**

Android is designed to run on power and memory constrained hardware. This allows android to be used from low-end to high-end devices, and for vendors to produce devices suitable for any use. There are “no minimum processor requirements” as quoted by Google executives (Jones, 2010).

Our proposed system will use a low-powered and low-cost ARM processor that is capable of running all essential applications. It is fan-less and generates minimal to no heat that could interfere with the conditions of the in-vehicle environment (Levy & Many, 2006).

**Connectivity**

Being connected is the current aim for the in-vehicle infotainment field as more consumers are demanding connectivity from within their vehicle. The Android operating-system provides all the required connectivity features such as support for 3G modules that will give users standard 3G mobile Internet access. Android’s current vast application database provides many connected applications that make use of external web-services to provide users with necessary data. This data includes; en-route traffic information, weather, mapping functionalities, GPS tracking of colleagues and friends, news, stocks, business dashboards and many other functionalities that require internet use.

\(^{14}\) These changes often relate to custom manufacturer relevant changes such as applying custom UI
Integration

Application developers sometimes need to provide some custom-made features to devices that are not supported by the framework (e.g. due to the constraints of the current framework). Another reason to build a custom-feature is for the application developer to integrate custom-made hardware to be used in-vehicle. For example a vehicle alarm manufacturer might want to integrate their alarm system with the Android system.

Android provides a supported NDK (Native Development Kit). This means that if there is functionality that is not provided by the framework, Android developers can make applications with native code (C/C++) and can implement code for custom hardware that is not normally supported by the Android Framework (Google Inc., 2007a). This is important to allow third-parties to provide accessories and external peripherals that further extend the life-cycle of the device and provide end-users with special instruments that might be necessary for the user.

Increased security measures can be taken into consideration with gesture-based passwords. This allows the vehicle immobiliser to disarm once the correct gesture is supplied. This means that the vehicle will not be able to start (even with the correct key) unless the correct password is inputted in the Android system (i.e. immobiliser will not disarm). This can also enable remote notifications to security companies and the vehicle owner, as it has an internet connection, and send alerts via internet supplying threat details with supporting GPS data (if the vehicle has been moved).\footnote{An immobiliser prevents the vehicle from starting unless a certain key is provided. In normal circumstances, it will prevent anyone starting the vehicle if the car has not been disarmed (regardless of how the person got inside the vehicle).}
3.3. AIVI: comparison to current IVI systems

Kia UVO, Blue&Me and Ford SYNC systems are all running on a proprietary closed-source Microsoft Auto platform built with the affiliation of car manufactures, with each product having unique features. The Pioneer system is a custom system that is built in-house by Pioneer, made for generic use to be purchased as an aftermarket unit. Lastly we will look into the MeeGo platform, the current open-source solution to the IVI market.

We look into the major components of features indispensable in IVI systems and see how each system utilises these features compared with our proposed Android-based In-Vehicle Infotainment system (AIVI).

Please note: This information was collected during 2010 and the final date for these features is based as of December 2010. We note that as time progresses there could be updates and newer products for these systems to further enhance their capabilities.

The following is a legend that will assist when reading the diagrams used in this section.

<table>
<thead>
<tr>
<th>Description</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capable</td>
<td>✓</td>
</tr>
<tr>
<td>Capable but with exceptions (see notes)</td>
<td>✓ *</td>
</tr>
<tr>
<td>Not Capable</td>
<td>x</td>
</tr>
<tr>
<td>Capable by external synchronised Device</td>
<td>()</td>
</tr>
</tbody>
</table>

3.3.1. Multimedia

Multimedia functionalities are categorised to include music, video and radio/broadcast receiver capabilities.

Music Playback

<table>
<thead>
<tr>
<th></th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>MP3 Playback</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>M4A Playback</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>AAC Playback</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

All major in-vehicle infotainment devices are expected to support the current standard audio formats for music playback. This includes the following three formats:

- MP3 the de facto standard for digital music
- MP4(A) is the main format coming from music stores such as iTunes. iTunes accounts for 69% of the digital download market (Digital Production, 2009).
- AAC standardised format was designed as a successor for the MP3 format. This format is now widely used as the default format for many devices such as iPhone, PlayStation 3 and Wii (Elsen, Hartung, Horn, Kampmann, & Peters, 2002; Wolters, Kjorling, Homm, & Purnhagen, 2003).
All the IVI devices researched support these formats plus various other formats.

**Video Playback**

<table>
<thead>
<tr>
<th></th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVI</td>
<td>✓</td>
<td>✓</td>
<td>()</td>
<td>✓</td>
</tr>
<tr>
<td>MPEG-4</td>
<td>✓</td>
<td>✓</td>
<td>()</td>
<td>✓</td>
</tr>
<tr>
<td>MP4</td>
<td>✓</td>
<td>✓</td>
<td>()</td>
<td>✓</td>
</tr>
</tbody>
</table>

All major video playback formats are supported by the tested IVI systems; however, there were multiple issues to note.

AVI can have multiple encoding formats such as DivX and Xvid (Dranove & Gandal, 2003; Lampert, Militzer, & Ross, 2004). These are not always supported by a video-player that is capable of opening ‘.avi’ files due to the required decoder for the media file needs to be present. Codecs (encoder-decoder) are software algorithms used to encode and decode video formats (Ghanbari, 2003). When new codecs are being used the user must have the matching coder-decoder. This is typically handled by the music application itself, but not always.

A description of how the current IVI systems handle the video formats is given below:

- **Android IVI** – Handles these using third-party applications such as RockPlayer (Redirect Intelligence Corp, 2010). This is frequently updated and can handle all media types presently available with the appropriate codecs.
- **MeeGo IVI** – No official support for MP4 and AVI encoded formats. No codecs installed by default.
- **Microsoft Auto** – Video playback is done via device synchronisation, if the device is capable of viewing the synchronised video, then it will be viewable via the Microsoft Auto platform. However the device must have a dedicated display screen capable of viewing media content. It should be noted that not all Microsoft Auto implementations come with dedicated screens capable of displaying video.
- **Pioneer** – Uses the codecs shipped with the Pioneer unit. These are rarely updated and have limited support (support is available only for the popular video encoded formats, new formats and codecs cannot be added).

It can be observed that the Android system is the most flexible for video playback; one can install the appropriate third-party application which will provide the missing codecs for the video format of choice rather than limiting users to only the supported formats.

**Radio/Broadcast Receiver**

<table>
<thead>
<tr>
<th></th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM Radio</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Internet Radio</td>
<td>✓</td>
<td>✗</td>
<td>()</td>
<td>✗</td>
</tr>
<tr>
<td>Podcasts</td>
<td>✓</td>
<td>✗</td>
<td>()</td>
<td>✗</td>
</tr>
</tbody>
</table>
Broadcast receiver category applies to radio-style casts (note that we are not specifically referring to radio signals). This includes internet radio as well as podcasts

- **FM Radio** – Uses an antenna to receive FM (frequency modulation) radio signals that are broadcasted by radio stations.
- **Internet Radio** – Streaming live radio stations broadcasted over the Internet.
- **Podcasts** – Pre-recorded audio that is episodically released and retrieved by a form of syndication.

Here is how each IVI system deals with broadcasted material:

- **Android IVI** – Capable of streaming live internet radio and downloading podcasts via multiple internet radio clients and podcast applications. FM Radio framework is available and can be implemented in the Android system (as FM Receivers are supported by the kernel).
- **MeeGo IVI** – The MeeGo system currently has no support for any of these at present, but future support is expected since it is still at an early stage which lacks many useful applications.
- **Microsoft Auto** – The platform has native support for radio but for Internet Radio and Podcasts it assumes synchronisation with a media device (e.g. a cell phone). So your phone must support streaming radio and downloading podcasts for it to work.
- **Pioneer** – The Pioneer AVIC-Z1100BT model only supports native radio. It has no support for internet radio or podcasts.

Android seems to be the most complete system when it comes to various media types. Podcasts are now becoming more popular and more universities are distributing lecture material via podcasts (Huann & Thong, 2006; Tripp, DuVall, Cowan, & Kamauu, 2006). Internet radio allows us to listen to any music from around the world through thousands of radio stations that constantly streaming news, music and entertainment.

### 3.3.2. Navigation

Navigation is the ability to use assisted navigation services that allow users to easily find points of interest. Navigation technology has evolved significantly and now there are various techniques of navigation (standard GPS or A-GPS) often conveyed using various methods (i.e. turn-by-turn, street view, satellite view).

#### Hardware

<table>
<thead>
<tr>
<th></th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>GPS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>A-GPS</td>
<td>✓</td>
<td>✓</td>
<td>×</td>
<td>×</td>
</tr>
</tbody>
</table>

---

16 Podcasts are pre-recorded (not streamed live) and episodically released in digital audio. This is generally distributed by a form of syndication. Podcasts are typically downloaded and can be played offline.
• GPS Receiver (global positioning system) uses global navigation satellites to retrieve the user’s current location coordinates.
• A-GPS (Assisted-GPS) uses an ‘assistance’ server to retrieve extra satellite information. This helps to increase satellite locking speeds and reduce the time for retrieving a GPS fix.

Assisted GPS (A-GPS) is lacking in system with no internet connectivity. A-GPS requires an internet connection to obtain extra information from an external server and thus accelerate locking speeds for the GPS. This will provide a better performing GPS. Android and MeeGo’s assisted-GPS support still requires an A-GPS capable GPS receiver (hardware dependent).

Software

<table>
<thead>
<tr>
<th></th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Map</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Turn-by-Turn</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Satellite View</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Street View</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

Software for Navigation has evolved with various methods of appropriately presenting navigation information to the user. These can be done through different methods such as:

• **Map** – This is a basic, 2-dimentional map, with road labels as expected in a physical map and an indicator to show the current location.
• **Turn-By-Turn** – An assisted directive map to guide drivers through the desired route and inform them about specific points to turn at each part of the route.
• **Satellite View** – A satellite image of the map, typically with an overlay of the street names, street barriers, and other information which makes it easier to read.
• **Street View** – An in-car street view with live imagery of the location of interest. This renders imagery of the actual location in order to assist drivers in identifying specific landmarks and intersections. This technique is generally assisted by displaying an arrow for the actual direction in which to manoeuvre.

Please see a detailed explanation for these techniques in Appendix A.

Here is how each IVI system deals with Navigation:

• **Android IVI** – Provides users with multiple methods of mapping with support for Street View, Satellite and Turn-By-Turn. It also has the ability for many alternative methods based on whether they are online or offline as provided by third-party applications. This provides users with all sorts of mapping and navigation utilities on-demand from different providers.
• **MeeGo IVI** – Presently provides only basic map functionality. At this time, there are no external applications which further extend these capabilities.
• **Microsoft Auto** – The Microsoft Auto based IVI systems all provide a basic Turn-By-Turn navigation. There are no external applications to further extend the mapping capabilities.
Android’s large application-base gives users a variety of mapping software which can be used based on specific requirements.

### 3.3.3. Voice

<table>
<thead>
<tr>
<th></th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Speech Recognition</td>
<td>✔</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Speech Synthesis</td>
<td>✔</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>Multi-lingual Support</td>
<td>✔</td>
<td>✗</td>
<td>✔</td>
<td>✔</td>
</tr>
</tbody>
</table>

Speech recognition allows users to input direct voice commands in order to perform certain actions, while speech synthesis generates voice feedback by the system. These technologies are not completely black and white as some recognition engines apply to multiple applications with very flexible and extensive command ranges while others are limited to certain applications.

Here is how each IVI system deals with the ‘voice’ capabilities:

**Android-based IVI**

Speech Recognition, by default, is dealt via Google’s voice recognition engine. This engine is a web-service utilised by applications which provides real time recognition via internet connection to recognise speech and appropriately apply the programmed action. This is a very flexible technology; application developers can integrate speech recognition capabilities in conjunction with other applications. Examples can include; “get directions to Pizza Hut”, “What is the weather?”, “Multiply 354 times 26”. These capabilities can reduce the user’s physical interaction with the system in order to concentrate on driving. This technology, in-conjunction with speech synthesis, further increases driver safety.

The system is constrained by the fact that it does require an internet connection, but there are alternate services emerging for Android with Nuance releasing the SDK for Dragon Speak (a popular speech recognition engine)(Trout, 2011).

**MeeGo IVI**

This system does not currently support voice recognition and voice synthesis capabilities in its current release. Since this is still a maturing product we expect to see these features in future releases of this product.

**Microsoft Auto**

Microsoft Auto platform’s speech recognition engine is built-in and known as the Microsoft Voice Command. However, some device vendors that run the Microsoft Auto platform use different speech recognition methods. Ford and the Ford SYNC device use Nuance technologies for the speech recognition capabilities. Regardless, the speech recognition engine is still limited to only a few applications supplied by the manufacturer (Nuance Communications, 2010).
Pioneer AVIC-Z1100BT

Pioneer also provides a limited speech recognition engine that is built-in to the device using VoiceBox technologies (Davies, 2008). There are a standard set of commands set by the provider to make available basic music-navigation capabilities.

3.3.4. Display

<table>
<thead>
<tr>
<th>Feature</th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size</td>
<td>7-inch</td>
<td>✗</td>
<td>Varies</td>
<td>7-inch</td>
</tr>
<tr>
<td>Stylus Input</td>
<td>✓</td>
<td>✗</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Multi-touch Interface</td>
<td>Capable</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

The display is completely hardware dependant. Each manufacturer has different approaches to the display screen:

- **Android IVI** – Our Android IVI solution will be maximising the fixed area of the double-DIN standard in-dash slot space and use the 7-inch touch-capable screen (4-inches by 7-inches in size). It is currently a single-touch module but has the potential for a multi-touch interface for smoother, easier system interaction.

- **MeeGo IVI** – Supports various sizes and formats of touch-capable screens but there is no hardware currently being tested for these modules. However, the IVI system is expected to have a touch-capable screen (stylus or multi-touch capable). Applications for this system are not dependant on screen-size and thus are displayable on different screen sizes.

- **Microsoft Auto** – Supports various screen types from single-lined matrix style interfaces to full-colour 7-inch display screens.

- **Pioneer** – The module also has a 7-inch touch-capable screen and the software is designed specifically for that display.

All IVI systems are expected to have a display screen of some sort with a touch-capable interface.

3.3.5. Software

<table>
<thead>
<tr>
<th>Feature</th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating-system</td>
<td>Android O.S. 2.1</td>
<td>MeeGo 1.0 IVI</td>
<td>4.1</td>
<td>✗</td>
</tr>
<tr>
<td>SDK</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Application Store</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Updatable</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
<tr>
<td>Web-applications</td>
<td>✓</td>
<td>✓</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

The software-stack for the IVI systems is important to consider as they produce the functionalities and applications that allow users to interact. Having a standard, updatable operating-system, with an SDK (Software Development Kit), means a large and accessible application store is necessary to provide users with a complete and practical system. It is also important that these systems are able to
access web-applications with internet connectivity in order to provide users with live data which is constantly updated (for live traffic alerts and other features that require live data).

Here is some information regarding operating-system features and other software features in the selected IVI systems:

- **Android IVI** – Android provides all the crucial SDK tools and has over 100,000 applications in its application-store to be used on Android systems. Android is completely updatable, has access to many web-applications, and is built on top of a Linux kernel.

- **MeeGo IVI** – The platform is also built on Linux and provides users with an SDK with the intentions of an application-store, being that it is an updatable system and has access to web-applications. But the SDK and current platform’s API level is limited compared to the matured Android system. There is no current application-store and the known applications are very limited. It has been reported that the MeeGo platform is running ‘poorly’ on existing devices tested with MeeGo (Miller, 2011).^{17}

- **Microsoft Auto** – This system lacks developer API’s and a general SDK. It has no current support for external applications other than the applications that are shipped with the product. It is not extensible and has no connection to web-applications. The system is standard (in that it uses a kernel and the core is the same across different platforms such as Kia UVO, Ford SYNC and FIAT Blue&Me) but applications are not interchangeable across each system.

- **Pioneer** – The system lacks a standard operating-system (as it uses custom Pioneer software). This operating-system does not provide an SDK and extensibility to provide users with third-party and external applications. It is not connected; therefore there is no support for web-applications^{18}.

Android provides a well-established system with good support for third-party developers to further extend the capabilities of Android devices by providing third-party applications.

### 3.3.6. Communication

<table>
<thead>
<tr>
<th></th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Voice Calling</td>
<td>✓ *</td>
<td>x</td>
<td>()</td>
<td>()</td>
</tr>
<tr>
<td>SMS</td>
<td>✓</td>
<td>x</td>
<td>()</td>
<td>x</td>
</tr>
<tr>
<td>GPRS</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>3G</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

Communication facilities in an IVI system facilitate IVI’s to communicate with external systems. This includes basic voice communication and SMS (Short-Message Service), through to Internet capabilities such as GPRS and 3G connections.

---

^{17} These were not official MeeGo devices, but devices that have had MeeGo ported to them. These include Nokia N900 and HTC HD2.

^{18} Some Pioneer IVI systems and the Ford SYNC provide the ability to connect to web-applications, such as Pandora, on the user’s personal mobile device (such as an iPhone or Android phone).
Information regarding how these IVI systems and platform deal with communication:

- **Android IVI** – The Android-based solution has normal voice-calling capabilities. However, the current implementation lacks the GSM hardware to make voice calls via the GSM network. Regardless, the proposed solution can still make VoIP\(^\text{19}\) calls as well as send SMS and have full connection to 3G and GPRS (if 3G connection is not covered in a specific area). It has the potential to be synchronised with existing smartphones and make phone-calls by Bluetooth (currently not implemented).

- **MeeGo IVI** – At present, this IVI system tested does not have voice and SMS functionality (the MeeGo platform, for handsets, in general has support for these features).\(^\text{20}\) The MeeGo IVI platform also provides support for 3G connections.

- **Microsoft Auto** – This system currently uses synchronisation by Bluetooth to connect to the user’s mobile phone to make calls, and send and receive SMS messages. But the system has no native internet connectivity (no 3G or GPRS).

- **Pioneer** – Lacks all these features, but has Bluetooth connectivity to connect to your handset. The Pioneer system has no native support for an internet connection, or a GSM module to make regular phone calls natively (without the use of a Bluetooth synchronised device).

There is a growing demand for internet connected IVI systems, and for vehicles to be connected in general. Currently no real IVI system commercially available has internet connectivity, but the proposed Android-based IVI system will be able to have persistent 3G connection.

### 3.3.7. Peripherals

<table>
<thead>
<tr>
<th></th>
<th>Android IVI</th>
<th>MeeGo</th>
<th>Microsoft Auto</th>
<th>Pioneer</th>
</tr>
</thead>
<tbody>
<tr>
<td>USB Host</td>
<td>✓</td>
<td>x</td>
<td>()</td>
<td>()</td>
</tr>
<tr>
<td>External Storage</td>
<td>✓</td>
<td>x</td>
<td>()</td>
<td>x</td>
</tr>
<tr>
<td>CD/DVD drive</td>
<td>✓ *</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>OBD-II connection</td>
<td>✓ *</td>
<td>x</td>
<td>✓</td>
<td>x</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>✓ *</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Reverse Camera</td>
<td>✓</td>
<td>x</td>
<td>✓ **</td>
<td>x</td>
</tr>
<tr>
<td>Wi-Fi Hotspot</td>
<td>✓ *</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
</tbody>
</table>

* The proposed android-based IVI system currently does not have the required hardware but the Android application framework can support these features.

** Only the Kia UVO IVI system supports a reverse camera from the Microsoft platform.

**OBD-II**

On-board diagnostics - is a reporting and diagnostics feature that is present in almost all vehicles on the road (with slightly different variants).\(^\text{21}\) The diagnostics collects and reports relevant data that can help identify problems or provide analysis on various factors. To retrieve data from the OBD-II

\(^{19}\) VoIP – Voice over Internet Protocol is used to make voice (and video) calls via a VoIP service (such as Skype).

\(^{20}\) MeeGo is a platform that is delegated for many devices, including handsets, netbooks as well as IVI systems.

\(^{21}\) OBD-II protocol must be present in all vehicles from 1996 (in the United States of America).
module, a connector is designed specifically for the pin-layout of the OBD-II. Data is then analysed according to certain ID’s and values according to the OBD-II references (Birnbaum & Truglia, 2001). OBD-II connectors allow IVI devices to communicate with the vehicle’s computer and read data such as fuel consumption, oxygen sensor information, and error codes for mechanical and electrical errors.

OBD-II capabilities are present in the Microsoft Auto platform, but are limited to the pre-existing features available. These include Ford SYNC’s 911 Assist feature and FIAT’s EcoDrive in the Blue&Me system. This is because users are limited to using the applications provided by the manufacturer. By contrast, Android allows multiple third-party applications to be used with OBD-II connections to read and analyse vehicle behaviour according to the data retrieved (OBDroid, 2010).

Wi-Fi Hotspot

Wi-Fi hotspot allows users to connect their mobile devices (such as smartphones and laptops) to a Wi-Fi connection that is hosted by the IVI system which provides a wireless network within the vehicle. This can be useful in many cases for business or personal purposes. For example, a taxi company could take advantage of this to provide passengers with a Wi-Fi connection within the taxi vehicles. Users can use any of their Wi-Fi enabled devices, such as laptops or smartphones, to access the internet or communicate with other devices in the network (i.e. to play games, or share files with passengers).

At this time, no IVI system supports such a feature. However, Android has the capabilities to use the device as a Wi-Fi hotspot as long as the hardware is capable (possible as of Android 2.2) (Michael Arrington, 2010).

A complete comparison chart can be viewed in Appendix C.

3.4. Opinion Survey

It is important to know what users want from an IVI system, so an opinion survey was conducted in order to understand what the public feel about using such a system, and the priority of importance for the features offered by the system. The survey offers an opportunity for the participants to raise any concerns they might have about the system (e.g. safety concerns).

3.4.1. Opinion Survey Design

The opinion survey was designed to break down each feature and see how desirable these features are to the end-users. Users were presented with options using the likert scale and asked to choose the desired option from levels of 1-5 (Very Undesirable, Undesirable, Neutral, Desirable, and Very Desirable). The survey was undertaken by 47 participants.

The opinion survey also captured general information regarding the users to ascertain what the user demographic is. This included: occupation details, technical experience, and mobile internet consumption. These details are important to understand how such a system will be perceived by different user groups in the current market.

The distribution of the survey was done by social networking websites and email distribution. The actual survey was hosted online and users were asked to complete the survey using the form provided (by clicking a hyper-link), and submit it once completed. This helped in keeping participants anonymous and provided a less intrusive method of delivering the survey for participants to answer the questions, without any pressure and in their own time. This is in contrast to a researcher directly asking questions of the participants, or having them fill out a survey form in front of the researcher.
The opinion survey questionnaire is attached in Appendix B.

3.4.2. Desirability of the IVI Features

3.4.2.1. Audio player – Music and Radio

Description:

Audio player is a feature that is expected to be present in all IVI systems. All IVI systems that were considered as a part of this thesis have presented this feature, and it is the oldest feature of in-car entertainment (Cosper, 2006). It is important to have such an option in our questionnaire to gauge the interest of the participants in such a system, since an audio player is a basic feature that is present in all IVI systems.

Results:

![Desirability of 'Audio Player' feature in IVI systems](image)

Figure 5 - Opinion Survey Chart: Desirability of ‘Audio Player’

Analysis:

This question can be considered as a benchmark question. Historically, audio player is one of the earliest features of in-car entertainment (before infotainment). This question was used to identify the interest generated from an IVI system and verify whether people understand what an IVI system is. 83% of respondents answered ‘Very Desirable’ to audio playing features, while a further 9% considered it to be desirable feature. Only 6% of respondents found the ‘Audio Player’ feature to be undesirable. 2% were neutral on this issue.
3.4.2.2. Video player

Description:

A video player can play all motion-based media including streamed media. This feature is another important feature of IVI systems and it is used whenever a system has a display-screen. Safety of drivers is an important concern that arises from such a feature. Therefore, safety measures are regularly put into place to avoid driver distraction while the vehicle is in motion.

Results:

![Desirability of 'Video Player' feature in IVI systems](image)

Figure 6 - Opinion Survey Chart: Desirability of ‘Video Player’

Analysis:

Neutral results were obtained for the ‘Video Player’ feature of the IVI system. Almost half of the participants found the feature desirable (49% found it Desirable/Very Desirable). However, 32% of respondents were neutral regarding this feature. Written feedback from this specific question related to the safety concerns regarding the practicality of video playing features being present, especially when in-sight of the driver. This is considered in light of the possibility of distraction to the driver caused by watching the motion video while driving (and not paying attention to the road).
3.4.2.3. Turn-By-Turn Navigation (GPS)

Description:

This includes voice-guided navigation with voice input. A GPS system is present in most systems for drivers to effectively navigate to a desired location of their interest. This question on the desirability of ‘Turn-By-Turn Navigation’ allowed us to understand how important GPS navigation is for the end-users.

Results:

![Desirability of 'GPS Navigation' feature in IVI systems](image)

Figure 7 - Opinion Survey Chart: Desirability of ‘GPS Navigation’

Analysis:

Most participants felt that Turn-By-Turn navigation is an important feature of infotainment systems (87% of respondents overall see this feature as desirable, with 57% considering it very desirable).

Nowadays, the GPS infrastructure of in-vehicle infotainment systems is used as a basis to provide advanced and useful applications such as Street View (Google Inc., 2007b), Satellite View, and social-networking applications. Voice Actions also allow users to interact with the GPS Navigation application via speech, thus shielding drivers from distractions caused by interacting with the system using their hand.

---

22 Social-networking GPS Applications allow you to find friends and family with data that is updated in real-time.
3.4.2.4. **Reverse Camera**

**Description:**

A camera placed at the rear of the vehicle can provide the driver with a better view while reversing. This is a feature that is not present in Android systems and is more commonly known as the Reverse or Back-up camera. This question allows us to understand how desirable such a feature is in an IVI system.

**Results:**

![Desirability of 'Reverse Camera' feature in IVI systems](chart)

**Analysis:**

This feature seems to be desirable to 60% of respondents. A further 34% of the respondents seemed to have neutral feelings about the desirability of such feature. Another 6% of respondents found this feature to be undesirable (including 2% finding it very undesirable).

Upon further investigation, 78% of respondents, who were neutral or found this feature to be undesirable or very undesirable, were current holders of a Full Licence or international licences. This could mean that they are more experienced drivers, as they may be well accustomed to the effective use of mirrors and manually turning their heads to check the blind spots. However, it should be noted that having a reverse camera is a safety measure, and having safety measures, regardless of whether drivers feel like them or not, are important for all IVI systems.
3.4.2.5. Voice Actions

Description:
Speech recognition with voice feedback (speech-to-speech) is used for: composing and reading SMS messages and emails, obtaining weather information, getting driving directions, composing and receiving ‘tweets’, creating note-to-self, doing simple calculations based on voice commands such as “calculate…”, using a dictionary to define or translate words, and many other applications.

This is another feature that is not normally present in IVI systems and is unique to an Android-based system. This question allowed us to see how desirable such a new feature is to our end-users.

Results:

Figure 9 - Opinion Survey Chart: Desirability of ‘Voice Actions’

Analysis:
We can see from Figure 9 that ‘Voice Actions’ is found to be desirable by 58% of the survey participants. However, 21% of participants found this feature not desirable (including 6% rating it ‘Very Undesirable’). Our opinion is that most participants who answered these questions lacked faith in speech recognition technologies relating back to their own experiences (or second-hand knowledge) of earlier implementations of such where users had to train the system before effectively using it. This technology has now advanced enough to understand different dialects and accents of different cultures. It can be noted that 60% of users who had no previous experience with smartphone devices (iOS, Android, webOS, or BlackBerry) answered 3 or below for this feature. Therefore, these users may not be aware of the potential use of the voice-enabled actions that can be performed by the system.
3.4.2.6. Email

Description:

Email is important for businesses and workers, and therefore this question concerns the ability to receive in-vehicle email notifications with different methods of input (e.g. on-screen keyboard or dictating using ‘Voice Actions’, as previously noted in 3.4.2.5).

This is another feature that raises safety concerns since users may be distracted while reading emails, however there is an option for the system to read the emails aloud. It is important to understand how users feel about retrieving email updates and alerts instantly wherever they are. This feature is unique to a connected system.

Results:

![Desirability of 'Email' feature in IVI systems](image)

Figure 10 - Opinion Survey Chart: Desirability of Email’

Analysis:

This was another feature that showed mixed, but mostly positive, responses. Figure 10 shows us that Email is considered to be desirable by 54% of respondents. 32% of respondents were neutral about such a feature and a further 15% of the respondents found this feature at least undesirable.

This was interesting because email is vital to most people who require it for their occupation. Upon further investigation, we generated a list of the respondents who rated this feature as 3 or below (Neutral, Undesirable or Very Undesirable). 75% of the respondents who gave a score of 3 or below were students (with non-students being pharmacists, unemployed, as well as a kitchen chef). We also found that 84% of respondents who are working professionals consider this feature to be either desirable or very desirable (a score of 4 or 5).
This shows that this feature is a lot more attractive to working professional and less desirable to students and non-working people. This analysis is important because the graph, in Figure 10, does not show the complete story (i.e. the divided opinion between professionals vs. students) for this feature.

3.4.2.7. Internet Browser

Description:

This feature includes general internet browsing and the use of web-applications such as Facebook (with Voice Actions). As stated earlier, this feature poses safety concerns. However, users are demanding Internet connectivity in their vehicles and it is important to understand how significant having Internet browsing capabilities is for users.

Results:

![Desirability of 'Internet Browser' feature in IVI systems](chart.png)

Figure 11 - Opinion Survey Chart: Desirability of ‘Internet Browser’

Analysis:

Looking at the chart in Figure 11 we can see that most respondents are in-favour of web-browsing capabilities. 62% of respondents rated this with a desirability level of 4 or 5 (Desirable and Very Desirable). But 21% were neutral about such a feature, and a further 17% found this feature to be undesirable.

We believe that using the term ‘web-browsing’ might have caused a few concerns, especially regarding practicality and safety, since some respondents questioned these matters while answering the survey. Typical web-browsers have an address bar, require a lot of user input (using an on-screen keyboard), and only allow users to browse pages. Using web-applications specifically designed for mobile use and touch-input, and voice dictation, will address these problems as it conveniently
aggregates functionalities that users need and can use in a mobile environment, such as being on the road. These web-applications can also assist drivers by providing insightful information regarding live traffic information and weather conditions.

3.4.2.8. Installing external third-party applications (Apps)

Description:

Installing external third-party applications, more commonly known as Apps, allows users to add extra functionality to the system. This includes applications that supply information about traffic conditions, show traffic cameras, enable music streaming, and deliver news and weather information.

As more users are acquiring smartphones, we need to see how users value installing external applications. With the rise of smartphones, such as Android and iPhone, third-party applications are becoming common. Users now understand that they can install whichever application they require and remove the undesired applications to suit their personal needs.

Results:

![Figure 12 - Opinion Survey Chart: Desirability of ‘Installing Apps’](image)

Analysis:

This feature had an overwhelming positive response. With 32% considering this feature ‘Desirable’ and a further 43% finding this feature ‘Very Desirable’. In contrast, 17% were neutral with a further 8% finding this feature at least undesirable.

The responses obtained are interesting because they show that a significant amount of people are aware of third-party applications and smartphone capabilities. 90% of the survey participants had some experience with a mobile operating-system such as Android, iOS (iPhone, iPod touch, or iPad), BlackBerry, or webOS (PalmPre). This shows that people nowadays are much better informed when it comes to modern gadgets and their capabilities, as all these operating-systems allow third-party applications to be installed.
3.4.2.9. **In-Vehicle Wi-Fi Hotspot**

**Description:**

This is the ability for vehicle occupants to connect smartphones, laptops, and other devices via Wireless (Wi-Fi) in order to obtain an internet connection.

This is another feature unique to the Android platform; although not yet implemented in our current concept, it is a feature worth noting. Internet access is fine from a central IVI system, but how important is personal internet to the user? More people now carry personal devices with Wi-Fi receivers; the ability to connect to the Internet on demand, with personal devices, is an important feature to study and understand for future implementations.

**Results:**

![Desirability of 'Wi-Fi Hotspot' feature in IVI systems](image)

**Analysis:**

The overall response to the in-vehicle Wi-Fi hotspot feature was highly positive. It can be observed from Figure 13 that the idea of having a wireless hotspot within a vehicle, thus bringing wireless connectivity to vehicle occupants, was appealing to 81% of the respondents (including 62% who found gave this feature the maximum rating of 5).

A growing number of people need to be connected on demand, and having a centralised hotspot which can be accessed by multiple devices on demand is important to more people as they require Internet on-the-go. This can provide personal infotainment to passengers of vehicles, for example, in keeping occupants entertained by providing the ability to stream media (music and video) and browse the internet.
3.4.2.10. Feature Ranking

A rank ordered list might be desirable to present these features to users. In order to rank the features considered for our system, we used a weighting scheme for the options given to the users as presented in Table 14.

<table>
<thead>
<tr>
<th>Response</th>
<th>Weighting</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 – Very Undesirable</td>
<td>-2</td>
</tr>
<tr>
<td>2 – Undesirable</td>
<td>-1</td>
</tr>
<tr>
<td>3 – Neutral</td>
<td>0</td>
</tr>
<tr>
<td>4 – Desirable</td>
<td>1</td>
</tr>
<tr>
<td>5 – Very Desirable</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 14 - Response Weighting

Based on these weightings, we can then generate a Desirability score and effectively rank the IVI features in-order of desirability. Desirability score (Ds) is calculated using the following simple formula:

\[ D_s = (N_{vu} \times -2) + (N_u \times -1) + (N_d \times 1) + (N_{vd} \times 2) \]

- \( N_{vu} \) = No. of participants that considered a feature to be very undesirable
- \( N_u \) = No. of participants that considered a feature to be undesirable
- \( N_d \) = No. of participants that considered a feature to be desirable
- \( N_{vd} \) = No. of participants that considered a feature to be very desirable

The features were calculated based on the above formula and the result is given in Table 15.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Desirability Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Player</td>
<td>77</td>
</tr>
<tr>
<td>Turn-By-Turn Navigation</td>
<td>64</td>
</tr>
<tr>
<td>In-Vehicle Wi-Fi Hotspot</td>
<td>59</td>
</tr>
<tr>
<td>Installing External Apps</td>
<td>49</td>
</tr>
<tr>
<td>Reverse Camera</td>
<td>36</td>
</tr>
<tr>
<td>Voice Actions</td>
<td>29</td>
</tr>
<tr>
<td>Internet Browser</td>
<td>29</td>
</tr>
<tr>
<td>Email</td>
<td>28</td>
</tr>
<tr>
<td>Video Player</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 15 - Desirability Score of IVI Features
A visual representation of the results presented in Table 15 is given in Figure 16.

![Desirability Score of IVI Features](image)

Figure 16 - Desirability Ranking of IVI Features

These rankings will allow developers to determine the default layout, and order of applications for the user-interface designed for the proposed system. This is used in the navigation pane of our application in Figure 17.

![User Interface for In-Vehicle Use](image)

Figure 17 - User Interface for In-Vehicle Use

Users are not restricted to the default layout provided and will be able to modify the order and add other applications to the navigation pane.

Details of the application developed are discussed in Chapter 4.
3.4.3. Participant Demographics

General Questions:

- What is your gender?
- Do you own a car?
- What is your occupation?
- What is your current driver license?
- What is your age group?

These questions allow us to compare participants’ interest based on their demographics, and understand how people of certain occupations might lean towards certain features, while others do not. It also highlights differences in opinion between car owners and non-car owners, the different types of car licence holders, as well as different age groups regarding the desirability of certain IVI features.

Technical Questions:

3.4.3.1. Previous experiences with smartphone operating-systems.

Description:

This question allowed us to analyse users and their experience with the latest smartphone devices. This showed how tech-savvy users are, the uptake of the Android operating-system, and the importance of third-party applications.

Results:

![Participants Experience with Smartphone Operating System](image)

Figure 18 – Opinion Survey Chat: Mobile Operating-System Experience

---

Participants may select more than one checkbox, so percentages may add up to more than 100%.
Analysis:

It can be observed from Figure 18 that a vast majority of the users have experience with popular smartphone mobile-operating systems. 89% of participants have had experience with at least one of the operating-systems listed above. 74% of all respondents have experience with two or more of those operating-systems (usually experience with Android O.S. as well as iOS). This shows that consumers in recent times have an improved awareness when it comes to understanding the importance of the software running on a particular device rather than just the hardware itself.

3.4.3.2. Download vs. Stream user preference.

Description:

This question helped us to understand what the behaviour of current internet users is with regards to streaming or downloading media. This gave us an idea of which applications users prefer to use in order to retrieve media.

Results:

<table>
<thead>
<tr>
<th></th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Download</td>
<td>66%</td>
</tr>
<tr>
<td>Stream</td>
<td>32%</td>
</tr>
<tr>
<td>Not Sure</td>
<td>2%</td>
</tr>
</tbody>
</table>

Figure 19 – Opinion Survey Chat: Downloading vs. Streaming

Analysis:

These results show that 66% of users are downloaders, while 32% prefer to stream. This is important to note as there is a growing trend of streamers; now more applications are hosted on the cloud (where all data is stored on an internet cloud) that is accessible from any connected device. It is also important to note that while streaming might not be favourable at this point of time, when there is a rise in the number connected devices and the cost of mobile connectivity decreases, these numbers might change. More people may choose to have their data (multimedia or documents) accessible from any connected device.
3.4.3.3. Estimated user mobile data-usage per month.

Description:

This question aims at effectively analysing the behaviour of mobile internet users and their data consumption using their mobile devices. This will give us an understanding of how many users are using data on mobile devices, since this IVI system is effectively a connected mobile device.

Results:

![Mobile Data Usage Chart]

Figure 20 - Opinion Survey Chart: Mobile Data Usage

Analysis:

Figure 20 shows us that there are a lot more people connected and using mobile internet on a regular basis (77%). The majority of users (28%) are using between 50 to 100MB of mobile data monthly while 94% of all respondents appear to have internet capable mobile devices. Furthermore, there is an overwhelming rate (21%) of users using more than 1GB of data. This shows that more people are connected via mobile and there is a much higher demand for mobile internet.
3.4.4. Open-ended Questions and Comments

Open-ended questions posed to the participants were optional and some users gave their opinions about the proposed system for the following questions:

Are there any other features you deem necessary for In-Vehicle Infotainment (IVI) systems? - Any further comments you would like to add?

Some of the comments provided by respondents are as follows:

“In addition to the voice activation idea, using voice to initiate air conditioning, window wipers etc..

This was an interesting comment because these features are important for an ideal IVI system that can fully control vehicle mechanics. This is why a standard framework needs to be designed, but car-manufacturers must follow these and expose certain API’s of the vehicle computer to allow developers to build applications which can control certain functionalities. This is not a simple task since the idea can become problematic when considering the security measures that must be put into place when implementing such a system. Presently, vehicle computers only allow the reading of data and information from the vehicle. Controlling vehicle mechanics, such as the window wipers or Air Conditioning, needs the capability to write to the vehicle’s computer. If such a capability is exploited by malicious applications (which is always a possibility for any operating-system), then it could cause serious safety concerns while on the road.

Weather, Traffic Reports GPS Routing, updated maps for construction changes, Internet Radio / Music Downloads.

All of these are possible with the currently proposed system. Since the system is connected, it allows any applications to be connected to web-servers and retrieve the necessary data. This includes: traffic information, music downloads, and internet radio.

Audio and video player supports all file formats A built in alarm/Garage door controller would be nice

Currently all popular Audio and Video formats are supported. When there is a format that is not supported, the codec for that format can be retrieved via a third-party application and installed by the user.

A garage door controller is also possible since it only requires a method of communication which can be via Wi-Fi or 3G (with the garage door receiver connected via home network). It is an interesting scenario and can be considered for future work.

You didn’t say anything about making phone calls, dunno if it counts as infotainment.. Everything else is covered by 3rd party apps
Future work with the system will allow users to synchronise their phones via Bluetooth, and place phone-calls directly from the infotainment system. This also enables the ability to use a hands-free kit and ‘Voice Actions’ for a safer driving experience.

"Being able to plug in your devices using USB cable"

This is covered by our current implementation of the proposed system. There is an external USB hub which allows users to connect USB devices. These devices include flash drives and other forms of media (MP3 player, mobile-phones). These USB devices can be mass-storage based and multimedia devices, but this also extends to custom devices. Most devices are supported by the Linux kernel, but for custom devices to function appropriately in the Android operating-system it requires drivers to be written for that particular device. This was the case when implementing the Reverse Camera and 3G features of the device (these devices are interfaced by USB).

"Interaction between cars, location of other cars (friends and family etc)"

This is another very interesting comment. It shows that people are demanding such functionality even though this already exists as a third-party application. Applications such as Waze allow users to locate and connect to each other in real-time. It also allows users to socially interact by providing important information such as traffic conditions, or road hazards such as accidents on a particular path (Waze Mobile, 2010). There are also usually certain privacy measures taken into consideration when you need your location to be private or only visible within a certain group.24

Please see Figure 22 for a screenshot of Waze, a social networking GPS applications for Android, iOS and Blackberry.

Figure 21 - Waze on Android (Waze Mobile, 2010)

For all responses please see Appendix E.

24 Grouping feature is available in Waze 2.0.
3.5. Legal Implications

Although there is no specific amendment relating to In-Vehicle Infotainment, the following amendments are the most related to Infotainment systems in New Zealand.


2.5(1) Except as provided in 2.5(2), any part of the image on a television screen fitted in a motor vehicle must not be visible to the driver of the motor vehicle from his or her normal driving position while the motor vehicle is in motion.

2.5(2) **Subclause 2.5(1) does not apply if:**

(a) the television screen is fitted in the motor vehicle only for the purpose of assisting the driver to reverse safely, by showing a clear picture of the area directly behind the motor vehicle; or

(b) the motor vehicle is a passenger service vehicle and the provisions of *Land Transport Rule: Passenger Service Vehicles 1999* are complied with; or

(c) the screen is fitted as original equipment by the vehicle manufacturer and is designed so that only information relating to the navigation, safe operation and control of the motor vehicle can be displayed on the screen while the motor vehicle is in motion; or

(d) the screen is only capable of displaying text and any change to the text on the screen is controlled manually by the driver.

The amendment prohibits the use of motion pictures while the vehicle is moving (if in sight of the driver). Safe measures can be implemented to prevent certain applications from starting which deemed to be unsafe while the vehicle is not stationary. Applications that assist drivers such as GPS and reverse camera can still be launched while the vehicle is in motion.

Another related amendment is the Land Transport Amendment Rule of using mobile phones. It is not the same as an infotainment system, but it contains related features of a communicating device.

**Ban on use of mobile phones while driving** (Land Transport [Road User] Amendment Rule [No 2], 2009).

Rule 7.3A(1) is revoked and the following subclauses are substituted:

“(1) A driver must not, while driving a vehicle,—

“(a) use a mobile phone to make, receive, or terminate a telephone call; or

“(b) use a mobile phone to create, send, or read a text message; or

“(c) use a mobile phone to create, send, or read an email; or

“(d) use a mobile phone to create, send, or view a video message; or

“(e) use a mobile phone to communicate in a way similar to a way described in any of paragraphs (b) to (d); or
“(f) use a mobile phone in a way other than a way described in any of paragraphs (a) to (e).

“(1A) Subclause (1) is overridden by subclauses (2) to (7).”

(2) Rule 7.3A(5) is revoked.

(3) Rule 7.3A is amended by adding the following subclause:

“(7) A driver may, while driving a vehicle, use a mobile phone in a way described in subclause (1)(a) or (f), if both the following apply:

“(a) the phone is secured in a mounting fixed to the vehicle;

And

“(b) if the driver manipulates or looks at the phone, he or she does so infrequently and briefly.”

The objective of this rule is to amend the principal rule to allow a driver to use a mobile phone for any purpose, other than a text message, video message, email, or similar communication, while driving. The phone has to be secured in a mounting that is fixed to the vehicle and the driver is allowed to manipulate and look at the phone only infrequently and briefly. A driver will also still be able to use a hands-free mobile phone.

3.6. Summary

The requirement gathering phase allowed us to collect all the information necessary to effectively understand the requirements for the next-generation of IVI systems. This chapter has addressed the following issues:

Feature Gaps in current IVI Systems

After effectively explaining the features of current IVI systems with our proposed system, we can see (from the above charts and tables) that there are gaps when it comes to functionality. These include third-party applications, a developer SDK, and internet connectivity as some examples. Furthermore, when the relevant features do exist in current systems, they are not generally sophisticated or advanced enough to be used effectively.

Need for Connected Systems

We have also shown that there is a need for connected systems. Connectivity is important for an IVI system as it enhances the overall experience of a driver by providing required information (e.g. traffic and weather details). Connected systems also need a supporting web-application and access to web-services for acquisition of real-time data. The lack of features, and of sophistication of available features in existing IVI systems, can be addressed by using an Android-based system, as described in section 3.3.

Enhanced user technical expertise:

Looking back at the opinion survey, we can see that consumers these days are a lot more tech-savvy, even if they are outside of the IT field. More people want to be connected and use mobile internet regularly, and almost all users have experienced connectable and touch-capable smartphone
devices that are available at this time. This in-turn helps the deployment of such a system and eases the adoption process of the system if users are already familiar with some of these components.

Safety Concerns
The issue of safety is an important concern; however IVI systems could involve user-interaction while the vehicle is mobile. Taking into consideration current safety measures in existing systems, as well as concerns raised by users of the systems, we can effectively address some of the issues raised in regards to safety. This includes the ability to automatically turn-off motion video when the car is moving. Other safety measures taken into place regard the ability for the system to provide different means of input to reduce haptic interaction with the system: this includes voice recognition features (Voice Actions, where the user dictates commands) as well as speech synthesis (speech feedback, where the system reads back the output).

Legal Implications
Based on our investigations on the legal implications for the use of in-vehicle infotainment systems, we note that there are no specific amendments regarding the use of such systems. According to relevant amendments (use of televisions and mobile phone use in motor vehicles), we conclude that the typical interactions with these systems must be minimum and they should not distract the driver which may lead to hazardous situation. Safety measures must be taken into consideration as described in the previous paragraph.

When considering the IVI system comparison and the opinion-survey data gathered, we can see that an Android-based system is necessary to provide all requirements by end-users. The system will provide all the features deemed necessary for an IVI system, as well as bringing a whole level of extensibility that gives end-users the ability to customise and extend the system’s capabilities.
Chapter 4
System Design and Implementation

4.1. Overview

The objective of this chapter is to discuss the design and implementation of the AIVI system. This chapter is divided into a few sections. They are:

- **Hardware Components** –
  Explain the hardware components used, design decisions and considerations made for these components.

- **Software Implementation** –
  Covers embedded development, kernel development, Android core\(^{25}\) and hardware abstraction development that was used in this research.

- **Application Design** –
  Describes software components including UI design of the application used in the system that integrates the components implemented.

These sections will provide an insight in the design decisions made and implementation techniques used for the integration of all components to provide the proposed IVI system.

\(^{25}\) This refers to the main components of the Android operating-system.
4.2. Hardware Components

The proposed AIVI system has a set of hardware components that were selected as suitable for an IVI solution. The design decisions in picking the hardware were thoughtfully considered. The developed system should not only be an effective system, but also an affordable system that end-consumers can purchase off the shelf.

4.2.1. Android In-Vehicle Infotainment (AIVI) specifications:

This section covers the hardware specifications used in the proposed AIVI system:

- Processor – Includes the main core and motherboard used.
- Memory – Storage memory and random access memory (RAM)
- Audio and Video
- Peripherals – External components (such as GPS, 3G, Reverse Camera etc.).

**Processor**

Texas Instrument OMAP (Open Multimedia Application Platform) is a category of proprietary microprocessors that have capabilities for portable and mobile multimedia applications (Guming, 2003). OMAP is developed by Texas Instruments (TI) and is used by many mobile phone providers, including most of Nokia’s N-series range. Other smartphones such as the Palm Pre and the Motorola Droid also use an OMAP processor (OMAP3430).

- OMAP3530 processor (ARM, 2008; Guming, 2003)
- 600-MHz ARM Cortex™-A8 Core – High-performance processor that is used in a wide range of devices from high-end smartphones to netbooks. It generates no heat and is fanless (PLC, 2007).
- Integrated L1 memory for ARM CPU (16kB I-Cache, 16kB D-Cache, 256kB L2) and On-Chip memory (64kB SRAM, 112kB ROM) – The memory for the core, this space is used to write the boot loader (see section 4.3.1. Embedded Development).

The OMAP3530 also includes a PowerVR SGX530 GPU. This is a graphics module compatible with the standard Open GL ES libraries used in the Android. The GPU provides 3D accelerated graphics capabilities for the user with fast 2D/3D rendering and enhanced user interfaces. This GPU unit is found in many smartphone devices and is effectively optimised for mobile use providing high performance with lower power and memory consumption (PowerVR, 2009).

The OMAP3530 is slightly higher powered than the processor used in the Motorola Droid, which runs Android OS 2.0 (Conner, 2010). The specifications for both of the devices are given below:

- **AIVI** – OMAP3530 - 720 MHz ARM Cortex A8 + PowerVR SGX530 GPU + 520 MHz C64x+ DSP + ISP (Image Signal Processor)
- **Motorola Droid** – OMAP3430 - OMAP3430 - 600 MHz ARM Cortex A8 + PowerVR SGX 530 GPU + 430MHz C64x+ DSP + ISP (Image Signal Processor).

The OMAP3530 surpasses the minimum requirements for a basic Android device and has superior processor capabilities in comparison with the Motorola Droid, which is considered to be a higher-end
Android powered phone. The minimum requirements, as suggested by Google, are that the device must run at least 200MHz with 32MB of RAM (CNet News, 2007).  

Memory

The AIVI system will have 256MB DDR SDRAM and 256MB NAND Flash. This provides sufficient RAM and internal memory for the majority of applications. The device is also capable of expandable memory, providing interfaces for external storage such as SD Cards and USB storage devices.

- 256MByte DDR SDRAM, 166MHz
- 256MByte NAND Flash, 16bit
- Expandable memory; USB Flash disk and SD Card support

Audio and Video

The audio and video interfaces supported by the device are given below. These interfaces allow support for a microphone input and a two-channel audio output to provide basic audio input/output support for the system. The audio input provides further interaction by facilitating voice-inputs to the system (using voice recognition technologies). The audio output provides users with entertainment capabilities as well as audio feedback during system use; this avoids driver distractions and enables audible feedback for navigation applications (e.g. turn-by-turn instructions) and other on-road application use.

- An audio input interface
- A two-channel audio output interface
- A TFT LCD interface, 24-bit true colour (resolution supporting up to 2048 x 2048)
- 7-inch TFT LCD display - Resolution 600 x 480.
- 4 line Touch Screen

The LCD screen support allows for the use of a TFT\textsuperscript{27} based touch-capable LCD screen. This provides a visual and haptic interaction with the system. The LCD screen selected for the device is a 7-inch touch-screen display and is upgradable to a larger size or a multi-touch support display.

Peripherals

Peripherals are external hardware that expands the system beyond the board and core components. System peripherals allow further device capabilities and are attached using standard interfaces such as Serial PS/2 port or USB.

- **GPS Receiver** – Used to retrieve a constant stream of GPS coordinates of the current location to the system.
- **Reverse Camera** – Attached at the rear of the vehicle, the Reverse Camera provides a bumper level view behind the vehicle while reversing.
- **3G Modem** – Provides internet connectivity.

\textsuperscript{26} The minimum requirements were stated in an interview with Google executive Andy Rubin (CNET News 2008). There are no official statements regarding minimum requirements for any Android version released by the Open Handset Alliance or Google.

\textsuperscript{27} Thin-film Transistor (TFT) – It is a variant of LCD screens that is widely used in mobile phones, computer monitors and even TVs. It improves image quality in regards to addressability and contrast.
• **Microphone** – Provides speech input.
• **USB Hub** – Provides capabilities for external storage and USB-based peripherals.

4.2.2. Architecture of the AIVI system

Figure 22 shows all the necessary hardware components and how they are integrated.
Figure 22 shows a conceptual layout of AIVI components installed in a vehicle. Please note that the figure is not to scale (in size or precise location of the components).

**Current AIVI Components**

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AIVI Centre Console</td>
<td>This is the centre console used for the IVI system. This includes the AIVI motherboard and processor, as well as the touch-screen display.</td>
</tr>
<tr>
<td>3G Modem</td>
<td>Used for Internet connectivity. The component is interfaced via USB (connected to the USB Hub as shown).</td>
</tr>
<tr>
<td>USB Hub</td>
<td>Used to interface several components, including the 3G Modem and Reverse Camera as well as external storage provided by the user (such as flash memory sticks). The USB Hub is also used for the proposed Wi-Fi Hotspot component. The USB Hub has an independent 5V power-source to power USB components. This prevents drawing power from the AIVI Centre Console's own power-source and allows more peripherals to be connected which have need of the extra power requirements.</td>
</tr>
<tr>
<td>GPS Receiver</td>
<td>Used for receiving GPS data of the current location. The GPS is connected directly to the centre-console (motherboard) of the system via Serial PS/2 interface.</td>
</tr>
<tr>
<td>Reverse Camera</td>
<td>Reverse Camera is interfaced via USB (connected to the USB Hub). This camera is placed at the rear of the vehicle with the cable running along the vehicle.</td>
</tr>
<tr>
<td>Microphone</td>
<td>The microphone is placed in a fixed location that is directly accessible to the driver. It is generally placed on the rear-view mirror (as demonstrated by common IVI systems such as the Ford SYNC). The microphone is used for speech input for the AIVI system. It is also used for speech recognition and voice actions, as previously discussed.</td>
</tr>
</tbody>
</table>
## External Components

- **Vehicle Speakers**
  Emits audio to the vehicle. This is powered by the vehicle sound amplifier.

- **Vehicle Amplifier**
  Allows an interface for the AIVI system and other sound sources to use the vehicle speakers (by 3.5mm jack to RCA connector). Provides a power-source for the speakers and controls the power output. The amplifier is typically found within the vehicle components.

- **Portable Devices**
  Used by vehicle occupants, these devices can be laptops, smartphones and other devices that can use the vehicle’s Wi-Fi hotspot to connect to the internet.

## Actors

- **Driver**
  The vehicle driver has direct interaction with the AIVI system via touch-input and the microphone (voice actions).

- **Vehicle Occupants**
  The front-passenger also has direct access to the AIVI system, but all other vehicle occupants can only view the display of the AIVI system.

  Vehicle occupants’ interaction with the AIVI system is covered further in Chapter 6: Future Work.
4.3. Software Implementation

Implementation for the Android-based IVI system is broken up into several components. These are written in u-boot language, C/C++ and Java. The following is a brief description of each component and the parts that were implemented for these components:

1. **Embedded Development** – This component is implemented to set-up the development board used (OMAP3530). This will involve writing to the device’s memory and setting up the device to allow the kernel and Android to boot.

2. **Kernel modification** – Patches need to be applied to allow the kernel to run on the desired board (OMAP3530). Certain kernel components need to be modified to load the appropriate modules for the hardware components used in the AIVI system.

3. **Android O.S. Core** – The Android core components are already implemented by the community. Patches and modification were used to load the operating-system onto the OMAP board.

4. **Android Hardware Abstraction** – Components that are not supported by the Linux kernel require certain interfaces to be implemented for the device to function appropriately. These include:
   a. GPS Receiver
   b. 3G Modem
   c. Reverse Camera

5. **Android applications** – The implementation of the AIVI CarHome application

This section will give a brief overview of these implemented components and supply any core code required to clarify the implementation.

The source-code for these implementations can be found in the code repository outlined in Appendix J.

**Embedded Development**

The embedded development that was required allows us to load the necessary applications that lead to booting the Android operating-system. This requires writing certain applications to the board memory (TI OMAP3530). These applications are bootstrap components, known as a bootstrap chain. The chain of applications leads other larger applications to eventually boot the kernel and Android system. These are required to initialise the hardware, test the components, and eventually launch the operating-system. The order for the ‘bootstrap’ chain is shown in Figure 23.

![Bootstrap chain](image)

**Figure 23 - Bootstrap chain**

1. **X-Loader** – This is a small application, less than 64KB, and is stored in the internal static RAM of the OMAP board (loaded by ROM code). X-Loader is only small enough to boot essential
components because it is restricted by the internal static RAM’s limit of 64KB. X-Loader’s main job is to lead to the second stage bootloader, the U-Boot, to provide us with the larger boot loader which gives features (such as accessing SD-card etc.). X-loader is open-source and developed by the community

2. **U-Boot (Universal Bootloader)** – is the second-stage boot loader. This is an application that provides more functionalities than the X-Loader as it is loaded into the device’s flash memory. This allows us to initialise and test system resources as well as load certain peripheral components such as Ethernet. The u-boot is used to write (flash), the kernel and Android O.S. images to the NAND memory of the device. These images are generated once the source-code of the kernel and Android are compiled. Furthermore, the u-boot allows us to setup environment variables for the device, i.e. specifying display type, and file-system location (whether it should be loaded from SD-card or NAND memory of the device). U-Boot is open-source and originally developed by DENX (Denk, 2004).

3. **uImage (Linux Kernel)** – The uImage is generated from the compiled source-code of the Linux Kernel. The Kernel provides us with the core computing components for the operating-system. See section 3.2.1 for more details.

4. **User-space (Android operating-system)** – Once the Android source-code is compiled, it provides a system image that can be written to the flash memory from u-boot. This is loaded once the kernel completes booting. More details about the Android operating-system and architecture are discussed in section 3.2.1.

These steps are necessary for allowing the development board to run a basic bring-up of the Android operating-system on to the custom board. These steps are required when developing any Android based devices.

Please see a more detailed explanation of embedded development including sample code that was used for this project in Appendix D.

**Kernel Modification**

There are several components that allow Android to function on a standard, unmodified Linux kernel. These changes are documented appropriately and developed closely with the Android team. The changes can be tracked via open-source mailing lists and eLinux wiki (Embedded Linux Wiki, 2010).

The Android kernel will require further modifications to allow certain modules and components to be enabled for various board types, these include; several OMAP boards, x86 systems as well as other ARM based systems. Most of these systems have the specific modifications required distributed as patches by the open-source community.

**Kernel Patches** - Specific patches are required; the TI OMAP3530 development board will require several components to be functional. These include sound components (audio and microphone), display and touch-screen, and several other minor tweaks specific to the board configuration. These patches can also be obtained from open-source communities, specifically an OMAP3530 development mailing-list.
**Linux Modules** - The AIVI system required several kernel modules to be enabled for certain peripherals to be functional. These peripherals include the 3G modem, and the camera hardware used for the Reverse Camera. The Linux modules are discussed in detail within each component implemented in Chapter 4.

**Android O.S. Core**

The Android O.S. Core components are retrieved from the AOSP (Android Open-Source Project) repository. Patches are then applied to allow it to function appropriately on OMAP3530 board.

More details on how to retrieve and compile the source-code used for the AIVI system is described in detail in Appendix D.

**Android Hardware Abstraction**

The hardware abstraction layer allows us to implement any necessary components that are not directly supported by the Linux kernel (see Figure 4 in Chapter 3). In our case, the GPS module must be implemented. This section will contain all the information necessary for implementation of the GPS library for Android 2.1, Texas Instrument OMAP3530 Board, and GPS hardware Module.

**4.3.1. AIVI GPS Development**

**GPS Hardware**

The GPS receiver used for the AIVI system is known as GPS8000-S (YoungTek Corp., 2009). GPS8000-S is connected by a serial connection (RS232 Level\(^28\)). The Linux kernel communicates with the device using a file connected to a standard input (located at ‘/dev/ttyS0’). The data is transmitted by a serial connection and is received in a stream at a specific baud rate\(^29\). Data is also updated at a certain update cycle rate.

**GPS8000-S (SiRF Star III GPS1155) Specification:**

- L1 band (1575.42MHz) Receive Frequency
- 20-channel parallel receiver
- SiRF Star III high performance GPS Chipset, GSC3f/LP
- Operating Temperature: -40 °C to +85 °C
- Storage Temperature: -40 °C to +85 °C
- Output data format: NMEA0183V3.0
- Operating Voltage: 1.8V/3.3V
- Data Update cycle: One second
- Cable length: 5M
- Start Time
  - Hot Start (approximately): 1s
  - Warm Start (approximately): 35s
  - Cold Start (approximately): 42s

---

\(^{28}\) **RS-232** (Recommended Standard 232) is a standard for serial communication that specifies signal voltages, signal timing, signal function, a protocol for information exchange, and mechanical connectors.

\(^{29}\) **Baud Rate** is the number of distinct symbol changes per second over a transmitting medium. The correct baud rate must be set on the receiver according the transmitting device to view the data appropriately.
Serial configurations for the AIVI GPS system:

- Serial Port: TTYS0 - for AIVI
- Baud Rate: 9,600Bd
- Data Bits: 8
- Parity: None
- Stop Bits: 1
- Handshake: None

GPS Data

Data is formatted for NMEA (National Marine Electronics Association) standard output. NMEA is a data specification for communication between electronic devices; it was first designed for standard use for marine electronic devices such as the echo sounder, gyrocompass, autopilot, and GPS receivers. Each NMEA sentence provides a protocol which defines GPS attributes. Each protocol has its own interpretation of GPS data and is defined in the NMEA standard (Langley, 1995).

An example of a typical NMEA output (retrieved from AIVI’s GPS8000-S unit):

```
$GPGSA,A,3,03,06,11,08,19,25,23,07,,,,,,1.8,1.2,1.3*3E
$GPRMC,012358.000,A,2234.2781,N,11408.0685,E,0.00,,261109,,,A*7C
$GPVTG,,T,,M,0.00,N,0.0,K,A*13
$GPGGA,012359.000,2234.2781,N,11408.0685,E,1.08,No data 1.2,76.3,M,1.7,M,,0000*7E
$GPGSA,A,3,03,06,11,08,19,25,23,07,,,,,,1.8,1.2,1.3*3E
$GPRMC,012359.000,A,2234.2781,N,11408.0685,E,0.00,,261109,,,A*7D
$GPVTG,,T,,M,0.00,N,0.0,K,A*13
$GPGGA,012400.000,2234.2781,N,11408.0685,E,1.07,1.2,76.3,M,1.7,
No data
```

The NMEA sentence is led by a dollar-sign ($) followed by the protocol name then the data. Some protocols might repeat certain information from other protocols but will also provide new data. The GPS8000-S unit provides five protocols which include (Yuanliang, 2003):

- **$GPGGA** - Global Positioning System Fix Data. Time, position and fix related data for a GPS receiver.
- **$GPRMC** - Recommended minimum specific GPS/Transit data.
- **$GPGSV** - Satellites in View, and information about satellites that the GPS unit might be able to find.
- **$GPGLA** - GNSS DOP (Dilution of Precision) and Active Satellites
- **$GPVTG** - Track Made Good and Ground Speed.

The GPS data can include longitude, latitude, speed, a GPS timestamp and other information according to the protocol’s definition. The NMEA sentence is concluded by a checksum (always begins with a ‘*’ symbol). This checksum ensures that the data is not corrupt.

Please see Appendix F for a detailed understanding of the protocols and an explanation of each parameter provided.
GPS Integration with the AIVI system.

The official Android source-code provides no implementation of GPS products, only code for an emulated GPS hardware. For this research, an Android GPS library was implemented for the system to provide the navigational capabilities for AIVI. The library was based on the emulated GPS sample-code provided by AOSP, rewritten for the specific hardware configuration used for the AIVI system, and GPS hardware used.

Android’s GPS is defined in the Hardware Abstraction Layer (HAL) of the android device. This is the user space C/C++ library layer which allows a standard interface between the Android platform logic from the hardware interface (Open Handset Alliance, 2008b). The interface header is defined at ‘include/hardware/gps.h’. In order to integrate GPS with Android, a library must implement this interface. Once the library is built, it is stored in with the Android libraries as a ‘.so’ file. These files can be accessed by the Android framework. In this case, the GPS libraries are used by the ‘GPSLocationProvider.java’ (see Figure 24). Applications then use the framework to retrieve GPS data as required without worrying about the hardware. Therefore, any Android application, including Google Maps and any third-party applications from the Android Market, can use the GPS system without any modifications to the framework or the application source itself which would normally be expected. Please refer to Figure 24 for a better understanding of the GPS components for the Android framework and the integration with the GPS hardware.

For more details on the implementation of the GPS library, please refer to Appendix F.

30 Android libraries are stored as ‘.so’. These libraries provide an interface for the framework to communicate with the hardware components. This is similar to the ‘.dll’ files in the Microsoft Windows system.
Future Work for AIVI GPS

At present, the GPS provides all data necessary for the applications, so the hardware is good and practical when it comes to navigation. However, considerations must be made with regards to GPS power usage and whether the GPS should be on or off. The GPS might drain power once the vehicle is turned off, so one must consider if the GPS should be kept on while the device is sleeping as it might deplete the vehicle’s battery. This is important because GPS hardware has a warm and a cold start time of 35 and 42 seconds. This is the minimum time required for a GPS lock once the device has been turned off for a certain period of time. The GPS lock is dependent on signal strength, satellite geometry, and how many satellites are in view. Essentially, if the GPS is powered off, then we need to spend several minutes for the GPS unit to get an appropriate fix on a satellite; this time varies on weather conditions and satellite availability. Appropriate power management tests need to be taken and put into real testing, including power consumption tests, and analysis of the most practical approach to this solution.

4.3.2. AIVI 3G Modem Development

The Vodem (ZTE K3765-Z) is a USB 3G modem that allows users to connect to a 3G connection from any USB interface. It is officially supported in Windows and Mac desktop operating-systems. For the Vodem to function appropriately for the proposed AIVI system there are several steps needed to be taken (ZTE Corp, 2009).

ZTE K3765-Z (Vodafone Vodem Stick Pro) Specification

- HSDPA/UMTS (900/2100MHz)
- GSM/GPRS/EDGE (850/900/1800/1900MHz)
- HSDPA connectivity at up to 3.6Mbps.
- Micro SD Memory Slot; acts as Memory stick (up to 8GB)
- Smaller and more compact; Stable, Reliable, Solid Body.
- Compact USB stick form-factor: 87.5 x 26 x 11.5 mm
- Zero CD Technology: No CD required; Auto plug-and-play.

1. Enable Modules in Linux kernel

This requires the kernel to enable the modules/drivers\(^{31}\) required for the 3G modem to be recognised by the kernel. This will do all the necessary lower level data transfer and expose the TTY’s\(^{32}\) that are used to establish a connection.

2. Switch modes to enable RIL

3G Modems have two modes; one for RIL (radio interface layer), while the other is used for the Zero CD feature. ZeroCD must be disabled manually; this is done by sending an AT command to the modem thus disabling the ZeroCD mode. This command can be sent via a serial terminal, by the development host machine.

31 The standard ‘options’ module is out-dated. A more updated options driver was used for AIVI. See: [http://www.spinics.net/lists/linux-usb/msg35976.html](http://www.spinics.net/lists/linux-usb/msg35976.html).
32 TTY is used as a communications terminal in Linux platforms.
3. Set up and Connect

The network interface that the proposed Android-based IVI system uses is PPP (point-to-point protocol). Android uses a Linux based PPP daemon (pppd) that allows for the establishing of internet connections via a serial modem (the 3G modem). The following documented scripts are used to establish an internet connection using the 3G modem.

There are three important things to consider when establishing a 3G connection:

- Point-top-point daemon (PPPD) script.
- Connect Script.
- Automate connection on start-up.

Point-top-point daemon (PPPD) script

The PPPD script provides the modem with the necessary information about the modem configuration (such as speed, authentication method, and the interface used for communication).

The scripts set up Vodafone NZ, as a peer, for the connect script to use. The PPPD can store multiple peers; these are stored as files which can be located in the file-system. The connection uses the syntax ―pppd call vodafone‖ allowing the PPP daemon to use the connect script on the ‘Vodafone’ peer (‘vodafone’ file within the peers directory).

Connection script

The second part of the connection is the actual connect script. This is used to establish the connection and uses commands to communicate with the modem. The Hayes command set (more commonly known as the AT Commands) is a command set used to complete certain operations for modems (A. ZTE Corp, 2007). These include basic commands for dialling, terminating a connection, and various other settings. Furthermore, these provide feedback for the application on whether a connection was established successfully, or the exception that was raised during the connection. These commands are sent via the PPP daemon, as described above, using the specified interface.

Automate connection on start-up

Lastly, the connection must be automatically called so that the 3G is connected on boot. This is done using the Android ‘init’ application (an application that runs once the Android operating-system boots into desktop). The ‘init’ commands are specified in the ‘init.rc’ file in the device’s root folder.

The code in Appendix G (Figure 49) starts the ‘ppp’ (point-to-point protocol) service with the parameters ‘call vodafone’ (call a specified peer). The service is also given permissions (service executed as ‘root’ and under the group ‘system’ and ‘radio’). These are permissions structure the Linux kernel uses.

The PPPD, Connection and Automation scripts are attached in Appendix G.

These steps will provide internet connectivity to the AIVI device using the Vodem (ZTE K3765-Z) 3G Modem.

---

33 Although the modem uses a USB interface, the USB emulates a serial connection.
4.3.3. **AIVI Reverse Camera Development**

The AIVI Reverse Camera gives enables a rear bumper view behind the vehicle, providing a digital video of the obstructed areas behind the vehicle. The camera that was used for the AIVI system includes the following specifications (Logitech, 2011):

**Logitech Webcam C120 Specification:**
- Plug-and-play setup (UVC) CMOS webcam
- Video capture: 640 x 480 pixels
- Still image capture: up to 1.3 megapixels
- Frame rate: Up to 30 frames per second.
- Hi-Speed USB 2.0 certified

The camera used is the Logitech C120 USB camera. It is a UVC (Universal Video Class) camera that is normally used as a webcam. This camera was chosen as it gives us an easy to implement API set (V4L2 and UVC) that is supported for all platforms that provide the UVC driver. Therefore, it requires no specific driver from the manufacturer for it to be supported by the Linux kernel.

Figure 27 illustrates the camera structure in the Android operating-system:

---

**Figure 27: Android Camera Sub-System (Android Open-Source Project 2010).**
The steps used to implement follow a similar structure to the 3G modem, they include:

- Enable kernel modules
- Camera setup
- V4L2 Camera Implementation

1. Enable kernel modules

Like most of our hardware components, the first step taken is to enable the necessary modules and drivers that the kernel requires for the hardware to be recognised by the kernel. The kernel support is an important step to provide the hardware abstraction needed by the Android operating-system.

The modules required are:

- USB Video Class (UVC) – Standard Linux driver used for UVC based video devices such as the Logitech C120. UVC is becoming a standard driver that is supported across multiple platforms (Mac OSX, Windows and Linux). It does not require third-party driver support from manufacturers and is supported by the Linux kernel itself (i.e. different cameras from different manufacturers can be used as long as they have UVC support).

- V4L (Video4Linux) USB Drivers – This module provides the video driver used for V4L and V4L2 compatible devices.\textsuperscript{34} V4L is a widely supported camera API (application programming interface) that is used in most modern cameras in the Linux platform. The Android operating-system will use the V4L API to retrieve images captured from the camera and convert raw data into a compressed format.

Once these modules are enabled, the camera should be recognised by the kernel and a corresponding video node is generated (/dev/video0).

2. Camera Implementation

The Android operating-system provides a ‘CameraHardwareInterface’ that must be implemented to expose the camera data used by camera services (for the application framework). Once implemented correctly, any applications accessing camera hardware will be able to use this class transparently without any further modifications to the actual application (see Figure 27: Android Camera Sub-System (Android Open-Source Project 2010)).

The camera object used is a V4L2 camera object which provides the necessary pixel format and colour conversions used to expose the camera images to the software. This normally includes the saving of images into a compressed format (such as JPEG), but we are at present only interested in retrieving the video feed to provide us with a video stream from the Reverse Camera in the AIVI system.

\textsuperscript{34} V4L2 is a second revised version of Video4Linux. It is backwards compatible with V4L (Video4Linux 1).
Sequence Diagram

The following is a sequence diagram that illustrates the interactions between the classes used for the camera implementation (the implemented classes and methods for this research are highlighted in bold):

Figure 28 - Reverse Camera: Sequence Diagram
Sequence Diagram Description
To better understand the sequence diagram in Figure 28 we divide the operational details of the camera into: starting the camera and stopping it.

1. Starting the camera
   - CameraHardware class is created once the camera service calls the camera hardware to be open. This is triggered when a camera application starts.
     - The file descriptor is opened with read-write permissions (/dev/video0). This is the node that is generated from the camera hardware (in the Linux kernel). It is the channel of communication between hardware and software.
     - Initial parameters are set for the camera hardware, these include:
       - Frame size: 680x480 pixels.
       - Frame rate: 30 frames per second.
       - Preview Format: YUV422SP.
     - Parameters are queried and returned to the application. This is usually used to setup the canvas. Setting parameters, from the application, is currently not implemented as it is not relevant to our study. The default parameters are always overwritten if the application attempts to change these parameters.
     - Once parameters are retrieved, the application can attempt to ‘start’ the video streaming of the camera hardware. Retrieving the video stream is referred as ‘previewing’ (i.e startPreview).
     - On preview, the V4L2 camera object is created and initialised
     - Start streaming method is called, and this will in-turn run a looped thread to retrieve image data captured from the camera hardware.
       - Data is retrieved from the camera according to the camera’s colour format (YUV422SP35).
       - Once retrieved, the colours are converted to a format that can be understood by the Android system (RGB56536).
       - This thread is looped until ‘stopPreview’ is triggered.

2. Stopping the camera
   - When the application exits the camera service calls the preview stop method of the CameraHardware class. This will initiate the clean-up process to safely stop the camera. This includes:
     - Stopping the looping thread that retrieves the camera data in the CameraHardware class.
     - Un-initialise the V4L2 camera object. This will clear the queue of waiting data to be processed.
     - Calling the ‘stopStreaming’ method. This will send a signal to the camera hardware to turn off streaming.
       - ioctl(fd, VIDIOC_STREAMOFF , $type);
     - Close the file descriptor (/dev/video0).
     - Call destructor for CameraHardware
       - CameraHardware deletes the V4L2 camera object
       - CameraHardware is destroyed (singleton).

35 YUV is a colour format standard used in most cameras for retrieving raw imagery data from image capturing devices.
36 RGB is a 16-bit color mode that evaluates to 65,535 colours.
Future work for AIVI Reverse Camera

These features are some of the future considerations for the Reverse Camera feature for the AIVI system:

- **Wide angle viewing**

  Obtaining a wide-lens camera provides users with a wider viewing pane. This is limited by the camera hardware itself; using a wide lens attached to the normal camera will provide the wide-angle support necessary. This requires no extra software considerations and is only a matter of replacing the camera with a wide-angle camera. The current camera is constrained to the viewing area it can provide, and it might not be enough in cases where users would like to see slightly to the side of the rear of the vehicle.

- **Proximity Sensor**

  A proximity sensor (proximity switch) can be added to support the camera by providing users with a digital reading of the physical distance between viewable objects and the rear of the vehicle. The sensors can provide data for the application to use, and the system can provide a visualisation on the screen that will appropriately prompt the user if there is a hazard.

  Proximity sensors can be interfaced to a Linux system by general-purpose input/output pins (GPIO). The output of the proximity switch is then exposed via special file ‘event’ interface by the Linux kernel (OMAPpedia, 2010). Lastly, a hardware abstraction library needs to be implemented, for the Android, to parse the data inputted by the system (from the events interface). The sensors are then exposed to the Android framework by the ‘SensorManager’ readily available to the Android framework.

  Figure 29 illustrates each component required and how the proximity switch would communicate with the AIVI system.

![Figure 29 - Proximity Sensor: Flow Diagram](image-url)
4.4. Application Design

The Android operating system, although built to be an all-purpose system, is most specifically used in handsets and mobile phones. There are currently no applications for dedicated in-vehicle Android devices that make use of the large 7-inch display (as of Android 2.1)\(^{37}\). This can prove problematic as users need an application to interact with directly without browsing through menus and application clusters of the standard Android Home and Launcher menus.

The proposed AIVI system comes with a custom application for easier navigation and a clear interface for in-vehicle use. This application will supply all the necessary components of the Android system with an easy-to-use interface that provides less clutter and guides users to the most relevant applications. The application also offers other components that will help drivers in understanding certain road and weather conditions when driving.

4.4.1. AIVI CarHome

AIVI CarHome is the application that was developed specifically for the AIVI system to make use of the 7-inch display and provide users with necessary and relevant navigation and information for drivers to use with minimal distractions. The homepage of the AIVI CarHome application is shown in Figure 30. It has been annotated by the author for illustration purposes.

[Image: AIVI CarHome Application]

Figure 30 - AIVI CarHome Application

\(^{37}\) Android does come with an application named ‘CarHome’ that is built for in-vehicle use, but it is very limited in functionality and optimised for the smaller phone display (3.5 to 4.5 inch display).
AIVI CarHome Features:

Navigation Slider

The Navigation slider provides easy, natural manipulation of the application panel. It displays applications that are ordered according to user desirability (as discussed in section 3.4.2.10.) and can be flung left-to-right and vice versa.

![Applications' default placements are ordered according to user desirability.]

When an application is selected, corresponding text will appear to provide the user with the Application name. Furthermore, the system gives voice feedback to inform the user of the application name.

Weather Conditions

The user is provided with weather condition details, as shown on Figure 30. Current weather conditions are collected using external web services (currently Google Weather web-services)\(^{38}\). The weather icon changes, according to the weather conditions, based on the pre-loaded set of icons saved in the system. These are included to minimise data transfer. Weather details are updated in accordance with a set duration inputted by the user, or updated on demand (when the icon is touched).

Weather conditions are also read back to the user, like most items in the application. When the user ‘taps’ a certain

---

\(^{38}\) Currently, Google Weather web-services are not officially documented.
area, the information in that area is read back to the user in human-readable format. For the weather conditions shown in Figure 32 the system will read back the following sentence:

\[ \text{The weather conditions are [PAUSE] Chance of Rain [PAUSE] The Temperature is [PAUSE] 13 degrees Celsius.} \]

The [PAUSE] is required for the system to read the sentence at an understandable pace.

This uses Android’s Text-To-Speech service (TTS) that can be achieved by the following sample code:

```java
private TextToSpeech tts;
public void onClick(View v) { //When Weather panel is touched.
    String speech;
    if(!tvTemp.getText().equals("N/A")){
        //Speech string is concatenated
        speech = "The weather conditions are. " + 
v_friend_to_his.getTempCelcius() + 
        degrees celsius";
    }else{
        speech = "Sorry, there is no weather information available.";
    }
    //TTS plays speech according to the string built.
    tts.speak(speech, TextToSpeech.QUEUE_FLUSH, null);
}
```

Figure 33 - Sample Code: Text to Speech used for weather information.

Road Information

This feature provides information about the road that the vehicle is currently travelling on. The application will provide information on:

- Current speed
- Current road name.
- Current suburb.
- Current speed-limit on the particular road.

The system, using the GPS coordinates obtained from the GPS receiver, provides the current location data to an external web-service that returns up to date road information.

The speed-limit approach is presently problematic; there is currently no public database specifying speed-limits on New Zealand roads, so the approach used was to distinguish normal roads with highways.\(^{39}\) This way, we can assume that the speed-limit is either 50 or 100km/h, or otherwise specified which is saved into the database (for a certain road section). Please note: these are only

\(^{39}\) The type of road is determined from the web-service used to retrieve the road information data.
temporary assumptions used for the prototype of the experiment and do not always apply in all cases. This is used until a complete database of speed limits is retrieved. Users are prompted with this warning.

If the speed-limit is exceeded, a dynamic warning sign is used to indicate this and alert the driver that they have exceeded the legal speed-limit. This is conveyed by the ‘current speed’ label which blinks between the default colour (white) and the alert colour (red). This can be seen in Figure 35. A study shows that the use of dynamic speedometers will result in safer driving behaviour (Kumar & Kim, 2005).

![Figure 35 - Speed Limit Exceeded](image)

Colour alternates from red and white when the speed limit is exceeded. A verbal (speech) warning will sound informing the user that “you are exceeding the speed limit”.

The vehicle speed is calculated according to the difference of two GPS coordinates. This is automatically calculated by the GPS receiver and can be retrieved by the Android framework.

**Future work for AIVI CarHome**

The ability to completely customise the behaviour of AIVI CarHome is important. Users have different levels of expertise when it comes to using these systems. We would like users to customise the application appropriately according to their needs. Some of the customisation options for the future include:

1. **Control over safety features**

   Having control over the safety features of the system will take into consideration different levels of driver expertise. More advanced drivers might like options such as turning the speech alert warnings off completely; moderate users, rather than speech feedback, may want a more familiar subtle sound of a ‘ping’ which is similar to the implementations of some modern vehicles.

2. **Customisation over the navigation pane**

   This feature is expected as we would like users to be able to customise what can be displayed in the navigation pane (see Figure 31 - AIVI CarHome: Navigation Slider). This will allow users to add their own custom applications that they have installed in the system, and manually order the application hierarchy.

3. **Custom widgets displayed in the application**

   We would like to provide a flexible system for the users. We do not want them to be constrained to the widgets we have provided for them, which are the weather and road information panels and...
speedometer (see Figure 30 for the main display). Users might want to add different widgets. In the future, we would like to add more features, including the ability to retrieve data from other web-services such as traffic notifications, email notifications, and other ‘push’ notifications\(^{40}\) that the users might require.

### 4.5. Summary

In this chapter we discussed the hardware and software requirements of building an Android-based In-Vehicle Infotainment system. The details were covered and broken up into several components. These were:

- **Hardware Components** –
  This included all hardware components that were required in building the AIVI system. This includes all requirements from the board and processor to the peripherals such as the Reverse Camera and the 3G module.

- **Software Implementation** –
  This section covered all the necessary development work that was needed to build such a system. This includes all software components that needed to be modified and built (from the low-level core development, to the Android application development). These were implemented in the following order:
  i. **Embedded development** – Board development and writing to the device memory to be able to boot a built Android system.
  ii. **Kernel development** – The modifications and development that is required for the AIVI system.
  iii. **Android O.S. core development** – this includes the necessary development for the core operating-system that is required for the proposed AIVI system.
  iv. **Android hardware abstraction** – This includes the development of the integration of hardware components for the AIVI system, including 3G module, Reverse camera and GPS unit.

- **Application Design** –
  This includes Android application development and design. This mainly involves the user-interface development which the user will be interacting with. Considerations were made for the application to be used on-road.

In summary, this chapter described the design and developmental decisions made during the building of the Android-based In-Vehicle Infotainment system. This includes the complete integration of all system components with the Android system in an in-vehicle environment.

\(^{40}\) Push notifications allow applications to retrieve notifications and data from an external service whenever updated data is detected.
Chapter 5

System Evaluation

5.1. Overview

This chapter will look into how the proposed system was evaluated by end users. This was achieved by compiling a video demonstration of the features that were used in the AIVI system and asking for the viewer’s comments. Feedback was collected in the form of a survey.

This chapter will cover:

- The video demonstration (i.e. what the video covered)
- The feedback collected from the users in response to the video demonstration
- Analysis of the feedback collected

Notes regarding the System Evaluation:

The project’s main objective is to design and develop an AIVI system. Initially we had planned to conduct a practical system evaluation. This would give users a ‘hands-on’ experience of the system while collecting the necessary feedback as they utilized the proposed system. There was an issue in regards to obtaining the necessary ethical approval of such a study, since imposed risks had to be considered. The study required users to operate a vehicle which had the potential to put the participants in danger, perhaps leading to vehicle damage or harm to other road-users. However, the nature of the study that we had planned to conduct had ethical and regulatory implications. The study required obtaining the necessary ethical approval in order to employ human participants. Additionally,
the users required to operate vehicles which had the potential to put the participants and the other road
users in danger which bore regulatory implications. Hence, this method was replaced by conducting a
video evaluation of the system which demonstrated the relevant features and allowed users to provide
the feedback on the developed system in a safe environment.

5.2. Video demonstration

The video demonstrates main components of the Android IVI system. These components are
broken up into several categories given below.

The video can be accessed online at:
http://media.otago.ac.nz/44mq1wMWAcZ/wsHpwdMM/AIVI_Demo_3.wmv

The following are categories covered in the video demonstration:

- Multimedia
- Navigation
- Connectivity
- Reverse Camera\(^{41}\)
- Third-party Applications

The video demonstration showed the general use of the system from the driver’s point of view.
This was done using several scenarios such as navigating to a particular point of interest and running
certain applications associated with the features implemented.

Several features are demonstrated in the video, which include:

- AIVI CarHome – Custom interface designed for In-Vehicle use (see section 4.4.1).
- Reverse Camera – The camera placed at the rear of the vehicle that displays the obstructed
  view while reversing.
- Music, Video – Streaming or simply playing from a device storage.
- Internet – Using an internet browser and web-applications such as Facebook.
- Third-party Applications – Using applications such as Last.fm and TubeDroid.

Users were given a hyperlink to view the video online. They then provided the necessary feedback
regarding each category using an online form given on a separate link.

\(^{41}\) The ‘Reverse Camera’ is a functionality that was developed specifically for this system. It has not been
developed in any other Android-based device and this is why it is covered in its own category.
5.3. Video feedback

Users were asked to fill out a survey-form that was available once they had viewed the video demonstration of the proposed AIVI system. The survey asked for feedback regarding the satisfaction level of the viewer for each category in the demonstrated video. The users were also given the opportunity to provide feedback for each category in the form of open-ended comments.

The survey form offered a mechanism for the users to input their overall satisfaction with the system from the driver’s point of view, and also provided room for overall comments regarding the system’s general usability as well as any other feedback.

The survey form used is provided in Appendix G.

The following section, 5.4, outlines the results gathered from the surveys carried out.
5.4. Survey Results and Analysis

The survey was completed by 40 respondents. The survey results and analysis for each category of the AIVI system are described in the following section.

5.4.1. Multimedia Features

Description:

The video demonstration of the system established some aspects of the multimedia features present in the AIVI system. These included audio and video playback. The system also demonstrated that media can be streamed or downloaded, depending on the user’s preference.

Results:

![Interest of Multimedia features for the AIVI system](image)

**Figure 36 - System Evaluation: Satisfaction of Multimedia Features**

Analysis:

The majority of respondents (with 82.5% picking ‘Very Satisfied’) highly rated the multimedia features that were presented for the system. This reflects the desirability poll that was reported in section 0, and shows a high satisfaction rate for the multimedia features in IVI system with the different types of multimedia formats presented. This also shows that users like the choice between the type of media that they would like to use (e.g. H.264, MP4, 3GP etc.), as well as the method of delivery (whether it is streaming or downloading). This is apparent from the users’ preferences on whether they would like to stream or download (with 32% preferring to stream and 66% preferring to download, see section 3.4.3).
5.4.2. Connectivity

**Description:**

The connectivity features that were demonstrated covered various aspects of the system. The main connectivity demonstration was of the web-browsing capabilities. This was shown using an Internet browser that displayed various web-applications. Connectivity was used when demonstrating the use of some third-party applications. This included music streaming, news reader, and Google Maps to retrieve mapping data (place information, reviews, contact details and routing data).

**Results:**

![Satisfaction of 'Connectivity' features for the AIVI system](image)

**Analysis:**

The system evaluation showed that 45% of participants were ‘Very Satisfied’ with this feature, while a further 32% were ‘Satisfied’. There were some neutral and unsatisfied votes (15% and 7.5% respectively). The main complaints received from the users were about safety concerns. Section 5.4.8 provides more details regarding the comments about this feature.
5.4.3. Navigation

Description:

Navigation features were demonstrated using the Google Maps for Android application. This included using web-services to retrieve details about the desired point of interest. These details specified the type of place (restaurant, bar etc.) as well as reviews, contact details, and routing data (used for navigation). We also demonstrated the navigational features by physically driving towards the point of interest to portray how the system was behaving.

Results:

![Satisfaction of 'Navigation' features for the AIVI system](image)

**Figure 38 - System Evaluation: Satisfaction of the 'Navigation' features**

**Analysis:**

Navigation is a feature that generated a lot of interest from the respondents. 65% of respondents found this feature to be ‘Very Satisfying’ and a further 27.5% found it to be ‘Satisfactory’. This reflects the previous data that was gathered from the opinion survey (see section 3.4.2.3), and shows that respondents were generally satisfied with the demonstrated navigational features of the AIVI system.
5.4.4. Applications

Description:

Some sample applications were demonstrated in the video. These included some web-applications that were used to retrieve multimedia online (Last.fm), as well as a news reader (Pulse News Reader). Other applications also included Google Maps (Places and Navigation). The video demonstrated how users can access these applications by the implemented AIVI CarHome menu.

Results:

![Satisfaction of 'Applications' for the AIVI system](image)

**Figure 39 - System Evaluation: Satisfaction of Third-party Applications**

Analysis:

Third party applications overall generated a positive satisfaction rate (with 32.5% of ‘Satisfied’ participants, and a further 37.5% who were ‘Very Satisfied’). However, 30% of participants were neutral or were unsatisfied with such a feature (with 5% responding as ‘Very Unsatisfied’). This could mean either the video did not demonstrate enough third-party application capabilities, or that the users were simply not interested in external applications in IVI systems. Furthermore, it was hard to establish if the participants did not grasp the concept of external applications, since further investigation revealed that 83% of respondents who voted 3 or below (Neutral, Uninterested or Very Uninterested) had some experience with at least one smartphone operating-system (such as Android O.S, iOS, webOS or Blackberry) capable of running third-party applications. Moreover, participants did not provide any comments regarding these results.
5.4.5. Reverse Camera

Description:

The Reverse Camera was demonstrated by using it to display a rear view of the vehicle, in comparison to performing a manual head-turn to obtain the view. The video showed a comparison of the two viewing panes and how the reverse camera can be a safer option (in cases such as bad weather conditions). Figure 40 and Figure 41 are snapshots of the comparisons of these views (with and without the Reverse Camera). Figure 40 shows what the driver sees when they look through the rear windshield of the vehicle. On a rainy day it might be hazy and also the field view is low. On the other hand, Figure 41 shows that the field of view via the Reverse Camera is much larger since the camera is mounted on the bumper of the car. This will help the driver identify low-lying obstacles or hazards (e.g. shrubs, children). Please note that the picture quality appears to be bad due to the poor film quality of the demonstration. The actual image quality is 1.3MP for the Reverse Camera.

![Figure 40 - Vehicle rear view (head-turn)](image)
![Figure 41 – Reverse camera view](image)

Results:

![Satisfaction of 'Reverse Camera' for the AIVI system](image)

Figure 42 - System Evaluation: Satisfaction of the 'Reverse Camera'
Analysis:

Feedback concerning the Reverse Camera was generally positive; 87.5% of participants gave positive responses (with 55% selecting ‘Very Interested’). There was some concern regarding the functionality of the camera and the automatic launching of the camera application when the vehicle is in ‘reverse’. This is discussed in the open-ended comments that were collected (Section 5.4.8).

5.4.6. Overall system use

Description:

This category covers the overall system use of AIVI. This includes the general usability of the system as well as the user-interface presented. This allowed respondents give an overall rating of the level of interest generated from the demonstration of the system.

Results:

![Overall user satisfaction of the AIVI system](image)

**Figure 43 - System Evaluation: Overall satisfaction of the AIVI system**

Analysis:

Respondents of the evaluation showed a high satisfaction with the demonstrated AIVI system. This is apparent based on the overall positive rating of the system. Figure 43 shows that 67.5% of respondents were ‘Very Satisfied’ and a further 27.5% were ‘Satisfied’. Only 5% were “Neutral” towards the demonstrated system, however no respondents were ‘unsatisfied’.
5.4.7. Summary of poll results

<table>
<thead>
<tr>
<th>Feature</th>
<th>Unsatisfied</th>
<th>Neutral</th>
<th>Satisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multimedia Features</td>
<td>7.5%</td>
<td>2.5%</td>
<td>90.0%</td>
</tr>
<tr>
<td>Connectivity</td>
<td>7.5%</td>
<td>15.0%</td>
<td>77.5%</td>
</tr>
<tr>
<td>Navigation</td>
<td>2.5%</td>
<td>5.0%</td>
<td>92.5%</td>
</tr>
<tr>
<td>Third-party Applications</td>
<td>10.0%</td>
<td>20.0%</td>
<td>70.0%</td>
</tr>
<tr>
<td>Reverse Camera</td>
<td>2.5%</td>
<td>10.0%</td>
<td>87.5%</td>
</tr>
<tr>
<td>Overall</td>
<td>0.0%</td>
<td>5.0%</td>
<td>95.0%</td>
</tr>
</tbody>
</table>

Table 44 - Summary of poll results

In summary, overall there is a positive satisfaction rate regarding all the features demonstrated for the AIVI system. However, there are still a number of unsatisfactory responses to consider. The participant comments allowed us to investigate some of the criticism that was retrieved from this poll.

5.4.8. Participant comments

These comments were collected from each section of the survey. The comments are open-ended, anonymous, and optional for respondents to freely express their opinions about the demonstrated AIVI system. However, please note that only selected comments are discussed in this section, since several were similar or did not provide adequate details to be reviewed fittingly.

For all the comments gathered from the system evaluation, please see Appendix I.

Multimedia:

"I’m worried about playing videos while driving.

Playing of motion videos was an issue that was raised previously (see section 3.4.2.2). The system has security restrictions imposed to promote driver safety. This has been achieved by disabling any video playing application while the vehicle is in motion, only re-enabling them when the vehicle is stationary. Since there is a slight inaccuracy in regards to what is considered a ‘stationary speed’, the system is set to 5km per hour as the cut-off point for video applications.42 In the future, users will be able to customise these security settings according to their preferences with default security measures set (parental controls and passwords). See Chapter 6 on Future Work for more details.

"Will this support DIVX and other video formats?"

Multiple video formats are supported, some by default. However, more specific ones are not supported. This issue can be overcome by installing third-party applications that will provide the

---

42 The value was used due to the margin of error for the GPS device used to retrieve the vehicle’s current speed.
necessary codecs to decode different video formats. An example of this type of application is RockPlayer (Redirect Intelligence Corp, 2010).

"Would like to see this synchronise with my personal multimedia devices…
Like iPod, my phone etc."

With the current implementation of the system, synchronisation\(^{43}\) of external multimedia is only done via USB connection (or by a cloud-based service). Expansions to the synchronisation features will be included in the future works of this research. This will allow users to synchronise to their favourite multimedia device using various methods of data synchronisation (such as Wi-Fi and Bluetooth).

**Connectivity:**

"Not sure if the keyboard is practical while driving."

Users are able to use the ‘Voice Actions’ feature of the Android system (but this was not demonstrated in the video). This allows users to input voice commands and using the Android’s cloud-based speech recognition engine; the user will be able to interact with the system by voice (see 3.3.3 for more details).

The keyboard is available to give users a more familiar method of input and is more relevant to when the vehicle is stationary.

**Navigation:**

"One concern, will this always use data?"

Users are not limited to data consuming applications. Google Maps does use online services to retrieve mapping data. However, there are applications that can be installed on the system that download maps and store them locally; therefore there is no need to use additional data. AndNav2 is an example of an offline navigation application that is free and open-source (Gramlich and Neis 2009). This depends on the user’s preferences.

**Applications:**

"Will I be able to load applications from external storage (USB etc.)"

Android applications are stored as APK file-formats. These can be transferred and installed onto the system using any external media. This same applies to any Android device.

---

\(^{43}\) Multimedia synchronisation allows your media, such as music, to be shared with multiple devices (or an online cloud-based service).
Reverse Camera:

Would be nice if the reverse camera started automatically

Future considerations have been made for the Reverse Camera to start automatically once the vehicle is detected to be in the ‘reverse’ gear. This will instantly display the ‘reverse view’ from the camera thus minimising the time for starting and stopping the application.

Overall system use:

Thats crazy. Very distraction while your driving

Safety is an issue that has been raised by a few participants. The research looked into various ways that we can tackle these problems while at the same time providing a practical IVI system that is informative to the user. There are different approaches for the types of input and output considered for the system. These include voice recognition as well as voice feedback (where the system ‘speaks back’ to the user, relaying the information displayed on the screen).

Applications, such as Edwin for Android, are available that take advantage of these techniques where the user can ask the system a question, and the system replies with an answer (Neureau, 2010). Questions can include: “What is the weather?”, “Compose Email - Dear John – I am running late, meet me in 10”, “Where is the closest movie theatre?”, and “Get directions to Octagon, Dunedin”.

These techniques help users to retrieve the necessary data with minimum interaction and thus concentrate on the road.

What about making phone calls? Skype?
I hope it will support handsfree kit.

Future work will look into integrating more hardware components, such as Bluetooth and Wi-Fi, which allow for different means of communication with the vehicle occupants’ own devices. This includes phone contact synchronisation and the ability to make calls directly from the AIVI console by the Bluetooth or Wi-Fi connected device. This will also allow hands-free calling as the microphone is placed in a fixed position and the speakers are connected to the vehicle’s standard sound system.

Skype (and other Voice-over-IP) services are functional as of the current implementation. The services use the 3G connection that is presently available and only require users to download and install the associated applications separately to the AIVI system.
5.5. Overview of the system evaluation

In summary, the system evaluation provided us with the feedback necessary to understand the level of satisfaction of our respondents. It also provided us with the necessary feedback to further improve the system and take into consideration some aspects that were not previously considered. Some of these aspects include:

- Further considerations for safety features (e.g. disabling video, using Edwin for Android)
- Provide synchronisation features for the user’s current devices
- Automatically starting the Reverse Camera application when the vehicle is detected to be in reverse.

Limitations of the Video Feedback

There was some limitation with the system evaluation that was conducted. The video demonstration was posted in various forums and video hosting sites, and users had the choice to view it or not. Furthermore, the users also had the choice whether to submit feedback. The video had 406 views, but only 40 viewers chose to respond to the survey (statistics recorded as of 9th of March, 2011).

The respondents were mostly of the 18-24 (62%) and 25-39 (34%) age groups, while older age-groups (40+) accounted for only 2% of the total respondents. Therefore, we could not get an accurate figure on the interest level that was gathered for the older generations towards such a system.

The technique that was used to retrieve the data was still necessary to avoid any bias influence, and to give respondents the freedom to watch and appropriately analyse the video without the intrusion of the researcher being on-site.
Chapter 6

Conclusion & Future Work

6.1. Overview

This research examined state of the art infotainment systems and established that there are missing requirements and features which everyday users demand. Users are becoming more dependent on technology and we have found that the infotainment industry has not been able to keep in pace with today’s user requirements for infotainment systems.

Our research led to the implementation of an Android-based In-vehicle Infotainment system (AIVI) that addressed some of the shortfalls and challenges faced by the infotainment systems of today.

This section will conclude the research by providing an overview of:

1. The current state of infotainment systems and the challenges that were identified.
2. The process of obtaining the requirements of the system and the implementations that were required to achieve the tasks.
3. The current state of the implemented system (AIVI) and how the system was evaluated by end users.
4. The future work planned for this system.

This chapter will also provide an understanding of how this research could affect the development of infotainment systems and the factors manufacturers should consider when designing a system for today’s technological requirements.
6.2. Problem Area

The research investigated the contemporary state of the infotainment industry by analysing and providing an extensive comparison of current, top-end models available in the infotainment industry. This included several candidates that are leaders in the infotainment sector: Pioneer, Ford SYNC, FIAT Blue&Me, Kia UVO, and the upcoming MeeGo IVI system. This comparison table allowed us to analyse and identify gaps in these systems, and enabled us to conceptualize how an Android-based system can address such issues.

The comparison diagram is given in Appendix C.

The issues identified are:

- **Lack of the ability to install third-party applications**
  - Third-party applications utilise the device’s in-built components and provide new functionalities for the same set of hardware.
  - The proprietary systems do not use a centralised application-base (e.g. Android Market)
  - The proprietary systems do not provide a framework where third-party developers can develop applications for these devices.

- **Lack of a standardised open-source platform**
  - An open-source platform would allow each manufacturer to use the same platform (software platform) and customise it according to their requirements. The advantages of a standardised open source platform include the following:
    - It allows manufacturers to implement specific hardware designs (from their preferred suppliers).
    - It gives the ability to apply their custom user-interface and other functionalities to create a competitive advantage.
    - Manufacturers do not need to build a new platform from scratch and can concentrate more on improving and extending their hardware, all the while using a trusted software platform.
  - The platform would allow for the use of an already existing and vast range of third-party applications. This eliminates the need for application developers to rewrite their applications to support specific devices.
  - A standard-platform would give end-users a familiar, well established platform that is widely supported. Thus, end-users will have little or no need of additional training to familiarise themselves with the platform when moving from one manufacturer’s device to another.

- **Lack of Internet**
  - There is a high demand for users to be connected with the wider world while they are on-the-move. Since more data and multimedia are now stored in the cloud, there is a need for infotainment systems to be connected to the Internet. The driver can be notified of traffic, weather, and road conditions which are available as web-services.
  - This also provides better productivity as connectivity will enable commercial companies to track vehicles and provide data to assist in supply chain management, also facilitating a better asset utilisation.
These issues discussed above led to the development of the Android-based infotainment system (referred to as AIVI in this thesis) and the use of Android operating-system as a platform for infotainment systems.

6.3. Solution and Implementation

Android was identified as the platform of choice for addressing the issues that were recognized. Implementing an Android system in the form of an infotainment system allowed us to use its flexibility and openness to an advantage. It allowed us to implement the new features required by modern infotainment systems (e.g. multimedia support and turn by turn navigation). It also provided the features that are deemed to be of importance in the mobile world, such as connectivity and the ability to install third-party applications.

The implementation process was broken up into three main components:

- **Embedded Development**
  - Allowed us to load the necessary components required by the Android operating-system to run on our own custom hardware setup. This mainly involved the porting of the Android kernel (core components) to the new hardware set-up.
  - The embedded development also allowed us to setup the touch-screen to be compatible with the Android kernel.

- **Kernel Development**
  - The kernel is the core component in an Android operating-system. This allows hardware to communicate with the Android software (such as the Android framework for third-party applications to use).
  - Our custom hardware requirements needed several modules to be implemented for this research, which include the following.
    - **GPS Driver** – The GPS receiver is specifically made to be used in vehicles, as it has a strong range and a fast update speed. These features are much more important for a vehicle in motion than mobile phones. A GPS driver was implemented to connect and interface the device with the AIVI system.
    - **3G Modem** – The 3G module for AIVI is USB based. This type of modem is not supported by default in Android and a driver needed to be specified as well as certain scripts that needed to be written to allow it to be compatible with the point-to-point protocol daemon used by the Android platform.
    - **Reverse Camera** – This was a unique implementation that has not been previously developed in an Android platform; camera components had to be used as a Reverse Camera for the Android system. However, a driver also had to be implemented to support our V4L-based camera for it to be compatible with the Android framework.
- **Android Application Development**
  - A special application was developed and tailored for the AIVI system. This will be the default running application while no other particular application is being used in the AIVI system.
  - The application provided certain information to the driver, such as the following:
    - The current speed of the vehicle.
    - The current weather conditions.
    - The current road details including the speed limit.
  - The application also provided some safety by:
    - Providing an easy to use navigational menu that reduces the interaction rate of the driver with the system. The interaction time was reduced by displaying the applications using a ranked order. The ranking was measured by a survey where features were ranked based on desirability.
    - Providing a voice-feedback implementation for users to hear certain selections such as the weather conditions, the currently selected application, as well as current road details. The verbal feedback from the system eliminates the need for drivers to view the screen in order to understand the response from the system.

These three implementations were required to build the AIVI system that was developed as a part of this thesis.

**System Evaluation**

User feedback regarding the implemented features was collected and effectively analysed. The system was demonstrated by a short video shown to respondents. The respondents were then given a choice to complete the survey provided and give feedback regarding the implemented system’s functionalities.

The feedback from the survey showed a generally positive response for the functionalities demonstrated. Certain user concerns were raised which included the following:

- **More consideration for safety features.**
  - Feedback suggested that the system could prove to be distracting for drivers (e.g. using video and email features).

- **More synchronisation features.**
  - Users wanted to use their existing devices, which contained their media and contact details. The users wanted this data to be automatically synchronised with the AIVI device.

These issues can be considered as a part of the future work of this research.
6.4. Current state of the system

Currently the system has the following features:

**Completed**
- Complete Android operating-system running version 2.1 (Éclair)
  - Ability to run any Android third-party application compatible with version 2.1 and below.
- 7-inch touch-screen
- 3G modem and 3G internet connection
- GPS turn-by-turn navigation
- Reverse Camera (back-up camera)
- ‘AIVI CarHome’ – A custom application specifically designed for in-vehicle use made particularly for the larger AIVI display area (7-inch screen).

**Partly completed**
- Voice-recognition feature
  - Microphone hardware is functional and has tested to be useful. However, the feature was not completed due to certain software complications regarding Google voice services and the relevant licensing requirements.
6.5. Future Work

This research has potential for further study through the investigation of some features that were not included in the implementation of the current system.

6.5.1. Extended AIVI Architecture Diagram

The extended AIVI architecture Diagram (Figure 45) shows the current system and some of the planned components for the AIVI system (Wi-Fi Hotspot, OBD-II connection, and Bluetooth synchronisation module). These components can be made available in future implementations of the system.

Figure 45 - Extended AIVI Architecture Diagram
Integrating more components will allow for seamless communication between the vehicle components to the Android system, which also enables the Android system to be used as a central console to control these components. Three of these components are discussed in the following subsections.

6.5.2. Wi-Fi Receiver/Broadcaster

Being a connected system, the system could provide users with different methods of connectivity for occupants of the vehicle. There are two important aspects of the Wi-Fi component in the AIVI system: a Wi-Fi receiver, and a Wi-Fi broadcaster (Wi-Fi hotspot).

Wi-Fi Hotspot

Android phones, iPhones and other mobile devices with built-in Wi-Fi are capable of wireless connectivity (Wi-Fi). Using the AIVI system as a Wi-Fi hotspot allows users to connect to the AIVI system using multiple devices (including smartphones, laptops and other connected devices with Wi-Fi component). This will provide the Internet connection for all the wireless devices connected in the AIVI system’s network.

Wi-Fi Receiver

Any Wi-Fi capable device can access the Internet by connectivity through the Wi-Fi hotspot. Android devices, as of version 2.2, can be used as a Wi-Fi hotspot to provide Internet connectivity for other Wi-Fi capable devices (with no additional hardware).

6.5.3. Vehicle computer Integration

Integrating the AIVI system with the vehicle’s computer allows the AIVI system to retrieve information about the vehicle itself. Some of this information includes digital interpretations of various sensors such as oil pressure, fuel consumption rate, engine temperature, speed and RPM (engine revolutions per minute). Other information from the engine can also help assess the vehicle’s condition, achieved by retrieving engine error codes. These raise alerts when the vehicle has: low oil, or a broken headlight. There are various other error reporting capabilities that are supported by the vehicle’s computer which can also be integrated with the AIVI system.

The OBD-II Connector (shown in Figure 45) can be used to communicate with the vehicle’s computer. This will allow the AIVI system to retrieve data from the various sensors. Further investigation is required to give us an understanding of how to manipulate some of the vehicle’s functionalities (i.e. retrieving data from various components). Some of these components include: controlling the air-conditioning system, windows, and the centralised alarm system. Once the data is retrieved and written back to the vehicle’s computer by the AIVI system, this will allow the system to act as a central console for all vehicle manipulations (e.g. raising and closing windows via the AIVI console). Furthermore, the AIVI system can provide extensibility to allow other personal devices (via Bluetooth, Wi-Fi, and even 3G) to control the vehicle by using a secure interface to the AIVI system. This must be done with appropriate security considerations such as an authentication system to verify the actual owner of the vehicle.

When considering this type of integration it is important to ensure that the security aspect is handled appropriately, and mandate that any applications that require interfacing with the vehicle’s computer require special moderation and a specialised approval system. This will ensure that there are
no security holes that hackers and external entities could exploit. This moderation is also important to avoid malicious or badly written code to lead to any undesired or hazardous behaviour that could affect the functionalities of the vehicle.

6.5.4. Device Synchronisation

Using some of the implemented solutions (Wi-Fi and Bluetooth) we can provide applications that offer device synchronisation with the AIVI system. This allows users to use media from their own personal devices as well as synchronise contact data (from the device’s address book).

These are only some of the proposed features for future implementations of the system. Other potential features are mentioned throughout this thesis. These features have been identified from the feedback gathered in the survey conducted for this study, as well as by experience gained while testing the system.

6.6. Summary

The proposed system, AIVI, can be used as a stepping stone for the implementation of a standardised open system to be used in vehicles. This research is important as it provides users with a system that is based on an open standard that is well established (i.e. an Android system) which users are lately becoming more familiar with. While we recognise that we have developed a prototype system that the users like, we also realise that there is room for improvements.

Section 6.5 only describes some of the improvements that can be considered. We believe this system has the potential to be used by all vehicles. As users increasingly demand integrated IVIs, manufacturers would be well advised to provide standard interfaces if they wish to take advantage of third party IVI solutions. These interfaces allow communication with some of the vehicle functionalities (such as unlocking, starting the vehicle, and controlling the windows). We must also consider some of the security structures that must be put into place to avoid some of the complications that could arise from hackers who might manipulate the system without the knowledge of the vehicle owner.

We note that several Android-based systems have started to emerge since the development of this system. These have not been widely commercialised and have mostly emerged from enthusiasts and private projects.

To conclude, there are many aspects that are demonstrated in this system, and there are more to consider for future implementations of the system. With the growth of Android and its increasing adoption rate, an Android-based system could be a solution for a standardised system in vehicles of the future.
Appendix A.

IVI System Features

Multimedia

Multimedia functionalities are appropriately categories to include music, video and radio capabilities.

Music Playback

MP3 Playback
The ability to play MPEG-1 (or 2) Audio Layer 3 based media most commonly known as MP3. This media is the most commonly used media for distribution of music. MP3 is widely used for music playback and is a preferred method of sharing music.

M4A Playback
This is an audio format known as the MPEG-4 file format. This comes under two common extensions which are .mp4 and also .m4a. The .m4a format is a container which allows more than a single audio or video to be contained in a MP4 file, this also includes subtitles or lyrics. This format has been popularised by Apple Inc with the iTunes store as it has the most market in digital music retail accounting for 69% share in the digital download market in 2009.

Compact Audio Disc
Most commonly known as CD it is the standard physical storage medium for audio. Compact Discs can hold up to 80 minutes of uncompressed audio playback. This accounts for around 700MB of data. This media is used to distribute music via physical music retail stores. It is still regarded as the most common way of distributing music but it is being overshadowed by an alarming rate of digital music retail market. This type of media requires a physical disc reader, known as a CD player, to be able to read the media from the disc.
Other media
This category includes the ability to be able to play other popular media formats, compressed or uncompressed. These include WAV, ACC, and WMA etc. Many popular music players today have the ability to play multiple music formats aside from the common CD or MP3.

Video Playback

AVI
Audio Video Interleave format, most commonly known as AVI, is an audio and video format that is most commonly used for digital video distribution. This format uses a ‘codec’ to encode or decode raw data into compressed data and vice versa. This means that the ‘codec’ must be available on the device for the video format to be understood and played. This is a common media used to view movies, TV Shows and other videos which are lengthy and normally, large in size and of high quality.

MPEG-4
MPEG-4 is a file standard which is used for audio and video digital data. It is a video standard which is widely used for digital media. This format is normally of less quality than AVI format but is better suited for portable devices such as an iPod or a portable multimedia player as it’s smaller in size. This format is the common format used to distribute video media, such as TV Shows and movies via the iTunes store.

DVD Playback
Digital Video Disc most commonly known as DVD is the most common media for distributing video material through physical retail stores including video rental stores. DVD has been around since 1995 but is now being over-taken by newer more capable media such as Bluray discs. DVD playback requires a physical DVD player to be able to read the DVD discs in the device.

Radio

FM Radio
FM radio is a broadcasting technology which uses frequency modulation (FM) to produce sound over a broadcast radio. This method allows the users to listen to music, news and any broadcast provided by local radio transmitters provided by radio stations. This requires a radio receiver unit to be able to receive FM radio via an antenna.

Satellite Radio
Satellite radio broadcasts audio from radio stations to satellite radio receivers which provide crystal clear audio with a vast amount of channels. FM radio can only cover a distance of 48 to 64km in distance. Satellite radio can cover a range of up to 35,000km. Satellite radio is usually subscription based and requires a monthly instalment and is currently limited as there are only so many broadcasters providing this technology.

Internet Radio/Podcasts
Internet radio is a method of streaming live audio provided by a radio station. This is a very popular approach and usually a vast majority of popular FM broadcasters tend to broadcast via online streaming as well. This method requires internet access to be accessible and sometimes not very clear if internet connection is not reliable.

Podcasts
These are broadcasts which are distributed online. These are usually a series of digital media files released episodically. Podcast material range from academic journals to music session broadcasts by DJ's. The lengths can vary from a few minutes to podcasts lasting hours. Podcasts are gaining popularity as they allow the users to listen to a topic of interest at any time they would like with the ability to fast-forward, rewind and pause. Podcasts can be retrieved periodically through a web syndication. These are usually retrieved online.
but once downloaded can be played offline without further data consumption.

**Navigation**

**GPS**

Global Positioning System, more commonly known as a GPS, is a device used to retrieve the current geographical co-ordinate of your current location and time via multiple GPS satellites. This information is available at all times and reliable in all weather conditions anywhere in the world. The information can be retrieved by any GPS receiver. This information can be used in conjunction with a map to provide a navigation system for users to be able to navigate through road systems.

**A-GPS**

Assisted GPS (A-GPS) is an improved GPS system. An assistance server is used to supply the GPS receiver orbital data for the GPS satellites; this allows the GPS receiver to lock to the satellites more rapidly. This is used where signal is very poor in areas such as cities where signals can bounce off buildings. The contact with the assistance server requires internet connectivity but usually the GPS device will still operate if there is a lack of internet connection but will not be in the A-GPS mode. This method is commonly used in cell phones where there is GPS module.

**Map**

Maps are used in conjunction with a GPS system to provide knowledge of the surroundings. This includes a visual representation of a road system to allow ease of navigation. Maps are not dependant on GPS systems but they will not accurately visualise your current position on the map. Maps are provided online, where they are cached and saved as they are needed, or offline, where they can be accessed any time. Offline maps are usually large in size and must be loaded per location while online maps are retrieved as required but will need an internet connection.

**Satellite View**

This view, with the support of satellite imagery, allows you the user to see a high fidelity view. This allows better navigation as it helps the user visualise the route and understand the context of the route being navigated on. This feature is often used with an overlay of a map which assists in providing street and landmark names and a clear outline of the route.

**Street View**

This is a new technology which is currently provided by Google that portrays panoramic views along the actual streets. These images are composed from a vehicle carrying multiple cameras capturing every angle. This in total provides a 360° view of the current location. This is important for navigation as it allows an actual visualising of the navigation path and the surrounding landmarks. This better helps recognition of the points of interest as they are mark as you would see them in reality.

**Turn-By-Turn**

This feature provides navigation instructions as you approach your desired destination. This provides the driver with instructions regarding upcoming turns and junctions. This feature is usually integrated with speech feedback where instructions are ‘spoken’ to the driver allowing more concentration on the road. This feature is present in most professional

**Voice**

**Speech Recognition**

This feature, in context of navigation systems, allows the user to communicate with the device using certain commands. This feature is rather a safety feature as it keeps interactions at minimum with the hands safely occupied on the steering wheel.

**Speech Synthesis**

Speech synthesis, also known as Text-to-speech allows users to hear spoken instructions from the navigation device by audio playback. This feature is a safety feature as it allows the user to
concentrate on the road and less interactions with the device for a safer driving experience.

**Recognition Accuracy**
Speech recognition is not perfect but there are newer and more efficient ways of improving the accuracy of the speech recognition. Different algorithms are used to apply different methods of speech recognition. Some allow a more general and vague speech input to be used and understood well. Other methods need a more specific speech commands for successful operation. Although some systems require training for the system to understand a certain person’s voice, most navigation system today require no training as they have a limited vocabulary.

**Multi-lingual**
This feature, in conjunction with speech synthesis and speech recognition, lets the user interact with the system in different languages supported by the system.

**Display**

**Size**
Size of the display can vary. A standard size for an on-board infotainment system is 7inches. Sizes can vary between systems and vendors and different operating-systems can support multiple size for different display devices.

**Stylus Input**
This method uses the stylus pen, a tool used for touch-screen interaction to provide accurate input.

**Multi-touch Interface**
This interface allows the use of multiple inputs to provide the user with a more interactive experience. Using multi-touch allows more uses of gestures to manipulate applications such as maps to easily navigate, zoom in/out and pan the map. Using multi-touch interfaces allows a more natural input by the user and less use of menus and user-interface widgets to communicate with the application.

**Software**

**Operating System**
The operating system (OS) is a communication mechanism which regulates the user’s interaction with the device hardware. This operates on a kernel, which provides the basic level of control over the hardware. The OS usually provides the user with a graphical user-interface (GUI) to allow easier interaction with the system.

**SDK (Software Development Kit)**
The software development kit is provided by the operating system to allow developers to create software applications for the operating system. The SDK provides a set of API’s (Application Programming Interface) which enables the operating system to interact with software. Having an SDK allows a more extensible system which can be provide a much larger application-base providing the system with more features than the standard applications that come shipped with the device.

**Application Store**
An application store is digital distribution platform for mainly mobile devices to distribute applications created for specific platforms. This provides a place for developed application to be downloaded, purchased and installed on the user’s device. Common examples of an application store are; App Store (Apple), Android Market (Android), App World (RIM).

**Updateable**
Infotainment Systems these days allow users to update the devices’ firmware. These updates are usually provided by the company thus providing bug-fixes as well as new features to the device that further extend its function. These firmware updates also can come from community-based builds in cases where the software is open-source. Having a system which has the ability to update its firmware provides an always up-to-date system; with
enhanced security, better feature-base and a more bug-free system.

**Internet Browser**
More apparent these days is infotainment systems with internet support. This allows devices to be able to communicate to the internet via an internet browser. These browsers can be built by the vendors themselves or built by a third party developer. Having an internet browser is becoming more standard in infotainment systems but this requires a means of internet connectivity via 3G hardware or other means.

**Communication**

**Voice Calling and SMS (Short Message Service)**
Voice calling and SMS allows communication between two parties via usually a GSM network using voice or text messaging. These are known to be the norm of communication in mobile devices. For this feature to be operational the device needs additional GSM hardware as well as connectivity to a provided (using a SIM card usually). Voice calling used in infotainment systems, usually where it is built into a mobile environment, use the vehicles speaker system to allow hands-free calling. This keeps both hands on the vehicle’s steering while thus minimising interactions with the device.

Text messaging application have now developed where users can input messages via speech recognition functionality and messages can even be read back using speech synthesis techniques, usually provided by the operating system’s SDK. These features are important in driving environments.

**GPRS and 3G Internet Data**
These are standards of internet communication to allow mobile devices to access the Internet in high speeds. GPRS is the earlier implementation, more commonly known as 2G or 2.5G while 3G is the later implementation which is widely adopted. 3G not only requires 3G hardware, but also requires the network provider to have the necessary tools to deliver 3G data services. These standards are necessary to access any data online, these services can include applications that require retrieval of; traffic updates, news, weather, media streaming and such. 3G network is covered by both Vodafone and Telecom with around 97% coverage\(^4\).

**Peripherals**

**USB Host**
Being a USB host gives the device the ability to interface with USB devices as they are supported. This means that the user can insert a USB device in the slot provided and be able to send and retrieve data to and from that specific device. This can be as simple as a USB flash drive to a USB keyboard or even a 3G data dongle.

**External Storage**
Having the ability to provide external storage means that users can provide other mediums of storage rather than the device’s internal memory. As internal memory is usually limited, having expandable memory slots allows users to store media as well as applications which can be used by the operating system. These medias can include; USB flash disk, SD Card, micro-SD and other forms of storage medium.

**CD/DVD**
CD/DVD readers allow the reading of data from a CD or DVD medium. This can also be defined as external storage, but these devices are usually Read-Only. This means that the media can only be read from the devices; new data can be stored on them by the infotainment device. This media is still common where media is purchased through a physical retail store (music or movies) but is less common as digital media distribution becomes more common.

\(^{44}\) 3G coverage area (VodafoneNZ 2010)
**OBD-II**
Devices with the ability to interface with On-board Diagnostics (OBD-II as the current standard) are able to retrieve data from the vehicle’s Electronic Control Unit (ECU). Data can include speed, to RPM, to Air-bag deployed trigger. There is a vast range of usable data which can be further used by applications. OBD-II interfaces can be connected via WiFi or Bluetooth for data retrieval.

**Bluetooth**
Bluetooth support allows sending and retrieval of data from Bluetooth enabled devices. This allows communication, known as pairing, of Bluetooth devices to perform a task. This technique is used commonly for, but not limited to, hands-free sets and file transfer. OBD-II interface can also be connected via Bluetooth by having a Bluetooth enabled OBD-II interface module.
Appendix B.

Opinion Survey for Gathering Requirements

Sample copy of the opinion-survey used to collect opinions from participants regarding the proposed system including desirability of system features.
# In-Vehicle Infotainment System - Features

How would you rate having the following features in your car?

Please choose from:
1 – Very Undesirable; 2 – Undesirable; 3 – Neutral; 4 – Desirable, 5 – Very Desirable

<table>
<thead>
<tr>
<th>Feature</th>
<th>Rating</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audio Player</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>Music and Radio</td>
<td></td>
</tr>
<tr>
<td>Video Player</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>Turn-By-Turn Navigation (GPS)</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>This includes a voice-guided navigation with voice input.</td>
<td></td>
</tr>
<tr>
<td>Reverse Camera</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>Camera placed at the rear of the vehicle to give the driver a better view while reversing.</td>
<td></td>
</tr>
<tr>
<td>Voice Actions</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>Speech recognition with voice feedback (speech-to-speech): Text, email, weather, get directions, tweet, note-to-self, search, calculate 12 times 8, dictionary etc.</td>
<td></td>
</tr>
<tr>
<td>Email</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>Able to get email notifications in-vehicle with the ability of hearing it read back to you.</td>
<td></td>
</tr>
<tr>
<td>Internet Browser</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>Browse the Internet, Facebook, etc. with voice actions.</td>
<td></td>
</tr>
<tr>
<td>Installing external party Applications (Apps)</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>This includes getting applications regarding Traffic Information, Traffic Cameras, Music streaming, News, Weather etc. (Same applications as smartphones).</td>
<td></td>
</tr>
<tr>
<td>In-Vehicle Wi-Fi Hotspot</td>
<td>1 2 3 4 5</td>
</tr>
<tr>
<td>Ability for vehicle occupants to connect smartphones, laptop or any connected device in the vehicle to have an internet connection via Wireless (Wi-Fi).</td>
<td></td>
</tr>
</tbody>
</table>
General Questions

1. What is your gender?
   a. Male
   b. Female

2. Do you have your own car?
   a. Yes
   b. No

3. What is your occupation?
   a. Undergraduate Student
   b. Postgraduate Student
   c. IT Professional
   d. Other:

4. What is your current driver licence?
   a. Learners
   b. Restricted
   c. Full
   d. International

5. What is your age group?
   a. 14-17
   b. 18-24
   c. 25-39
   d. 40-54
   e. 55+

6. Do you any previous experience with the following (you may pick more than one option):
   - Android O.S (Android Smartphones)
   - iOS (iPhone, iPad, iPod Touch)
   - BlackBerry
   - webOS (Palm Pre)
   - None of the Above

7. Do you prefer to stream or download your media?
   a. Stream
   b. Download
   c. I have no idea what you just said.

8. What is your estimated personal data usage per month on your mobile device?

   - My phone is not capable of browsing the internet
   - My phone is capable but I do not use it.
   - Less than 10MB
   - 10-100MB
   - 100-500MB
   - More than 1GB

Data Usage Guide (BroadbandGenie.co.uk, 2010)

<table>
<thead>
<tr>
<th>Activity</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 hour of instant messaging</td>
<td>0.25-1MB</td>
</tr>
<tr>
<td>1 hour of web browsing</td>
<td>1.5-25MB</td>
</tr>
<tr>
<td>Download 100 emails</td>
<td>1-10MB</td>
</tr>
<tr>
<td>Download 1 photo</td>
<td>0.05-2MB</td>
</tr>
<tr>
<td>Download 1 MP3</td>
<td>3-8MB</td>
</tr>
<tr>
<td>Download 1 film trailer</td>
<td>7-50MB</td>
</tr>
<tr>
<td>Download 1 film</td>
<td>700-1500MB</td>
</tr>
<tr>
<td>Streaming 1 hour of video</td>
<td>250-500MB</td>
</tr>
<tr>
<td>Streaming 1 hour of audio</td>
<td>50-150MB</td>
</tr>
<tr>
<td>Navigating 10KM (Google Maps)</td>
<td>0.5-1MB</td>
</tr>
</tbody>
</table>

Please Note: 1GB is roughly 1000MB
Open ended comments

Are there any features you deem necessary for In-Vehicle Infotainment?

Any further comments you would like to add?

Thank you for participating in this study.

Your contribution to this research is invaluable for helping developers understand what the end-users needs and requirements are to building an effective system.

We appreciate your honesty and willingness to assist with this important research.
Appendix C.

Comparison Matrix

Complete comparison matrix that was conducted to compare features of current In-Vehicle Infotainment systems available.
<table>
<thead>
<tr>
<th>Multimedia</th>
<th>AIVI Android-based IVI</th>
<th>MeeGo (IVI) GENIVI Alliance</th>
<th>Ford Sync By Ford and Microsoft</th>
<th>Kia UVO By Kia and Microsoft</th>
<th>Blue&amp;Me By Fiat and Microsoft</th>
<th>Pioneer AVIC-Z110BT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Music Player</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Video Player</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Radio Receiver</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internet Radio</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Podcasts</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Navigation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hardware</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GPS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>A-GPS</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Software</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Map</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Satellite View</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Street View</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Turn-By-Turn</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Voice</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Speech Recognition</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Speech Synthesis</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Multi-lingual</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Display</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size</td>
<td>7 inches</td>
<td>No Current Hardware***</td>
<td>Varies**</td>
<td>Varies**</td>
<td>Varies**</td>
<td>7 inches</td>
</tr>
<tr>
<td>Stylus Input</td>
<td>✓</td>
<td>❌</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Multi-Touch</td>
<td>✓</td>
<td>❌</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Software</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating System</td>
<td>Android O.S 2.1</td>
<td></td>
<td>Microsoft Auto 4.1</td>
<td>Microsoft Auto 4.1</td>
<td>Microsoft Auto 4.1</td>
<td>Proprietary Pioneer Firmware</td>
</tr>
<tr>
<td>SDK</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Application Store</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Updatable</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Internet Browser</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Games</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Communication</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Voice Calling</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>SMS</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>GPRS</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>3G</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Peripherals</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>USB Host</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>External Storage</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>CD/DVD Disk Player</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>OBD-II</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>Reverse Camera</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
<tr>
<td>WiFi Hotspot</td>
<td>✓</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
<td>❌</td>
</tr>
</tbody>
</table>

**Legend**

- ✓: Available
- ✗: Unavailable
- ①: Capable - Future Release
- (): Via Synchronised Device - Not Directly
Appendix D.

Embedded Development

Low level embedded development is necessary to load any application to a development board. This involves writing to NAND memory several bootstrap components. The bootstrap is broken up into several components (bootstrap chain) that in-turn leads to the boot of the Kernel then the Android operating-system.

NAND Flash Memory

NAND flash is a special type of memory that holds non-volatile data. This means that the data is stored even when the power source is cut-off. NAND flash is currently the common type of Flash memory available in the market as most USB flash drives and memory cards use this technology to store data (Shirota, 2000).

The TI OMAP3530 board, which is used by the AIVI system, uses NAND flash memory to store data onto the board itself which is kept once power source is turned off. The NAND capacity for the AIVI system is 256MB.

The bootstrap, that allows the booting of the system, is written to the NAND flash. The bootstrap contains X-Loader and U-Boot. The X-Loader will lead the U-Boot, which is the actual boot loader that will determine which and how to load the operating system (kernel) as well as the system configuration required.

1. x-loader ->
2. u-boot ->
3. kernel ->
4. file system ->
5. user area

Figure 46 - Bootstrap Order

X-Loader

X-Loader is a small first stage boot loader derived from the U-Boot base code to be loaded into the internal static RAM by the OMAP ROM code. Because the internal static RAM is very small (64k), X-Loader is stripped down to the essentials and is used to initialize
memory and enough of the peripheral devices to access and load the second stage loader (U-Boot) into main memory.

**U-Boot**

U-Boot, a boot loader for Embedded boards based on PowerPC, ARM, MIPS and several other processors, which is used to initialize and test the hardware or to download and run application code (Denk, 2004).

U-Boot is the second phase of the boot loader following X-Loader. This is run from the ROM and used to load and execute the operating system (the Linux kernel in our case) into the RAM. U-Boot includes some methods for access SD/MMC devices, Ethernet as well as setting up system configuration before executing the Linux kernel.

The U-Boot used in the AIVI system will initialise the hardware, and setup the environment setting and is used to boot the Linux Kernel.

**Linux Kernel**

The Linux operating system, known as the Linux kernel, is required for Android to run. The kernel represents the most basic element of an operating system. Android's Linux kernel is responsible for managing the system's resources and acts as an abstraction layer between hardware and a system's applications (Bovet, et al., 2002).

The kernel used is customised to be used with Android as well as AIVI specific components (Reverse camera, GPS, 3G etc.) as it contains the necessary drivers for the hardware being used. The Linux kernel used is version 2.6.29 as of Android 2.1 (Éclair). This is usually a small file sized around 2MB.

The Linux kernel image is generated once the source-code for the kernel has been compiled. This usually generates a file named uImage.bin.

**File-System (Android O.S.)**

The file-system will contain our UBIFS system image which is the actual Android OS that is built from the source. This file is around 40MB and will be based on the system image produced once the Android source-code is compiled.

**Writing to NAND**

To be able to write into flash memory; SD/MMC cards is used. We assign certain address slots for the x-loader, u-boot as well as the Linux Kernel and system image (Our Android build).

The system image can be located at out/target/product/devkit8000/system.img once compiled. The kernel can be obtained once compiling it or retrieving a prebuilt kernel (usually this is uImage.bin).

This is done subsequently using the following documented code:

```
OMAP3 # mmcinit

OMAP3 # fatload mmc 0:1 80000000 x-load.bin.ift

reading x-load.bin.ift
9664 bytes read

OMAP3 # nand unlock

device 0 whole chip
nand_unlock: start: 00000000, length: 134217728!
NAND flash successfully unlocked

OMAP3 # nand hw

OMAP3 # nand erase 0 80000

NAND erase: device 0 offset 0x0, size 0x80000
Erasing at 0x60000 -- 100% complete.
OK

OMAP3 # nand write.i 80000000 0 $(filesize)

NAND write: device 0 offset 0x0, size 0x80000
Writing data at 0x7f800 -- 100% complete.
524288 bytes written: OK
```
The commands used in the previous sequence assume that we are booting via SD/MMC Memory Card. The commands start by initialising the MMC/SD card (mmcinit), then the file is loaded into memory by the _fatload_ command. The NAND is then unlocked using the _nand unlock_ command. Then the existing data on the NAND for that memory allocation is erased via the _nand erase_. Then the new file is written into the memory allocated for it (_nand write_).

This process is then repeated for each application/file that needs to be written on the NAND; X-Loader, U-Boot, Linux kernel and the Android O.S.

**U-Boot Environment Variables**

Once flashing the NAND, we must set our u-boot environment. The U-Boot environment is a block of memory that is kept on persistent storage. This is then copied to the RAM once U-Boot is loaded. These environment settings are used to configure the system including display type, resolution and indicating the mount point for the file-system.

**Booting Android**

Once the U-Boot environment settings are saved on the NAND, then we can reset the device and it should automatically boot our kernel as well as our system image.

Our initial boot will load longer than expected as it sets up all the environment settings and one-time executions including creation of necessary system directories and system permissions. This is all done automatically by the Android _init.rc_ script; a script that initialises and runs any applications and code necessary on boot.
Retrieving Android-source

The host, development, machine must have necessary prerequisite applications before it can successfully retrieve and build the Android source. The applications that are required are retrieved using the following code:

```bash
$ sudo apt-get install build-essential
$ sudo apt-get install make
$ sudo apt-get install gcc
$ sudo apt-get install g++
$ sudo apt-get install libc6-dev
$ sudo apt-get install patch
$ sudo apt-get install texinfo
$ sudo apt-get install libncurses-dev
$ sudo apt-get install git
$ sudo apt-get install gawk
$ sudo apt-get install flex bison gperf libstdc++-dev libbsd0-dev libwxgtk2.6-dev build-essential zip curl
$ sudo apt-get install ncurses-dev
$ sudo apt-get install zlib1g-dev
$ sudo apt-get install valgrind
$ sudo apt-get install sun-java5-jdk sun-java5-jre
```

To retrieve the source code in order to modify and build according to our modifications we need to retrieve the source code via git and repo (version-control and source-code repositories) on our Linux development machine (Ubuntu 10.04 was used as the host).

- **Git** – This is an open-source version-control repository system that handles large projects that span across multiple repositories. The Android project is hosted in a git repository and operations such as branching, applying patches, and committing changes are handled by git.

- **Repo** – This tool is specifically designed for Android. It is built on top of git and is used to automate several Android development workflows. This eases the development process for the Android source-code.

Next, the git client must be installed, and then the repo script must be retrieved

```bash
$ sudo apt-get install git-core
$ curl -o ~/bin/repo http://android.git.kernel.org/repo
$ chmod a+x ~/bin/repo
```

Next we need to retrieve the build we would like to use for our development board

```bash
$ mkdir aivibuild
$ cd aivibuild
$ repo init -u git://gitorious.org/[build]/manifest.git
$ repo sync
```

The **repo init** command allows us to set the repo in our current directory. The -b option retrieves a specific branch in the repo, in this case the beagle-eclair branch is retrieved. This branch is the latest 2.1 version. **repo sync** downloads new changes and updates.

To build the source:

```bash
$ echo "TARGET_PRODUCT := aivi" > buildspec.mk
$ echo "INSTALL_PREBUILT_DEMO_APKS := true" >> buildspec.mk
$ make
```

Once successful, the system image is generated here:

```
out/target/product/devkit8000/root/
```

The image is copied into an SD-card which can then be written into the NAND flash of the device from u-boot.

---

45 *curl* command retrieves URL while -o means to the specified output
46 *chmod* is a command that allows us to change file access permissions. a+x refers to all users (a) to execute (x). The + operator is used to add permission to existing permissions.
Appendix E.

Opinion Survey Results based on Gathered Requirements

Raw data collected from the results of the opinion-survey.
<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Voice</th>
<th>Video</th>
<th>Audio</th>
<th>GPS</th>
<th>Reverse Camera</th>
<th>Email</th>
<th>Internet</th>
<th>Apps</th>
</tr>
</thead>
<tbody>
<tr>
<td>12/28/2010 18:23:01</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/28/2010 17:03:19</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/28/2010 6:02:16</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/28/2010 4:51:52</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/28/2010 3:31:25</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>12/28/2010 1:09:47</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>12/28/2010 1:01:42</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>12/28/2010 0:50:24</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12/28/2010 0:23:13</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>12/28/2010 0:16:03</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>12/26/2010 19:11:32</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>12/26/2010 5:35:05</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>12/26/2010 5:16:29</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>12/26/2010 5:09:14</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>12/26/2010 4:52:27</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>12/26/2010 4:45:14</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>12/26/2010 4:42:09</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>12/26/2010 4:40:51</td>
<td>1</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>12/26/2010 4:37:54</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/26/2010 4:37:18</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>12/26/2010 4:34:58</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>12/24/2010 16:40:02</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>12/24/2010 14:05:35</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>12/24/2010 1:13:56</td>
<td>4</td>
<td>1</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>12/24/2010 1:12:23</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>12/23/2010 12:33:41</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>12/23/2010 12:06:01</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 8:06:39</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 5:19:00</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 4:59:41</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>12/23/2010 4:56:09</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 4:26:09</td>
<td>2</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 4:01:54</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>12/23/2010 3:43:41</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 3:10:59</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 2:57:04</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>12/23/2010 2:43:18</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 2:38:42</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>12/23/2010 2:25:30</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 2:03:52</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 1:54:51</td>
<td>2</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 1:50:56</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/23/2010 1:46:27</td>
<td>5</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>12/21/2010 3:48:12</td>
<td>3</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>12/21/2010 3:43:04</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>12/21/2010 3:39:32</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
## General Participant Information

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Gender</th>
<th>Own Car</th>
<th>Occupation</th>
<th>Licence</th>
<th>Age Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>12/28/2010 18:23:01</td>
<td>Male</td>
<td>Yes</td>
<td>Postgraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/28/2010 17:03:19</td>
<td>Male</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>International</td>
<td>18-24</td>
</tr>
<tr>
<td>12/28/2010 6:02:16</td>
<td>Female</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Restricted</td>
<td>25-39</td>
</tr>
<tr>
<td>12/28/2010 4:51:52</td>
<td>Female</td>
<td>Yes</td>
<td>Cabin crew</td>
<td>International</td>
<td>18-24</td>
</tr>
<tr>
<td>12/28/2010 1:01:42</td>
<td>Male</td>
<td>No</td>
<td>Kitchen Chef</td>
<td>Full</td>
<td>25-39</td>
</tr>
<tr>
<td>12/28/2010 0:50:24</td>
<td>Male</td>
<td>Yes</td>
<td>Postgraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/28/2010 0:16:03</td>
<td>Female</td>
<td>No</td>
<td>Undergraduate</td>
<td>Learners</td>
<td>18-24</td>
</tr>
<tr>
<td>12/26/2010 19:11:32</td>
<td>Male</td>
<td>Yes</td>
<td>N/A</td>
<td>Full</td>
<td>25-39</td>
</tr>
<tr>
<td>12/26/2010 5:35:05</td>
<td>Male</td>
<td>Yes</td>
<td>Engineer</td>
<td>Full</td>
<td>25-39</td>
</tr>
<tr>
<td>12/26/2010 5:09:14</td>
<td>Female</td>
<td>Yes</td>
<td>Accountant</td>
<td>Restricted</td>
<td>25-39</td>
</tr>
<tr>
<td>12/26/2010 4:52:27</td>
<td>Male</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/26/2010 4:45:14</td>
<td>Male</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/26/2010 4:37:54</td>
<td>Female</td>
<td>Yes</td>
<td>Intern Pharmacist</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/26/2010 4:37:18</td>
<td>Male</td>
<td>Yes</td>
<td>Safety Engineer</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/26/2010 4:34:58</td>
<td>Male</td>
<td>Yes</td>
<td>Postgraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/24/2010 16:40:02</td>
<td>Male</td>
<td>No</td>
<td>Undergraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/24/2010 14:05:35</td>
<td>Female</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/24/2010 1:13:56</td>
<td>Female</td>
<td>Yes</td>
<td>Postgraduate</td>
<td>Full</td>
<td>40-54</td>
</tr>
<tr>
<td>12/24/2010 1:12:23</td>
<td>Female</td>
<td>No</td>
<td>Postgraduate</td>
<td>None</td>
<td>25-39</td>
</tr>
<tr>
<td>12/23/2010 12:06:01</td>
<td>Female</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Restricted</td>
<td>25-39</td>
</tr>
<tr>
<td>12/23/2010 8:06:39</td>
<td>Male</td>
<td>No</td>
<td>High School</td>
<td>None</td>
<td>14-17</td>
</tr>
<tr>
<td>12/23/2010 5:19:00</td>
<td>Male</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Full</td>
<td>25-39</td>
</tr>
<tr>
<td>12/23/2010 4:56:09</td>
<td>Female</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>International</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 4:01:54</td>
<td>Male</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Restricted</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 2:57:04</td>
<td>Female</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 2:43:18</td>
<td>Male</td>
<td>No</td>
<td>Undergraduate</td>
<td>None</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 2:38:42</td>
<td>Male</td>
<td>Yes</td>
<td>IT Professional</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 2:25:30</td>
<td>Male</td>
<td>Yes</td>
<td>IT Professional</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 2:03:52</td>
<td>Female</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Restricted</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 1:54:51</td>
<td>Female</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 1:50:56</td>
<td>Male</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/23/2010 1:46:27</td>
<td>Female</td>
<td>Yes</td>
<td>Freelancer</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/21/2010 3:48:12</td>
<td>Male</td>
<td>Yes</td>
<td>Undergraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
<tr>
<td>12/21/2010 3:43:04</td>
<td>Male</td>
<td>Yes</td>
<td>Postgraduate</td>
<td>Full</td>
<td>18-24</td>
</tr>
</tbody>
</table>
## Technical Participant Information

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Do you any previous experience with the following</th>
<th>Download vs Stream</th>
<th>Data Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>12/28/2010 18:23:01</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Stream</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/28/2010 17:03:19</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>1GB</td>
</tr>
<tr>
<td>12/28/2010 6:02:16</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Stream</td>
<td>100-500MB</td>
</tr>
<tr>
<td>12/28/2010 4:51:52</td>
<td>BlackBerry</td>
<td>Download</td>
<td>10-50MB</td>
</tr>
<tr>
<td>12/28/2010 3:31:25</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/28/2010 1:09:47</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad), BlackBerry, webOS (PalmPre)</td>
<td>Download</td>
<td>1GB</td>
</tr>
<tr>
<td>12/28/2010 1:01:42</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>Less than 10MB</td>
</tr>
<tr>
<td>12/28/2010 0:50:24</td>
<td>None of the Above</td>
<td>Stream</td>
<td>0MB</td>
</tr>
<tr>
<td>12/28/2010 0:23:13</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/28/2010 0:16:03</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>Not Capable</td>
</tr>
<tr>
<td>12/26/2010 19:11:32</td>
<td>None of the Above</td>
<td>Download</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/26/2010 5:35:05</td>
<td>Android O.S - (Android Smartphones)</td>
<td>Stream</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/26/2010 5:16:29</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/26/2010 5:09:14</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Stream</td>
<td>100-500MB</td>
</tr>
<tr>
<td>12/26/2010 4:52:27</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>Not Capable</td>
</tr>
<tr>
<td>12/26/2010 4:45:14</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad), BlackBerry, webOS (PalmPre)</td>
<td>Download</td>
<td>1GB</td>
</tr>
<tr>
<td>12/26/2010 4:42:09</td>
<td>None of the Above</td>
<td>Stream</td>
<td>0MB</td>
</tr>
<tr>
<td>12/26/2010 4:40:51</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/26/2010 4:37:54</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/26/2010 4:37:18</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/26/2010 4:34:58</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/24/2010 16:40:02</td>
<td>Android O.S - (Android Smartphones)</td>
<td>Download</td>
<td>10-50MB</td>
</tr>
<tr>
<td>12/24/2010 14:05:35</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>10-50MB</td>
</tr>
<tr>
<td>12/24/2010 1:13:56</td>
<td>None of the Above</td>
<td>Stream</td>
<td>0MB</td>
</tr>
<tr>
<td>12/24/2010 1:12:23</td>
<td>BlackBerry</td>
<td>Stream</td>
<td>0MB</td>
</tr>
<tr>
<td>12/23/2010 12:33:41</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>0MB</td>
</tr>
<tr>
<td>12/23/2010 12:06:01</td>
<td>Android O.S - (Android Smartphones)</td>
<td>Download</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/23/2010 9:25:13</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>Not Capable</td>
</tr>
<tr>
<td>12/23/2010 8:06:39</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>100-500MB</td>
</tr>
<tr>
<td>12/23/2010 5:19:00</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Stream</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/23/2010 4:59:41</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/23/2010 4:56:09</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>10-50MB</td>
</tr>
<tr>
<td>12/23/2010 4:26:09</td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Stream</td>
<td>100-500MB</td>
</tr>
<tr>
<td>Date/Time</td>
<td>Platform Description</td>
<td>Action</td>
<td>Size</td>
</tr>
<tr>
<td>-----------------</td>
<td>--------------------------------------------------------------------------------------</td>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>12/23/2010 4:01:54</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>0MB</td>
</tr>
<tr>
<td>12/23/2010 3:43:41</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>10-50MB</td>
</tr>
<tr>
<td></td>
<td>Android O.S - (Android Smartphones), iOS (iPhone, iPod Touch, iPad), BlackBerry, webOS</td>
<td>Download</td>
<td>100-500MB</td>
</tr>
<tr>
<td>12/23/2010 3:10:59</td>
<td>(PalmPre)</td>
<td>Download</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/23/2010 3:10:59</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>0MB</td>
</tr>
<tr>
<td>12/23/2010 2:57:04</td>
<td>iOS (iPhone, iPod Touch, iPad)</td>
<td>Download</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/23/2010 2:38:42</td>
<td>(iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>More than 1GB</td>
</tr>
<tr>
<td>12/23/2010 2:25:30</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Stream</td>
<td>10MB</td>
</tr>
<tr>
<td>12/23/2010 2:03:52</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Stream</td>
<td>Less than 10MB</td>
</tr>
<tr>
<td>12/23/2010 1:54:51</td>
<td>None of the Above</td>
<td>Download</td>
<td>0MB</td>
</tr>
<tr>
<td>12/23/2010 1:50:56</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Download</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/23/2010 1:46:27</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry, webOS (PalmPre)</td>
<td>Download</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/21/2010 3:48:12</td>
<td>iOS (iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Stream</td>
<td>50-100MB</td>
</tr>
<tr>
<td>12/21/2010 3:39:32</td>
<td>(iPhone, iPod Touch, iPad), BlackBerry</td>
<td>Stream</td>
<td>50-100MB</td>
</tr>
</tbody>
</table>
Open-ended Comments

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Are there any other features you deem necessary for In-Vehicle Infotainment?</th>
</tr>
</thead>
<tbody>
<tr>
<td>12/28/2010 6:02:16</td>
<td>Make sure that you have IVI enabled for AHS (automated driverless car systems) to connect in countries with the following projects: <a href="http://en.wikipedia.org/wiki/Driverless_car">http://en.wikipedia.org/wiki/Driverless_car</a> for more info :) And good luck!</td>
</tr>
<tr>
<td>12/26/2010 19:11:32</td>
<td>Bluetooth Connectivity, Over The Air (OTA) Updates, Remote SIM, USB Port(s) for keyboard and/or mouse, Multi-Tasking, ie still playing music while Nav is on.</td>
</tr>
<tr>
<td>12/26/2010 5:16:29</td>
<td>GPS, Bluetooth, Charger, Speaker phones,</td>
</tr>
<tr>
<td>12/26/2010 5:09:14</td>
<td>Nothing that hasn't already been mentioned.</td>
</tr>
<tr>
<td>12/26/2010 4:34:58</td>
<td>Interaction between cars, location of other cars (friends and family etc) on the road.</td>
</tr>
<tr>
<td>12/23/2010 12:33:41</td>
<td>Audio and video player supports all file formats, A built in alarm/Garage door controller would be nice</td>
</tr>
<tr>
<td>12/23/2010 12:06:01</td>
<td>Being able to plug in your devices using USB cable</td>
</tr>
<tr>
<td>12/23/2010 5:19:00</td>
<td>Internet radio</td>
</tr>
<tr>
<td>12/23/2010 4:59:41</td>
<td>You didn't say anything about making phone calls, dunno if it counts as infotainment.. Everything else is covered by 3rd party apps</td>
</tr>
<tr>
<td>12/23/2010 4:01:54</td>
<td>In addition to the voice activation idea, using voice to initiate air conditioning, window wipers etc.</td>
</tr>
</tbody>
</table>
Appendix F.

GPS Development

AIVI GPS Implementation

The steps taken to create a GPS library, as documented by the Android open-source project (AOSP) web-page\(^47\).

1. Need to build a shared library that implements the gps.h interface header
   (include/hardware/gps.h)
2. Library must be named in the format ‘libgps.so’, this will be loaded from the
   /system/lib at runtime.
3. Sources, as well as the Android make-file, are placed in the user
   spacevendor/acme/chipset_or_board/gps/ (where "acme" is your
   organization name and "chipset_or_board" is your hardware target).
4. The Android.mk is used to define the library/module name as well as the source
   files to be used to build the module.

Android’s official sources do not provide a hardware based GPS implementation. AIVI’s GPS implementation is based on the GPS QEMU\(^48\) official source. This source-code provides implementation for software emulator-based GPS (rather than hardware).

---


\(^48\) QEMU is a processor emulator which is what the Android Emulator is based on. It provides a virtual ARM mobile device on which you can run your Android applications.
AIVI’s GPS library consists of:

```c
gps_get_hardware_interface()
```

This is what the framework uses to communicate with hardware native code:

- `gps_state_init` - GPS Initialisation – Open GPS port for transmission
- `gps_state_start` - Sends a GPS Start command
- `gps_state_stop` - Sends a stop command to the GPS port
- `gps_state_update_fix_freq` - This sets the GPS Fix Frequency
- `gps_state_done` - Cleanup code once GPS state is set to QUIT

The specific GPS TTY port is set in a system properties file which the library will read and identify, so the GPS is not hardcoded in the system. The property to set the GPS port is `ro.kernel.android.gps`. This property is identified in the code and saved in the `system.prop` file located in the user space directory.

The port is then opened according to several configurations set, as documented in the source code in Appendix B, and a stream of data is retrieved in an NMEA format as described above. This data is then effectively parsed according to the definitions of the NMEA protocols (Can be seen in the `nmea_reader_parse` method). This can be seen in the NMEA Tokeniser and NMEA Parser blocks of code in the GPS library source file.

Once the NMEA output is parsed, the parameters for longitude, latitude, time and other parameters are set and can be retrieved by the Application Framework.

This process also involves threading, a new thread is created for each NMEA sentence received and is processed in parallel. This allows the framework to keep up with the one second update cycle and provide the GPS readings as they are received. This is important as it allows a smoother navigation when displayed on the map.

The sources for the library implemented are defined in the user space. This is an Android development feature that allows vendors to build libraries to the specific chipsets or boards being used for certain devices and easily managed when building the source for multiple devices with different hardware.
GPS NMEA References

NMEA Reference guide for parsing GPS data retrieved from the GPS receiver.

$GPGGA - Global Positioning System Fix Data

Output:
$GPGGA,170834,4124.8963,N,08151.6838,W,1,05,1.5,280.2,M,34.0,M,,,*75

<table>
<thead>
<tr>
<th>Name</th>
<th>Example Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentence Identifier</td>
<td>$GPGGA</td>
<td>Global Positioning System Fix Data</td>
</tr>
<tr>
<td>Time</td>
<td>170834</td>
<td>17:08:34 UTC</td>
</tr>
<tr>
<td>Latitude</td>
<td>4124.8963, N</td>
<td>41d 24.8963’ N or 41d 24’ 54” N</td>
</tr>
<tr>
<td>Longitude</td>
<td>08151.6838, W</td>
<td>08d 51’ 41” W or 81d 51’ 41” W</td>
</tr>
<tr>
<td>Fix Quality:</td>
<td>1</td>
<td>Data is from a GPS fix</td>
</tr>
<tr>
<td>- 0 = Invalid, - 1 = GPS fix, - 2 = DGPS fix</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of Satellites</td>
<td>05</td>
<td>5 Satellites are in view</td>
</tr>
<tr>
<td>Horizontal Dilution of Precision (HDOP)</td>
<td>1.5</td>
<td>Relative accuracy of horizontal position</td>
</tr>
<tr>
<td>Altitude</td>
<td>280.2, M</td>
<td>280.2 meters above mean sea level</td>
</tr>
<tr>
<td>Height of geoid above WGS84 ellipsoid</td>
<td>-34.0, M</td>
<td>-34.0 meters</td>
</tr>
<tr>
<td>Time since last DGPS update</td>
<td>blank</td>
<td>No last update</td>
</tr>
<tr>
<td>DGPS reference station id</td>
<td>blank</td>
<td>No station id</td>
</tr>
<tr>
<td>Checksum</td>
<td>*75</td>
<td>Used by program to check for transmission errors</td>
</tr>
</tbody>
</table>

Break down of Parameters for GPGGA

$GPGGA, hhmmss.ss, ddmm.mmm, a, dddmm.mmm, b, q, xx, p.p, a.b, M, c.d, M, x.x, nnnn

- hhmmss.ss = UTC of position
- ddmm.mmm = latitude of position
- a = N or S, latitude hemisphere
- dddmm.mmm = longitude of position
- b = E or W, longitude hemisphere
- q = GPS Quality indicator
  - (0=No fix, 1=Non-differential GPS fix, 2=Differential GPS fix, 6=Estimated fix)
- xx = number of satellites in use
- p.p = horizontal dilution of precision
- a.b = Antenna altitude above mean-sea-level
- M = units of antenna altitude, meters
- c.d = Geoidal height
- M = units of geoidal height, meters
- x.x = Age of Differential GPS data (seconds since last valid RTCM transmission)
- nnnn = Differential reference station ID, 0000 to 1023
$GPRMC - Recommended minimum specific GPS/TRANSIT data

Output:
$GPRMC,225446,A,4916.45,N,12311.12,W,000.5,054.7,191194,020.3,E*68

<table>
<thead>
<tr>
<th>Name</th>
<th>Example Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentence Identifier</td>
<td>$GPGGA</td>
<td>Global Positioning System Fix Data</td>
</tr>
<tr>
<td>Time</td>
<td>225446</td>
<td>17:08:34 UTC</td>
</tr>
<tr>
<td>Navigation receiver warning</td>
<td>A</td>
<td>Valid position</td>
</tr>
<tr>
<td>Latitude</td>
<td>4916.45,N</td>
<td>Latitude 49 deg. 16.45 min. North</td>
</tr>
<tr>
<td>Longitude</td>
<td>12311.12,W</td>
<td>Longitude 123 deg. 11.12 min. West</td>
</tr>
<tr>
<td>Speed over ground (Knots)</td>
<td>000.5</td>
<td>0.5 Knots</td>
</tr>
<tr>
<td>Track made good in degrees True</td>
<td>054.7</td>
<td>True course</td>
</tr>
<tr>
<td>Date Stamp</td>
<td>191194</td>
<td>UTC Date of fix, 19 November 1994</td>
</tr>
<tr>
<td>Magnetic variation</td>
<td>020.3,E</td>
<td>20.3 degrees. East</td>
</tr>
<tr>
<td>Checksum</td>
<td>E*68</td>
<td>Used by program to check for transmission errors</td>
</tr>
</tbody>
</table>

$GPGSV - Satellites in View

Output:
$GPGSV,2,1,08,01,40,083,46,02,17,308,41,12,07,344,39,14,22,228,45*75

<table>
<thead>
<tr>
<th>Name</th>
<th>Example Data</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sentence Identifier</td>
<td>$GPGSV</td>
<td>Satellites in View</td>
</tr>
<tr>
<td>No. of Messages</td>
<td>2</td>
<td>Number of messages for all data</td>
</tr>
<tr>
<td>Message number</td>
<td>1</td>
<td>Sentence 1 of 2</td>
</tr>
<tr>
<td>Number of SVs in View</td>
<td>08</td>
<td>8 satellites in view</td>
</tr>
<tr>
<td>SV PRN Number</td>
<td>01</td>
<td>Satellite PRN Number</td>
</tr>
<tr>
<td>Elevation (deg.)</td>
<td>40</td>
<td>40 degrees</td>
</tr>
<tr>
<td>Azimuth (deg.)</td>
<td>083</td>
<td>Degrees from true north (000 to 359)</td>
</tr>
<tr>
<td>SNR</td>
<td>46</td>
<td>Signal to Noise ratio, higher is better for up to 4 satellites per sentence</td>
</tr>
<tr>
<td>Checksum</td>
<td>E*75</td>
<td>Used by program to check for transmission errors</td>
</tr>
</tbody>
</table>
Appendix G.

3G Module Development

This section contains the necessary code that was required for integrating the 3G module for the AIVI system.

1. **Enable Modules in Linux Kernel**

The necessary modules that need to be enabled are:

- **CONFIG_OPTION** – The ‘options’ 3G modem driver required for ZTE and Huawei based modems (including the Vodafone Vodem used).
- **CONFIG_USB_SERIAL GENERIC** – Generic USB serial driver, used to provide a serial connection via a USB interface.

These modules are enabled by editing the kernel configuration file, enabling the appropriate flags (i.e CONFIG_OPTION=y), and re-compiling the kernel to include the appropriate modules.

2. **Switch to RIL mode:**

   Issue AT Command - AT+ZCDRUN=8 (use 9 to enable).

   Command is issued via the PPPD (point-to-point daemon). A utility used for serial modems.
3. Setup and Connect scripts

Point-top-point daemon (PPPD) script

```bash
# /etc/ppp/peers/vodafone
# Vodafone NZ pppd script (ZTE K3765-Z)
# Usage: root > pppd call vodafone

/dev/ttyUSB3  # Interface/Communication channel used for 3G.
               # The node is generated once the kernel detects
               # the modem from the loaded modules
               # (4 USB nodes are generated).

460800  # Communication Speed

# Enables hardware handshaking

modem  # Device type

noauth  # Authentication method (No Authentication
         # specified).

debug  # Enable debugging

nodetach  # No detach mode (do not detach/drop).

usepeerdns # Ask peer (Vodafone) for DNS server address
	noipdefault # No default IP.

defaultroute # Use default route (No specific routing).
```

Figure 47 - 3G Modem: PPPD Script

Connection script

```bash
#!/etc/ppp/gprs-connect-chat
# chat script for Vodafone Vodem

ABORT "DELAYED"

ABORT "NO CARRIER"

ABORT "BUSY"

ABORT "ERROR"

ABORT "NO DIALTONE"

# Abort connection if response was: Delayed, No carrier (ISP),
# Busy signal or Error on connection

TIMEOUT 40
# Sets connection timeout (40 seconds)

' '  # AT

OK ATZ
# AT Commands used to initiate connection. Expects OK response
# to continue, else throws an error.
# ATZ = Reset modem to default config

OK AT+ZCDRUN=8
```
# Disables ZeroCD Mode

OK              "ATQ0 V1 E0 S0=0 &C1 &D2 +FCLASS=0"
# ATQ0 = Quite off (Enables messages)
# V1   = Verbal messages (Words instead of numerical messages - V0)
# E0   = No Local Echo
# S0=0 = No auto-answer
# &C1  = Modem asserts DCD (Data Carrier Detect) when carrier
text detected used to indicate that the modem is connected
to an external remote modem.
# &D2  = Reset modem when DTR signal (hangup) is set.
# +FCLASS=0  = Select active service (0)

OK              AT+CGDCONT=1,"IP","www.vodafone.net.nz"
# Create Connection ID 1 (AT+CGDCONT=1) for Vodafone APN

OK              ATDT*99#
# Initiates connection to the default Connection ID (1)

CONNECT         ''
# Connects

Figure 48 - 3G Modem: Connect Script

Automatically start the connection on start-up

# [root]/init.rc
...

# Run connect script
service ppp /system/bin/pppd call vodafone
    user root
group system radio
...

Figure 49 - 3G Modem: Init command to start 3G services.
Appendix H.

System Evaluation Survey

Sample copy of the survey used to collect feedback from participants regarding the proposed system including participant satisfaction of the system features demonstrated by video.
## Android In-Vehicle Infotainment System: Video Feedback

How would you rate the following features demonstrated in the system?

Please choose from then leave a comment in the box provided:
1 – Very Unsatisfied; 2 – Unsatisfied; 3 – Neutral; 4 – Satisfied, 5 – Very Satisfied

<table>
<thead>
<tr>
<th>Feature</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multimedia Features</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connectivity</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Navigation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Third-party Applications</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Comments and feedback regarding the system demonstrated.

Thank you for participating in this study.

You contribution to this research is invaluable for helping developers understand what the end-users needs and requirements are to building an effective system.

We appreciate your honesty and willingness to assist with this important research.
Appendix I.

System Evaluation Results

Result data from the system evaluation survey that was conducted for the AIVI system.
# Feature Satisfaction

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Multimedia</th>
<th>Connectivity</th>
<th>Navigation</th>
<th>Third-party Applications</th>
<th>Reverse Camera</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td>3/03/2011 1:48</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 2:18</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 4:33</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 5:33</td>
<td>3</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 8:05</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 8:55</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 9:31</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>3/03/2011 9:41</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 10:02</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 12:12</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 15:31</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 15:49</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>3/03/2011 15:58</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>3/03/2011 16:34</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4/03/2011 0:46</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4/03/2011 3:26</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4/03/2011 7:10</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>4/03/2011 7:57</td>
<td>5</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>4/03/2011 12:30</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>4/03/2011 12:52</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4/03/2011 15:35</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>4/03/2011 22:59</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>5/03/2011 1:58</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>5/03/2011 5:07</td>
<td>5</td>
<td>2</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5/03/2011 5:55</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>5/03/2011 6:25</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>5/03/2011 8:03</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5/03/2011 10:57</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>5/03/2011 11:35</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>5/03/2011 11:53</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>5/03/2011 14:30</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>5/03/2011 16:31</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>5/03/2011 20:48</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>2</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>5/03/2011 23:39</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6/03/2011 1:58</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6/03/2011 5:35</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6/03/2011 9:35</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6/03/2011 10:47</td>
<td>1</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>6/03/2011 12:40</td>
<td>1</td>
<td>5</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>6/03/2011 16:15</td>
<td>5</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Timestamp</td>
<td>Comments On</td>
<td>Multimedia Features - Comments</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>---------------</td>
<td>-------------------------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 9:31</td>
<td>Multimedia</td>
<td>I’m worried about playing videos while driving.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 15:49</td>
<td>Multimedia</td>
<td>Will this support DIVX and other video formats?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6/03/2011 12:40</td>
<td>Multimedia</td>
<td>Would like to see this synchronise with my personal multimedia devices... Like iPod, my phone etc.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 4:33</td>
<td>Connectivity</td>
<td>Worried about the keyboard use</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 10:02</td>
<td>Connectivity</td>
<td>Wouldn't data be expensive?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 15:31</td>
<td>Connectivity</td>
<td>Internet browser looks a bit distractive</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 15:49</td>
<td>Connectivity</td>
<td>Is this safe?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 15:58</td>
<td>Connectivity</td>
<td>Awesome!</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 8:05</td>
<td>Navigation</td>
<td>One concern, will this always use data?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 1:48</td>
<td>Applications</td>
<td>This is exactly what we need!</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 12:12</td>
<td>Applications</td>
<td>Will I be able to load applications from external storage (USB etc.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 4:33</td>
<td>Reverse Camera</td>
<td>Would be nice if the reverse camera started automatically</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 16:34</td>
<td>Reverse Camera</td>
<td>I like it. But I don't think we should start the application manually to use the reverse camera.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 5:33</td>
<td>Overall system use</td>
<td>This covers all your in car entertainment needs, bar being a ps3 in itself! Very impressive, nice looking visual interface too!</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3/03/2011 12:12</td>
<td>Overall system use</td>
<td>Wow dude, thats really cool! I'd love one of these for my car. If you could get Honeycomb on that bad boy it'd be sweet! Just wondering, how do you plan on setting up a rear view camera? Just connect a camera at the back of the car to the tablet itself, then run the Camera app? Seems like the simplest way to me..</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4/03/2011 7:10</td>
<td>Overall system use</td>
<td>That's crazy. Very distraction while your driving</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4/03/2011 15:35</td>
<td>Overall system use</td>
<td>This is incredible, you did such a great job, I would buy one!</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5/03/2011 1:58</td>
<td>Overall system use</td>
<td>Great job. One thing I would like to see is more safety features.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5/03/2011 11:53</td>
<td>Overall system use</td>
<td>Some concern regarding safety, but all in all its something I'd love to see in my car.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5/03/2011 16:31</td>
<td>Overall system use</td>
<td>What about making phone calls? Skype?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6/03/2011 5:35</td>
<td>Overall system use</td>
<td>I hope it will support handsfree kit.</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Appendix J.

**Code Repository**

This section will provide instructions on how to retrieve the source-code implemented for this project.

The GIT repository is hosted in github.com at [https://github.com/tariknz/AIVI](https://github.com/tariknz/AIVI).

The repository is broken up into subsection to contain specific modules for the AIVI project:

<table>
<thead>
<tr>
<th>Module</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3g</td>
<td>Scripts for establishing a 3G connection with the service provider (Vodafone) used.</td>
</tr>
<tr>
<td>AndreoCarHome</td>
<td>The AIVI CarHome application launcher. Contains all source code and resources used for this application.</td>
</tr>
<tr>
<td>dev_scripts</td>
<td>These scripts are used to assist in development of AIVI.</td>
</tr>
<tr>
<td>gps</td>
<td>Contains modified source-code for the GPS library used for AIVI.</td>
</tr>
<tr>
<td>images</td>
<td>Compiled Android O.S and Linux Kernel images ready to be flashed onto the same device.</td>
</tr>
<tr>
<td>mic</td>
<td>Modified mic components to allow the twl mic to interface with Android.</td>
</tr>
<tr>
<td>other</td>
<td>Contains other uncategorised code.</td>
</tr>
</tbody>
</table>


Birnbaum, R., & Truglia, J. (2001). Getting to Know OBD II.


Microsoft. (2010). Kia Motors and Microsoft Usher in New Era of In-Car Technology.


MSN Direct. (2010). MSN® Direct Service Announcement: MSN Direct service will be available only until January 1, 2012.


Qt community, & Nokia Corp. (1992). Qt: A cross-platform application and UI framework.
Redirect Intelligence Corp. (2010). RockPlayer: an embedded platform, high performance, almost all format video player.
AUTOMOBILTECHNISCHE ZEITSCHRIFT, 105(4; SPI), 50-53.
Trout, C. (2011). Nuance opens Dragon Mobile SDK to app developers, we see end to embarrassing dictation. Engadget.
Waze Mobile. (2010). Waze: Real-time maps and traffic information based on the wisdom of the crowd.