Mobile Application for Monitoring Driving Techniques

By Luke Styne

BSc (Hons) Computing Science

Staffordshire University

A project submitted in partial fulfilment of the award of the degree of BSc(Hons) Computing Science from Staffordshire University

Supervised by David Thomas

Assessed by Kelvin Hilton

April 2016

CONTENTS

TABLE OF CONTENTS

TABLE OF FIGURES

CHAPTER 1 – BACKGROUND

1.1 – PROJECT CONTEXT AND INTRODUCTION

Telematics – a generic term that loosely defines the combination of telecommunication and informatics. Telematics includes anything from GPS systems to navigation systems (Telematics.com, 2015). With the increase in availability of reliable mobile data communications, the use of telematics has been an upcoming phenomenon with the use of monitoring vehicles. Many car insurance companies now offer deals where they will fit a box in the car that uses accelerometer and GPS technology to monitor the driver of the car. The insurance company will react dependant on the score that the driver achieves. The boxes are connected to a mobile network, similar to a phone, and submits the data recorded from the user's journey (Confused.com, no date). Insurance companies use the technology to show users what their driving standard is like and can adjust their premiums accordingly, e.g. increasing their premium if the user is not driving well and vice versa. The idea behind telematics devices is to encourage people to drive more safely, reduce accident statistics and make the roads safer for everyone. In addition, reducing premiums for users thus making the insurance company more desirable for business and therefore increasing profit. Mostly aimed at young drivers as their premiums are substantially high (Confused.com, no date). The use of telematics for these insurance companies involve a special unit being installed into the car, and the user entering into a 12-month contract with the company for the insurance cover (Confused.com, no date). It is inherently more controllable for the insurance company to have their own device fitted into the car by an expert so they have tighter control over the variables used, resulting in more accurate data. However, this does pose a negative side to the users. Some users feel being monitored all the time has privacy issues. Also some of the specialised insurance impose curfews that warn and eventually charge the user if they drive after 11pm (Confused.com, no date) which users might see as burden.

Some insurance companies started to develop applications that could do this telematics style monitoring but using the user's smart phone. These applications act similar to the dedicated devices that are fitted, the user's smartphone GPS and accelerometer data are used to record the driving style, the collected data is then used by the insurance companies to give feedback and rate the user. However, the concept of using telematics via smartphone apps is different. Inherently it is impossible to guarantee that the user will always have the smartphone available to record their driving. Compared to the dedicated physical devices that are fitted to cars where they are always available to record driving. Therefore, the insurance companies changed the way they marketed the apps; rather than selling the use of telematics as being part of the drivers 12-month insurance policy, the companies offer the app for personal learning, so they can see areas of improvement, but mostly as an incentive to potentially decrease premiums, in the case of the Aviva Drive application, if the user covers 200 miles and their driving is of good standard (Aviva, 2015). It is clear to see that the concept must work otherwise the companies would not be offering discounts to do so.

1.2 – IDENTIFIED PROBLEM

The current metrics that telematics applications use: GPS and accelerometer data, identifies a clear limitation in gathering accurate statistics. GPS and accelerometer data can only be used to gather statistics on cornering, speed, acceleration and time of journey. What about the distance between the car in front? Having completed some preliminary research into vehicle accidents, there are many sources that speak about tailgating being in the top list for cause of accidents (Law Offices of Michael Pines, APC, no date) (Brake, 2014). Therefore, these applications could be giving the user a high rating, reporting the data back to the insurance company as driving within the speed limit, cornering appropriately, accelerating smooth etcetera, yet the driver could be placing themselves at risk by driving too close to the vehicle in front. Statistically making themselves lower rated drivers but the app and insurance companies cannot visualise this. Questioning the accuracy and validity of the gathered statistics on drivers.

If the distance between the driver and the vehicle in front can be used in conjunction with the other metrics for recording a user's journey then the accuracy of the statistics could be vastly improved, in a commercial sense be extremely useful for insurance companies, maybe giving the ability to offer greater discounts, and increasing profits, but on a personal level also to help the user too. If the app could do this in real-time and warn the user as they get too close could help the user identify that they are putting them self in danger and action upon that, therefore increasing driver safety.

CHAPTER 2 – PROJECT SCOPE

2.1 - PROJECT OBJECTIVES

This project aims to research and analyse current drive monitoring applications, with the inclusion of extensive domain research based on statistics, government guidelines, and findings relating to the areas involved that will ultimately lead to the design and development of a mobile application that is a proof of concept that will monitor the distance the user drives behind the vehicle in front.

Research will also include the methodologies undertaken and review technical areas that could be used to implement the application and justifications to why the chosen tools was used. All research will be carried out using resources including journals, online books, hard-copy books and online resources. Further research will be carried out by using a mobile device to download and review the current applications available in the domain area.

To ensure the project follows all the required ethics standards set out by Staffordshire University and the British Computing Society a full ethics statement will be provided.

2.2 - PROJECT DELIVERABLES:

- Justified and researched mobile application that will be fully tested and developed to a high standard of quality as specified in this report.
- Analysis
- Design documentation
- Testing strategy
- Test plans
- Logs
- Recordings of supervisory meetings
- Conclusion and Evaluation

The application will be evaluated against the following criteria:

- Fulfils the requirements set out in the report
- Tested
- Code being maintainable
- Easy to use
- Bug-free

CHAPTER 3 - RESEARCH

3.1 – RESEARCH METHODS

Primary

Primary research is gathering new research, undertaken to answer specific questions or gather more information on areas that cannot be found by reviewing current literature. This can involve questionnaires, surveys or interviews (Business Case Studies LLP, no date).

Secondary

Secondary research is the summary and collation of existing information that is publically available. This can be published articles, books, conference proceedings etc.

Techniques

There are 2 types of research techniques, qualitative and quantitative:

Quantitative

Used to quantify the problem by generating numerical data that can be used in statistics. Used to quantify opinions and behaviour (Business Case Studies LLP, no date). Using measurable data to uncover patterns and facts. Some methods include surveys and online polls.

Qualitative

Gain understanding of opinions motivations. Insight into areas that cannot be done mathematically. Qualitative research methods vary from semi-structured to un structured techniques (Wyse, 2011). Primary Qualitative research can be conducted by interviews, focus groups. Secondary qualitative research could be reviewing of existing literature found for the problem areas.

3.2 - RESEARCH PLAN

From the preliminary investigation into identifying the current problem that this project will address there are some areas identified that need to be researched. The project will require research into the following areas:

- Discipline – Methodology to conduct the project
- Domain area
  - Stopping distances
  - Tailgating
  - Collision statistics
  - Telematics
  - Current applications within the same problem area
- Technology – the technology used to implement the project.
- Testing Strategy

For this project, secondary research will be conducted to gain information into the areas that are required. Qualitative techniques will be used to gain deep information on the areas, and qualitative techniques will be used to gather information such as statistics that will collectively form conclusions and provide information for the analysis, design and implementation of the project. The initial areas listed above will form starting points and if extra areas arise as results of findings in these areas then these will be documented accordingly.

If throughout the secondary research it becomes clear that primary research is seen as a benefit to the project, then this will be conducted and documented.

3.3 – METHODOLOGY

The following section will discuss the various development methodologies available that can be followed and documented throughout the project lifecycle. Each methodology will be described, key characteristics highlighted and evaluated based upon the needs of the Driving Monitoring Mobile Application project. An overall conclusion will be made on the chosen design methodology if one fits, or a customised hybrid methodology will be discussed that will compliment all the characteristics of the project.

3.3.1 – SOFTWARE DEVELOPMENT LIFECYCLE (SDLC)

SDLC is a conceptual model that describes the stages involved in system development. Described by Avison and Fitzgerald (2006) the basic structure as follows:

- Feasibility study – Looking at the current system, the requirements that it was intended to meet, problems in meeting those requirements, any new requirements that have arisen since the system was first implemented and investigate alternative solutions.
- Systems investigation – Detailed fact finding stage:
    - Functional requirements of the existing system
    - Requirements of the new system
    - Constraints
    - Range of data types and volumes that are to be processed
    - Exception conditions
    - Problems of present working methods
- Systems analysis – Analyse the present system:
    - Why do the problems exist?
    - Why were certain methods adopted?
    - Any alternative methods?
    - Likely growth rates of data?

- System Design – Design both computer and manual parts of the system. Documentation including:
    - Input data and how data is captured
    - Outputs of the system
    - Processes, many carried out by computer programs
    - Structure of the computer and manual files
    - Security and backup provisions
    - Testing and implementation plans
- Implementation – Emphasis on quality control. Manual, Hardware and Software to be tested to the satisfaction of users and analysts. User training to be included. Documentations such as operation and user manuals to be included. Security testing to make sure system is robust. Once Testing and quality is assured, a changeover to the new system can be put in place.
- Review and maintenance – Once the system is operational a review is undertaken to ensure that the new system conforms to the requirements set out and the costs have not exceeded those predicted. Leads to a process of organisation learning, where improvements are made to the way other systems are developed. If it is deemed that changes are required such that the operational system is no longer appropriate the SDLC finishes and the cycle starts again.

3.3.1.1 – WATERFALL

There are many variants of the SDLC, most notably is the waterfall model described first by Royce (Royce 1970). Upon which many similar methodologies are based.

Illustrate the process as follows:

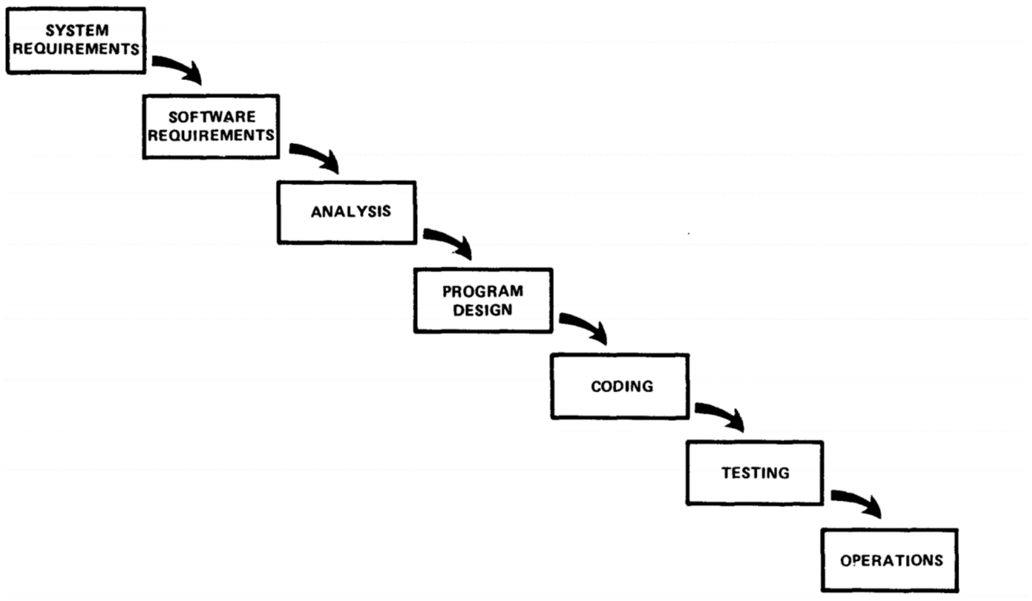


Figure 1 – Implementation steps to develop a solution for a customer (Source: Winston Royce 1970)

The waterfall model is a linear step by step method that begins with defining the system requirements, followed by the software requirements. Then analysis of the system, program design, coding, testing and operations. As you can see each step must be completed before the next step can be carried out.

PROS

- Allows a structured approach
- Understandable and explainable phases
- Identify Milestones in the development process (Hughey, 2009)
- Design errors are captured before any software is written, saving time during the implementation stage. (Hughey, 2009)
- Cost of the project accurately measured after requirements are defined (Hughey, 2009)
- Testing is easier as the product can be referenced to the scenario defined in the functional specification (Hughey, 2009)

CONS

- Clients will often fail to state their requirements at the functional specification stage, requirements are only fully appreciated when an application is delivered (Hughey, 2009)
- Does not cater for requirements changing during the development cycle – In the Proceedings of IEEE Wescon August (1970) Royce states himself “I believe in this concept, but the implementation described above is risky and invites failure.” Later stating “The required design changes are likely to be so disruptive that the software requirements upon which the design is based and which provides the rationale for everything are violated. Either the requirements must be modified, or a substantial change in the design is required. In effect the development process has returned to the origin” In agreement with Royce the waterfall process clearly highlights the issue where if requirements are to change so much that the previous stages are violated one must start the whole process again, occurring extra costs and time for the project.

SUMMARY

The waterfall methodology is a static linear walk through of the entire project. Documenting every aspect of the system requirements, analysis, design, and implementation and testing. This methodology does produce a fully functional artefact at the end of the process that satisfies the documented requirements. Compared to other methodologies like agile, later discussed in this chapter, where a project may never deliver a distinct solution measurable to the requirements of the customer. Although requirements are clearly defined and the methodology is easy to follow, very costly problems can arise if problems are found in later stages. This type of methodology is suited towards projects where requirements are very well understood, concrete and not subject to change.

3.3.2 – STRUCTURED SYSTEMS ANALYSIS & DESIGN METHOD (SSADM)

SSADM is a waterfall based as such the methodology is a staged step through process but concentrates on the analysis and design phase of the Waterfall model (What is SSADM? 2007). SSADM is a 7 stage 5 module framework and covers the life cycle from feasibility to design but not implementation and maintenance. (Avison & Fitzgerald 2006).

See Appendix B.1 for more details on the structure and Processes of SSADM

PROS

From the amount of information available on SSADM, being a British Standard BS 7738-1:1994 (British Standards Institute 1994) and it being a government standard in the engineering of Civil Service applications (Fitzgerald, Avison, 2006) SSADM is a very well tried and tested approach. It is extremely detailed and produces thorough documentation throughout the process. This leads to many advantages:

- Reducing risk – Avison and Fitzgerald state 'It is the most complete analysis and design method on the market' further 'SSADM's proven track record makes it a safe choice' Here they are describing that these aspects of the methodology reduces the risk of project failure.
- Quality – The methodology is extremely rigorous, involving users throughout the process with quality criteria to compare to.
- Hardware and software independence – Separation of logical design from the physical implementation allowing the solution to be implemented on any hardware software environments. (Bentley, 1996)
- Structured – Common understanding
- Good for visualising and designing for solutions that use a large amounts of data – SSADM documents the logical and physical design of data, structures, interactions and relationships.

CONS

- Over analysis – SSADM emphasises the analysis of the system and documentation. This presents danger of over analysing, resulting in much time and money expense. (ITC Infotech India LTD, No date). However, if a large organisation is undertaking many projects then the business can benefit from reuse of documentation in SSADM as many aspects of the business may have already been analysed and documented (ITC Infotech India LTD, no date).
- No implementation and testing phase – SSADM stops at the end of the design phase and doesn't specify the implementation or testing section of a systems SDLC.

SUMMARY

In summary SSADM is a very rigorous and structured approach to developing an information system. Using tools like Data Flow Diagrams to model business processes; various entity modelling tools for data, thoroughly documents the ins and outs of the system ready to be implemented. SSADM also being part of a British Standard and used by the government shows how effective the methodology can be at producing systems that meet the requirements of the users. As SSADM emphasises the analysis and design phase of the current and proposed systems, much documentation is produced for the flow, structure, processes around the data in the system. This makes it a good methodology to follow if the system proposed is to include large amounts of data processing, flow and storage.

3.3.3 – RATIONAL UNIFIED PROCESS (RUP)

The Rational Unified Process is a process framework for iterative-incremental software development. (Shuja, 2008). RUP has been around for a long time. The first version of what is known as RUP was released in 1998, but was influenced by the predecessor called Objectory, from 1988. (Shuja, 2008). RUP is a guide on how to successfully use Unified Modelling Language (UML) (Rational Software, 2001). UML was originally created by the company Rational Software, and is maintained by the standards organisation Object Management Group (OMG), it became an industry standard language allowing clear communication of requirements and designs (Rational Software, 2001). Later IBM acquired Rational in 2003 (Taft, 2002).

The process is described in two dimensions. Best represented in a graph:

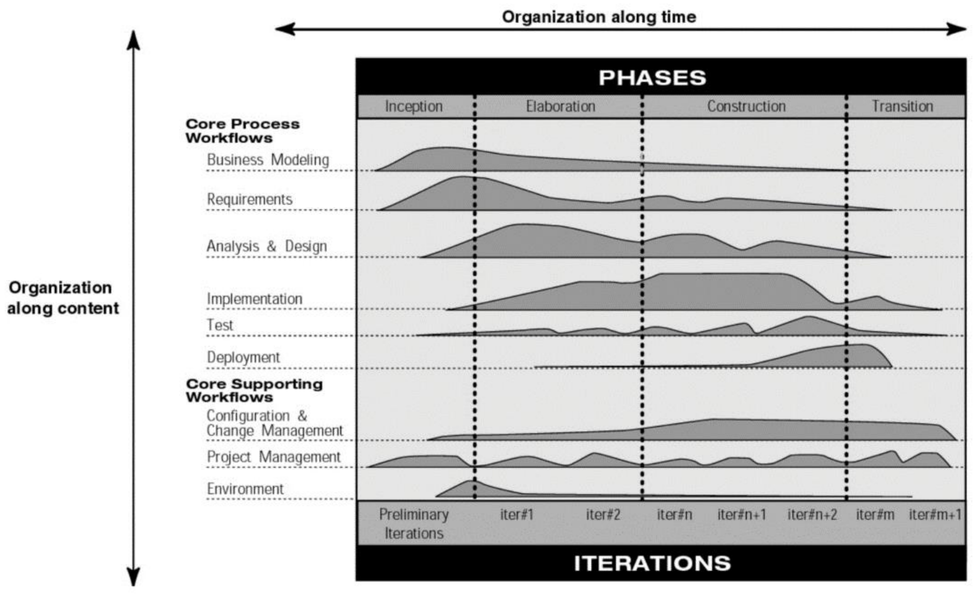


Figure 2 – Iterative model showing the process structure (source: Rational Software, 2001)

The life cycle is broken down into 4 iterations – inception, elaboration, construction and transition. Each phase ends with a well-defined milestone – key goals must have been achieved (Rational Software, 2001).

The RUP uses the UML throughout the process to document the aspects of the system and for requirements gathering. There are numerous different Diagrams specified in the UML. They are split into two categories:

Structure Diagrams – Static structure of the system of different abstraction and implementation levels and how they relate to each other. Representing concepts of the system, including abstract, real world and implementation concepts (uml-diagrams.org).

Behaviour Diagrams – Dynamic behaviour of objects in a system, can be described as a series of changes over time (uml-diagrams.org).

Structure Diagram

See appendix A.1 taken from uml-diagrams.org of a list of the structure diagrams:

Behaviour Diagrams

See appendix A.2 taken from uml-diagrams.org of a list of the behaviour diagrams:

PROS

- High Success rate – According to Shuja written in the book IBM Rational Unified Process Reference and Certification Guide: Solution Designer (RUP) he states “No other modern software engineering process has a higher adoption or success rate than RUP, which is attracting more and more organizations.” Although this is not backed up by any statistics the acquisition by IBM and the on-going development of the RUP and UML shows that the methodology is being used by many organisations with success.
- Complete methodology from start to finish –RUP covers all aspects of the product life cycle
- Emphasis on accurate documentation
- Proactive at managing evolving requirements – RUP has iteration stages with all workflows overlapping each other (see figure 2) this allows for a much more fluid process for any changes. Of course with careful change management to control the outcome. (TatvaSoft.com - Anon 2014).
- Less emphasis on integration – integration is part of the entire development process.

CONS

- Developers need to be an expert in their work – Multiple sources have a common opinion on the RUP. They all indicate that RUP is a complex process and that the developers need to be experts to successfully use this methodology (Zeil 2014) (My-Project-Management-Expert.com 2009) (TatvaSoft.com - Anon 2014)

- Complex Process
- Integration throughout the life cycle can be confusing – Big projects with multiple development streams can add confusion and cause issues during testing (My-Project-Management-Expert.com 2009)

SUMMARY

RUP is an object orientated methodology, concerning all stages of the system life cycle from inception to deployment and testing. It is an iterative-incremental model (Shuja, 2008) that allows small flexibility with changes to the requirements of the system, implementing several workflows that overlap each other during the process (see figure 2). RUP uses the industry standard UML to document all the processes in analysing requirements and designing the system (Rational Software, 2001). Therefore, using an industry standard means that there is solid, well known knowledge of the processes involved and can be recognised by many people in the industry, comparatively to bespoke techniques where only a select audience might only know. This widens the usability of RUP across multiple teams with different domain skills, allowing them to interpret and design software with greater success.

3.3.4 – DYNAMIC SYSTEMS DEVELOPMENT METHOD (DSDM)

DSDM is a framework for Agile project management. (DSDM Consortium). A group of developers from companies interested in Rapid Application Development (RAD) came together in 1994 to form a consortium to define a standard RAD Methodology (Avison & Fitzgerald 2006). RAD had been a methodology designed for minimal planning in favour for rapid prototyping, following incremental and iterative flow (tutorialspoint.com) Documented very early on by James Martin in the book Rapid Application Development published in 1991 (Martin, 1991). The consortium was concerned that RAD was becoming associated with a 'quick and dirty' approach, and that RAD should not only be rapid but also disciplined with high quality delivery. (Avison & Fitzgerald 2006). This lead to the advent of DSDM.

DSDM has 8 key principles (dsdm.org)

1. Focus on the business need – decisions should be made in the view of achieving the overall project goal.
2. Deliver on time – good way to demonstrate control over the evolution of the solution.
3. Collaborate – active cooperation outperforms individuals working in loose association.
4. Never compromise Quality – agree level of quality at the start. Aim to achieve this level.

5. Build incrementally from firm foundations – analysis and enough up front design (EDUF) to establish strong foundations.
6. Develop iteratively – Details and requirements emerge late, embrace change, and build business feedback into the process.
7. Communicate continuously and clearly – communication is a key factor for project success. Honesty, transparency, workshops, face to face communication, lean documentation and visual communication like modelling and prototyping all attribute to fulfilling this principle.
8. Demonstrate control – visible plans to all involved in the project. Proactive management, measuring progress by delivery of products. Evaluate project based on business objectives.

(DSDM CONSORTIUM 2014)

An overview of the DSDM lifecycle visualised in the diagram below:

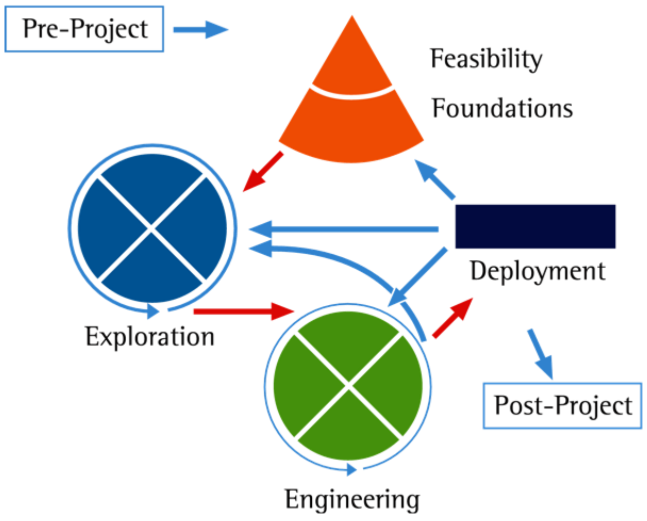


Figure 3 – DSDM Lifecycle (source: DSDM Consortium)

Pre- Project

- Business problem to be addressed (DSDM Consortium, 2014)

Feasibility & foundations

- Establish feasibility of the project. Identify benefits, estimates of timescale and cost. (DSDM Consortium, 2014)
- High level requirements and priority. (DSDM Consortium, 2014)
- Identify information used by the proposed solutions. (DSDM Consortium, 2014) – Can contain Data Flow Diagrams (DFD) and entity models (Avison & Fitzgerald 2006)
- Describe quality assurance (DSDM Consortium, 2014)
- Baseline schedule for development and deployment activities for the solution (DSDM Consortium, 2014)

- Risk management assessment (DSDM Consortium, 2014)
- Start designing the solution architecture, identify physical and infrastructural elements (DSDM Consortium, 2014).

Exploration

- Iteratively Investigate detailed business requirements – translate them into a viable solution (DSDM Consortium, 2014)
- Demonstrate a functional solution to meet the needs of the business. (DSDM Consortium, 2014)
- Evolve System Architecture Definition and Business Area Definition into models describing how the solution works (DSDM Consortium, 2014).

Engineering

- Iteratively evolve solution created in the exploration stage to achieve readiness for delivery. (DSDM Consortium, 2014).
- All 'must haves' and some 'should haves' delivered (Avison &Fitzgerald, 2006)

Deployment

- Get solution into live use (DSDM Consortium, 2014)
- Train users and provide documentation to support (DSDM Consortium, 2014).
- Assess if the deployed solution meets the benefit described in the Business case (DSDM Consortium, 2014)
- Evaluate performance and viability of the project and re-plan as required (DSDM Consortium, 2014).

DSDM has these sections that define the life cycle of the development. The diagram shown in figure 3 has red arrows indicating the natural flow through to the next stage when the acceptance of products has been agreed. (DSDM Consortium, 2014) The blue arrows indicate the paths that can be taken to re visit the other sections if business cases are not met and need to be re-evaluated or improved (DSDM Consortium, 2014).

MoSCoW

DSDM has a process for defining the importance of aspects in the system called MoSCoW prioritisation. Where the time aspect of a project is fixed prioritising requirements, tasks, products, use cases, user stories, acceptance criteria and tests is needed (DSDM Consortium, 2014). MoSCoW Stands for:

- **M**ust Have
  - Cannot deliver on time without
  - Not legal without
  - Unsafe without
  - Cannot deliver the business case without

- **S**hould Have
  - Important not vital
  - Not ideal to leave out but solution still viable
  - May need workaround

- **C**ould Have
  - Desirable but less important
  - Less impact if left out

- **W**on't Have this time
  - Agreement that the project will not deliver the requirement
  - Documented to clarify scope to avoid being reintroduced later

(DSDM Consortium, 2014)

Comparatively simply saying requirements are of High Medium or Low importance are not specific. Must, Should, Could and won't increases implies the result of failing delivery of that requirement (DSDM Consortium, 2014).

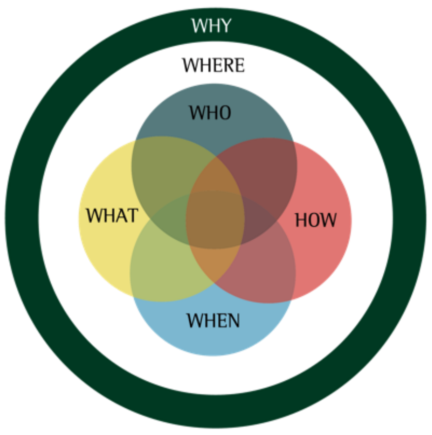
Timeboxing

Another technique in DSDM is timeboxing. DSDM Consortium 2014 defines time boxing as: "a well-defined process to control the creation of low-level products in an iterative fashion, with several specific review points to ensure the quality of those products and the efficiency of the delivery process." Each timebox comprises of three stages – investigation, refinement and consolidation (DSDM Consortium, 2014). Each timebox reflects a pass-through in the iterative development cycle and is used on all sections of the life cycle (see figure 8) (DSDM Consortium, 2014).

Modelling

DSDM defines the use of modelling important to improve communication and to help developing products (DSDM Consortium, 2014). Consider perspectives of what, where, when, how, who and why also the relationships between them. (DSDM Consortium, 2014). The DSDM Consortium also states that it does not recommend creating models for all of them unless it is useful to do so.

The diagram below shows in illustration of the viewpoints of modelling:



WHY	Rationale, ends and means	WHAT	Data and Relationships
WHERE	Locations	WHEN	Events, time and scheduling
WHO	People and Responsibilities	HOW	Processes and Outputs

Figure 4 – Modelling viewpoints in DSDM (source DSDM Consortium, 2014)

The DSDM Consortium states "it does not limit itself to particular modelling techniques, although there are some industry-standards. In a software development project, UML, user stories and class models may all be used" (DSDM Consortium, 2014). From this we can ascertain that there is no solid approach to modelling techniques. Compared to the RUP (explained previously in this document) where it is clear that UML is the only modelling technique to be used in the RUP methodology, DSDM allows for a flexible choice, permitting any modelling techniques are appropriate to capitalise on the skillset available within the organisation. DSDM also states "do enough modelling and no more" (DSDM Consortium, 2014). Meaning that modelling needs to be done to a sufficient extent but not to waste resource on doing too much that isn't needed.

PROS

- Modelling tool flexibility – as described by the DSDM Consortium there is no specific modelling technique described, allowing the flexibility of using multiple techniques to communicate effectively in the modelling process (DSDM Consortium, 2014)
- Provides structure to agile approach – Systems are analysed, defined and developed in an iterative way but structured using DSDM. Using MoSCoW prioritisation and Timeboxing allows for clear, on-time deliverables.
- High level of user involvement – DSDM requires the user's interaction throughout the process of requirements gathering, development and testing iterations of the product. Most likely they are going to be in tune with the system as it develops increasing acceptability and adoptability for when the system is delivered. (DSDM Consortium, 2014)
- Speedy Delivery – important features are defined and delivered iteratively allowing for working software to be delivered quicker.

- Fully equipped to handle requirements change – requirements definition is part of the iterative process, meaning that each iteration reviews the requirements and acts upon them, therefore changes can be reacted to quickly and effectively.

CONS

- Robustness of code can be questionable – DSDM is an iterative process, using timeboxing to define deliverables within a certain time frame. (DSDM Consortium, 2014). Such way of development could lead to rushing delivery to satisfy requirements at the expense at robust code.
- Requires careful management- MoSCoW and timeboxing require careful management to make sure the requirements are justified and that time estimations are as accurate as possible. If estimations deviate too far from the actual result, then problems can occur delivering on time and compromising the success of the project.

SUMMARY

In summary DSDM is an agile approach to systems development. It covers all aspects of the life cycle of the system from proposal to implementation and testing. It is an iterative methodology involving users throughout the whole process. Utilising techniques like MoSCoW for prioritising requirements and Timeboxing for defining time and deliverable constraints for aspects in a particular iterative section. DSDM emphasises that modelling should be carried out throughout the entire lifecycle to ensure clear communication, design, implementation and testability, although any modelling techniques are allowed that complement the organisation to successfully carry out the DSDM methodology. Working software will be delivered quickly with incremental releases providing more functionality through the process. With close association to stake holders and users, requirements are proactively managed, allowing for change and delivered in a timely manner.

3.3.5 – SCRUM

Scrum is an agile framework for software development (Scrum Alliance, n.d.). It is an incremental approach using one or more teams with 7 people in each team (James, 2012).

Scrum consists of:

- A product owner that creates a wish list called a product backlog
- Sprints are short, usually 2-4 weeks that specifies a timeframe in which to complete development of work
- ‘Daily Scrums’ are a meeting each day to assess progress
- Sprint planning involves the team pulling off a small chunk from the top of the backlog and deciding how to implement those tasks

- During the sprint a ScrumMaster keeps the team focused
- At the end of the sprint a potentially shippable product is ready
- Sprints end with a review
- Another sprint begins with choosing another set of requirements off the product backlog. And the process begins again

(Scrum Alliance, no date).

The iterative process continues until a deadline is reached, budget used up, or the backlog is completed.

The process is visualised in the diagram below:

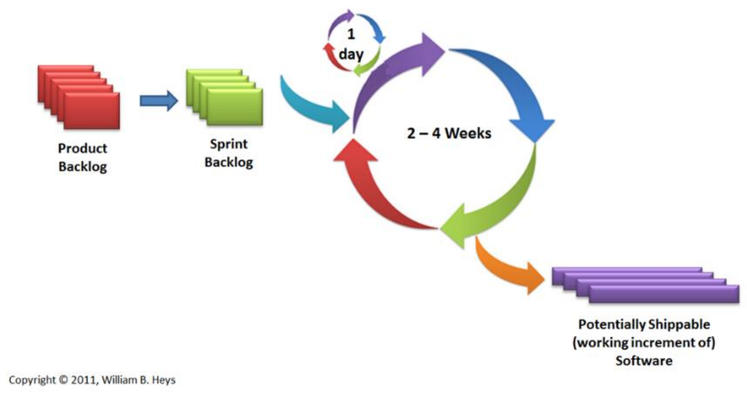

1 day
Product Backlog
Sprint Backlog
2 – 4 Weeks
Potentially Shippable (working increment of) Software
Copyright © 2011, William B. Heys

Figure 5 – Visualisation of the Scrum process. (Source: Heys 2011, http://blogs.msdn.com/b/billheys/archive/2011/01/18/branching-for-scrum.aspx)

A tool used in scrum is a Burn-Down Chart. This is a graph that shows the remaining effort for the time available. Two lines are plotted against this graph, the ideal effort hours and the effort remaining. The ideal effort hours are a prediction across the sprints as to how many hours should remain. With the effort remaining a reflection of what is thought to be the actual remaining effort (Mittal, 2013). Burn-down charts give visualisation to the team to see how much is remaining on the project, and how progress is being made (Mountain Goat Software, n.d.).

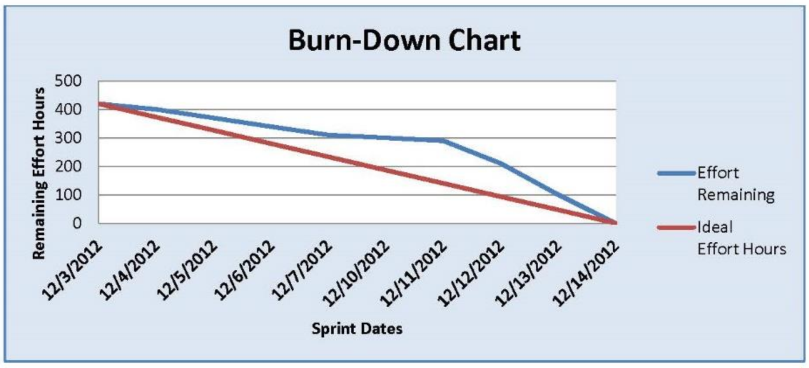


Figure 6 – Example of a burn down chart (Source: Mittal, 2013)

Figure 11 shows an example of a burn down chart that shows the team working at a slow pace for the start of the project, being pushed towards the end (Mittal, 2013).

PROS

- Reduces time to market – Incremental process allows for working software to be delivered quicker giving benefits to the business sooner (Uhlig, n.d).
- Scope and requirements changes are easily facilitated – iterative process with time constrained sprints allows for greater adaptation to requirements changes. Requirements are reviewed after every sprint iteration (Scrum Alliance, n.d.).
- Less time spent on documentation – scrum is an agile framework, of which the foundations of an agile methodology is based on the agile manifesto written by 17 software developers in 2001. (scrummethodology.com, n.d.) Part of this manifesto is “Working software over comprehensive documentation” (Kent Beck et al, 2001). Therefore, minimal or no documentation is produced during the agile development process. Cutting time spent working on documentation.

CONS

- Lack of documentation – as discussed in the pros of Scrum, it is an agile methodology underpinned with the statement from the agile manifesto of working software over documentation (Kent Beck et al, 2001). Leading to potential gaps in knowledge if the product needs to be re-developed later, or modified/improved.
- Difficult to plan – agile projects lack clear definition, with the addition of frequently changing requirements makes it difficult to predict timelines (Uhlig, n.d).

SUMMARY

Scrum is an agile methodology using interactive incremental approaches to software development. Using sprints to lock down a particular set of requirements to be delivered within 2-4 weeks as working software. Using techniques like daily scrum meetings to review the progress of the sprint, and Burndown charts to visualise the effort remaining over time to help the team succeed.

3.3.6 – METHODOLOGY CONCLUSION

The suitability of different methodologies for the drive monitoring mobile application project are comparable. Firstly, following a waterfall approach for this project could bring many benefits to the project. As with a waterfall methodology you get the benefits of the project being very structured, there are set methods that have to be carried out in order to flow onto the next. However, with such a structured approach this makes it difficult if changes arrive further down the project life cycle, requiring much rework of previous sections at the cost of time and expense. SSADM being a waterfall approach that concentrates on analysis and design phases, but does not include implementation and testing. SSADM also concentrates on modelling the current system, data and the proposed logical and physical system to be implemented, emphasis is on data modelling. The proposal objectives for the mobile drive monitoring application is to have the ability to record a journey. For this a way of modelling data will be needed but the application is going to be mostly real-time data that will more than likely not be stored. Leading to SSADM not being the most suitable approach.

The project is solely run, therefore a limited resource to carry out tasks on the project. Following a waterfall approach bares the issue of limited time. As documented waterfall approaches like SSADM require much time spent before any implementation is carried out. Leaving the project in danger of never being delivered due to over analysis. In comparison iterative incremental methodologies provide a shorter time to delivering working software. RUP is an iterative approach that also focuses on using the UML for documentation. UML is very extensive and provides industry standard modelling diagrams to help document the project journey. Compared to SSADM this does include all aspects of the project journey including implementation and testing. However, the methodology is also very documentation intense and complex with multiple phases with multiple iterations.

Agile approaches are more suited to the project as they allow flexibility over many aspects. Firstly, scrum is a very intensive development method, utilising time restricted sprints (2-4 weeks) to deliver a certain amount of requirements pulled from a backlog for the overall project at the end of each sprint. Delivering working software quickly. Using Burn-Down charts to help visualise the approximate effort hours left. Scrum however is defined as to be used in teams of 7 people, with other roles involved like scrum masters that control sprints etc. This makes it unsuitable as this project will be undertaken solely so no teams will exist. In addition to the lack of documentation using this method.

DSDM is comparable to Scrum. DSDM is still an agile development methodology, but was designed to mitigate the negatives associated with agile development: lack of documentation, poor rushed software, undelivered on requirements. DSDM is designed to offer a more structured approach to iterative development. DSDM uses a process called MoSCoW as discussed earlier – allows the project to prioritise requirements in the system, what must, should, could and will not be in the system. This is a good way of locking down requirements so developed software can be delivered to satisfy the business case. This process is highly favoured in application of the mobile driving monitoring application. The scope of the project is likely to increase as further research is carried out in the exploration stage, leading to many requirements that cannot simply be satisfied with the finite resource of the sole project executioner. Therefore, applying MoSCoW will not only show that careful research has been carried out to identify possible features and requirements to further the system but also prioritises what needs to be completed first in order to satisfy the basic business case in which the project is solving. Throughout iterations of the process the 'should' and 'could' can then be looked at if all the 'must haves' are completed. DSDM also specifies that you should use modelling just enough to allow clear communication on the elements of the development process. There is no definition for the type of modelling process used and how much. This flexibility allows for the project to choose what modelling tools are needed to satisfy the approach towards the next step in the methodology process. Compared to scrum where there is no definition to use any documentation at all or a waterfall approach like SSADM where documentation is the main concern of the project. The risk of over documentation and wastage of time and resource is greatly reduced using DSDM.

However, the nature of the project already has a focused scope and deliverable aim as discussed in chapter 2. This discounts some areas of DSDM being entirely useful. The best methodology for this project would be a hybrid between waterfall and DSDM. Requirements capture, analysis and design will be a linear step by step process due to the many constraints that limit the project. Incorporating areas of the DSDM methodology including MoSCoW prioritisation, modelling, timeboxing and iteration of the implementation stage will allow the project to deliver working software in the timeframe allowed. Reduced risk of wasting time and resources on over documentation and flexibility to incorporate changes throughout the process by developing iteratively.

3.4 - CURRENT APPLICATIONS

There are many driving monitoring applications available for smartphones now. Many of these monitoring applications are related to specific insurance companies that are giving incentives for drivers to use the app.

3.4.1 - AVIVA DRIVE

Aviva Drive is an app developed by the insurance company Aviva, it monitors driving skill based on accelerometer and GPS data. After the user has driven 200 miles the app gives them a rating out of 10 and from this rating users can receive a discount on their car insurance quote.

The application has a very intuitive minimalistic UI that is easy to navigate

3.4.1.1 - FEATURES

- View progress
  - Built in maps to see where the user has driven.
  - Summary of completed journeys.
- Earn badges
  - Incentive to earn badges dependant on driving skill
- Social Media
  - Compare scores with friends
- Auto Start
  - IPhone and android users with latest OS version can monitor GPS continually in the background and will automatically record a user's journey without any user input.

3.4.2 - CONFUSED MOTORMATE

Produced by the car insurance company Confused.com but no longer available, Motormate tracks users' journeys and rates them out of 100 based on their acceleration, cornering and speed using GPS and accelerometer. Offering incentives of a free cradle and charger on recording 20 miles, £25 cash back on completion of 250 miles and £25 cash back if the user takes out a policy with confused.com. Users can review journeys and see where events have occurred on the map.

3.4.3 - AXA DRIVOLOGY

Drivology created by AXA tracks user's journeys via GPS and accelerometer data. However, Drivology differs from the other apps, Drivology is designed to be used by the user every time they use their vehicle, it is part of the terms and conditions of taking out an insurance policy with the company that the user must use the app. The incentives are to decrease the user's premiums (via a refund or reduced monthly payments) if their driving standard is good. However, the user's premiums can increase if the user's rating is bad. Creating a more forceful incentive to drive safer.

Drivology is not available for windows phones.

3.4.3.1 - FEATURES

- View progress
  - Built in maps to see where the user has driven.
  - Summary of completed journeys.
- Earn Achievements
  - Incentive to earn achievements dependant on driving skill
- Auto Start

- IPhone and android users with latest OS version can monitor GPS continually in the background and will automatically record a user's journey without any user input.

3.5 – DRIVING STATISTICS

3.5.1 - TELEMATICS

A report written by the Road Safety Foundation for Parliament details a business case for removing Insurance Premium Tax for telematics insurance, although the overall target of the report is not of interest the underlying facts and statistics in the body are of great significance. The report details statistics on the number of crashes and deaths related to under 25 drivers compared to older drivers. Citing that young drivers on average drive half the distance of 40-70 year olds yet have a fatal crash rate 5 times more. The report contains a case study from a telematics insurance provider Ingenie. Within this case study Ingenie state that they report claims and losses are significantly lower than the usual statistics for 17-25 year olds without telematics. Further in the report it summarises the cost analysis of the government dropping the insurance tax for 7 years on telematics policies. Quoting that serious crash reduction of 30% on telematics policies and stating the average cost of a serious crash is £0.4m. Resulting in a total cost saving to the economy of £500m over 7 years (Road Safety Foundation, 2014).

It clearly shows that telematics does work in bringing the risk down for drivers. Resulting in the possibility of expanding the technology and posing the question of: if the introduction of distance tracking technology combined with telematics could these statistics be improved on, making safer drivers and reducing cost to the economy?

In addition, with the report and the research found earlier it is clear that the target market for these type of products are young drivers, typically 17-25, this will be taken into consideration during the requirements capture and design.

3.5.2 – TAILGATING

A report undertaken by Brake, a road safety charity, and Direct Line (2014) on safe driving shows surveys related to speed and tailgating. One of the surveys conducted questioned: "Within the past 12 months on motorways, how often have you left less than a two-second gap between your vehicle and the vehicle in front". Resulting in 57% admitting to leaving less then a 2 second gap at some point, with 28% doing to monthly or more. The report also looks at another survey questioning: Within the past 12 months, have you ever felt concerned about drivers driving too close behind you on motorways? Resulting in 95% feeling concerned at least occasionally. The report then states an interesting point about how a concerned driver could feel intimidated into driving faster or to close to the vehicle in front resulting in greater danger.

On the website of a personal injury law firm Law Offices of Michael Pines, they have a page listing the top 25 causes of car accidents, tailgating being number 14 (Law Offices of Michael Pines, APC, no date). Also on another solicitor website lists tailgating as a common cause of car accidents (JMW Solicitors LLP, no date). Interestingly within the section about tailgating it states that "Car accidents caused by tailgating could be reduced with the help of proximity monitors. These are a fairly new technological invention which detect how close you are traveling to the car in front and automatically adjust your car's speed to prevent you from getting closer than the recommended stopping distance." (JMW Solicitors LLP, no date). Interestingly from the research undertaken there does not seem to be any commercial products that can be fitted to a vehicle out there that can achieve such functionality, however the technology is present in many high end vehicles in Adaptive Cruise Control (ACC), this is a specialised factory fitted option that will automatically slow the car down if there is a car in front too close for the speed currently travelling (Volkswagen UK, no date). However, this technology does not record the user's journey or use metrics to judge a persons driving style.

The tone of the report along with multiple sources describing tailgating as a serious problem that causes accidents shows that there is a problem here that an IT solution can help address.

3.5.3 – STOPPING DISTANCES

The main concentration around the project is measuring the distance between the vehicle in front and the one being driven. This leads onto statistics around stopping distances, the ability to completely stop a vehicle from a given speed. The UK Government Department for Transport offers advice on the matter. They state to allow a 2 second gap between you and the vehicle in front and also published statistics on typical stopping distances at various speeds (Department for Transport, 2015):

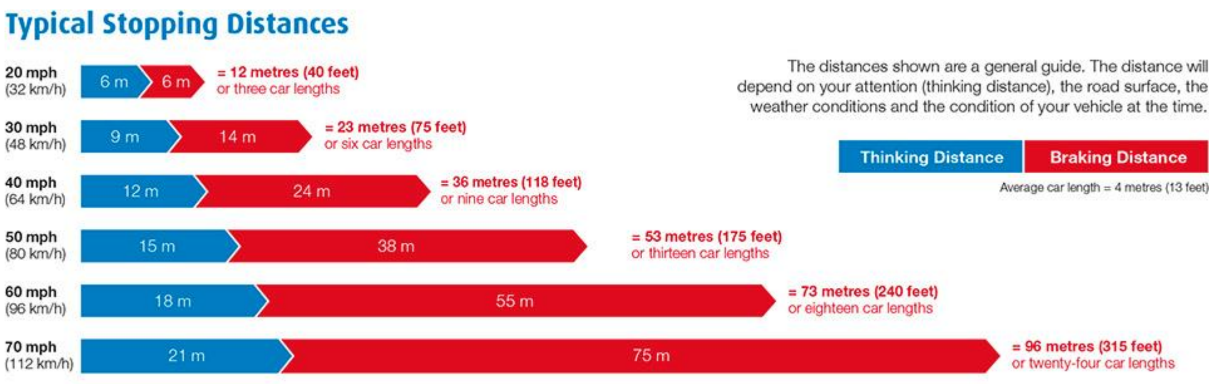


Figure 7 – Typical stopping distances at various road speeds (source: gov.uk)

Therefore, if a person was driving at a speed of 50mph and they were to see a hazard then it would take them 53 meters to stop more importantly if they were driving with a distance of less then 53 meters between the car in front then they are statistically going to collide with the hazard. These statistics will be taken into account in designing of the application so that the application will correlate the current user's speed and the distance between the vehicle in front and act upon the data. These statistics are also marked as typical. Stopping distances are greatly increased when driving in wet and snowy conditions. According to Driving Test Tips they state that stopping distances are multiplied by 2 in the wet (Driving Test Tips, 2014).

From these guidelines it is deduced that the formula for calculating a stopping distance is:

$(\text{speed})^2 \div 20$ + thinking distance = overall stopping distance in feet
For stopping distance formula in metres, multiply the result by 0.3

It is also noted that these are guidelines and that every car will have a different stopping distance due to the variation in braking systems used. It therefore promotes the possibility of analysing and designing a 'training' mode to be incorporated into the application that will give a far greater accuracy of the stopping ability of the vehicle that the user is using the application within. Making the marketability of the application as more accurate and safer.

3.6 – HARDWARE & SOFTWARE

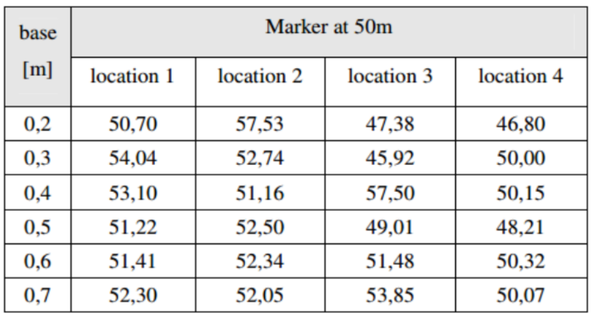
This section will discuss hardware and software implementation technologies available to fulfil the project. Details of potential technologies available will be discussed and how they could be implemented in this project. Advantages and disadvantages will be discussed where appropriate, concluding on the technology chosen to implement throughout the project with a thorough justification.

The aim of this project is to produce a prototype mobile application with the ability of tracking distance in real time between a vehicle being driven and the vehicle in front and to relay this information back to the user to warn of dangers of driving too close to the vehicle in front.

For the most desirable solution the application would be contained in one device (the user's smartphone), with no reliability on external hardware. This does present a challenge but could potentially be possible, discussion in more details as follows:

3.6.1 - STEREOSCOPY

You can measure distance in multiple ways using image processing techniques. Stereoscopy is a technique that enables a three dimensional effect, giving an illusion of depth to a flat image (Rouse, 2011). For this to be accomplished, 2 cameras are needed. When 2 cameras are placed at a certain distance between each other, defined as Binocular disparity, this can be used in multiple functions to calculate a perception of depth (Heeger, 2006). A paper written by Mrovlje and Vrančić (2008) documenting how to calculate distance using stereoscopic pictures, explains that there are 2 ways to measuring distance to objects. Active and Passive. Active methods measure distance by projecting signals onto an object and using sensory devices to read the projection whilst passive only uses information already given by light (photographs, video etc). The paper discusses ways to calculate depth, using trigonometry and algorithms. A similar paper written by Mahammed, Melhum, and Kotchery (2013) explains that many aspects of the calculations require known characteristics of the cameras involved e.g. focal length, viewing angle. Also the cameras must be horizontally aligned and taken at the same instant, in addition to this but regarded as not so important is that the object must be not moving (static) (Mrovlje, Vrančić, 2008). In order to calculate the distance, object recognition needs to be used, when a certain object is on the left picture it will be in a different place on the right picture. Leading for the need to be able to detect where the same object is on the right picture in order to calculate the correct distance (Mrovlje, Vrančić, 2008). Mrovlje & Vrančić, (2008) describes in some detail about how to recognise objects using more algorithms and matrices. Their paper follows up with real life testing using stereoscopic imagery at multiple distances, compared to different size base (distance between the two cameras).



base [m]	Marker at 50m			
	location 1	location 2	location 3	location 4
0,2	50,70	57,53	47,38	46,80
0,3	54,04	52,74	45,92	50,00
0,4	53,10	51,16	57,50	50,15
0,5	51,22	52,50	49,01	48,21
0,6	51,41	52,34	51,48	50,32
0,7	52,30	52,05	53,85	50,07

Figure 8 – results of stereoscopic distance measuring at 50 metres (source: Mrovlje, Vrančić, 2008).

Figure 9 shows an example of the set of results at 50 metres. Looking at their results the measurements are fairly accurate, the average of all the measurements deviation from 50 is 2.38m (4.76 %). Although the maximum deviation was 7.53m (15%) which is high. Referring back to the previous research undertaken about the typical stopping distance for a car, as written by the DVLA. At 50mph the stopping distance is 53 metres. So if you apply the maximum deviation that was observed in the set of results above of (15%) this is either 60.95m or 45.05m. Therefore, a user could potentially be driving within a dangerous distance (below the 53m stated length) but due to inaccuracy of the distance calculation could provide a reading that was thought to be safe. Please note that this is a very rudimental calculation only using one set of results from one experiment in a paper, taking the maximum deviation result but the idea stands that this technology has the potential to be highly inaccurate. However, taking the average deviation of 4.76% does lower the range to 55m – 50m which could be deemed as acceptable as a trigger range could be built into the application to warn at a lower level taking into consideration the inaccuracies. The experiment also stopped at the distance of 50m, the DVLA stated stopping distance goes up to 96 metres at 70mph. So the accuracy of anything beyond 50m is not known from this experiment. The experiment was also carried out using static imagery, with the mobile application a constant stream of video will have be used to calculate distance in real time. This leads onto other technologies that expands on stereoscopic imagery using video.

3.6.2 - VEHICLE DETECTION

There have been many advances in object detection in videos. Notably the work of Paul Viola and Michael Jones in 2001 in the conference paper: 'Rapid Object Detection using a Boosted Cascade of simple features'. This paper describes a machine learning approach for object detection, mainly for detecting faces from images. It expands from previous work by many publications leading up to the millennium. The paper describes the processes they went through to create a fast algorithm for face detection. They state that their functions are "reminiscent of Haar Basis functions" (Viola and Jones, 2001). Which derive from a Haar wavelet, a sequence that was proposed by Alfred Haar in 1909. (Haar, 1910). His work has been well used within the research and development of object detection, Viola and Jones refer to this as Haar Cascades and Haar features (Viola and Jones, 2001). Viola and Jones expanded on their work with multiple publications and has been used by many organisations developing object detection technology (OpenCV, no date) (MathWorks, no date) with google scholar indicating that their work has been cited over twenty-three thousand times (Google Scholar, no date)

Advancing from face detection the same techniques can be used to detect other types of objects. In this project vehicles need to be detected so calculations can be performed to estimate the distance. A document published by Anselm and Anton (2005) titled: 'A vehicle detection system based on Haar and Triangle features' discusses the processes on how to detect vehicles using a video camera. Expanding on Viola and Jones work to include Triangle filters to aid in more accurate detection. The paper is very technical with references to many complicated functions and algorithms implemented at a very low level to achieve this. They show that a camera mounted to a fixed point in a car pointing forwards. Knowing the height from the ground and the characteristics of the camera (angle, focal depth etc.) and the assumption that the road is flat, then a window can be identified with markers to estimate distance. They also state that in their experiments they assume that the vehicle width is between 1.5 and 2 meters, the known width is key to predicting the distance as calculations can be used to determine the distance by how wide the detected object is in pixels relative to the full image. See the illustration below:

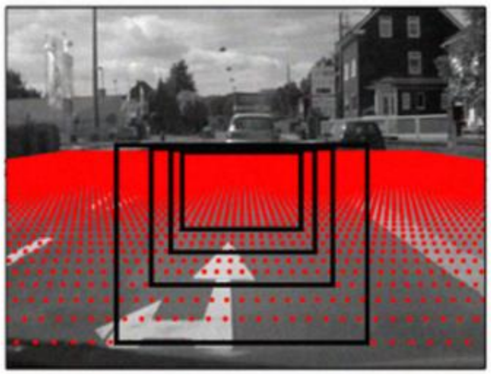


Figure 9 – Flat road assumption with markers placed on image (source: Anselm and Anton, 2005)

They later show some actual results of the system working. Detecting 2 cars within the picture and what distance they are away from the camera:

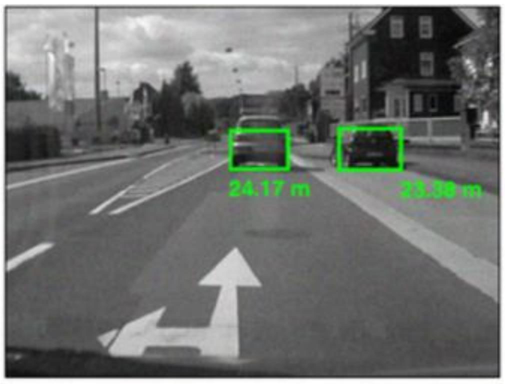


Figure 10 – Detection Results (source: Anselm and Anton, 2005)

This paper is a very low level technical document on the various functions, algorithms used to achieve this. It is also stated that vehicles can be detected up to 70 meters with the current resolutions they are using, but expand saying that accuracies can be increased with higher resolution video (Anselm and Anton, 2005) leading to maybe the possibility of being able to detect further distances and therefore being able to satisfy the top braking distance stated by the DVLA at 96 meters for detection. Written in 2005, video cameras were certainly not as high resolution compared to today in comparison with the price and size of consumer grade cameras. Which also leads to the possibility of being able to expand on their work and achieve a larger distance tracking rate with present technology.

3.6.2.1 - OPENCV

Undertaking more research around vehicle tracking technology led onto the discovery of a man named Marcos Nieto. He is a researcher in the Vicomtech-ik4 research alliance at the Department of Intelligent Transportation Systems and Industry (Nieto, 2011). He blogs about his work developing tracking systems and has many accepted publications in this field. He uses a technology called Open

Source Computer Vision Library (OpenCV), a library for vision applications enabling high level programming for image detection, incorporating many algorithms designed to be able to detect faces, identify objects, produce 3d point clouds from stereo images and many more (OpenCV, no date). Documentation provided by OpenCV expands on the work by Viola and Jones using their algorithms on Haar feature detection to provide a higher programming level approach to object detection, removing the low level technical aspects as discussed previously that used multiple mathematics algorithms and functions to achieve the detection.

Marcos has a YouTube channel showcasing his work. One of his videos shows real-time lane and vehicle tracking. Below is a screenshot taken at 0:22:

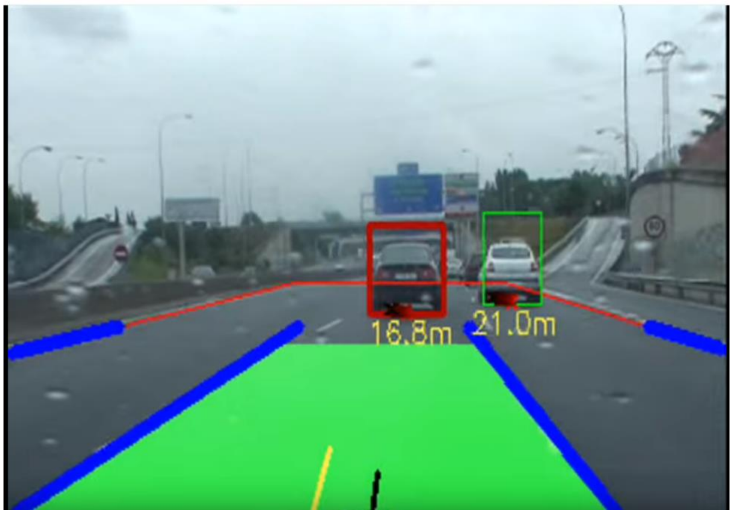


Figure 11 – Screenshot showing the work of Marcos Nieto with vehicle and lane tracking using open CV (source: Nieto, 2010)

In figure 11 it is shown that the software has detected 2 cars and is reporting the estimated distance. Note also that there is a green marker around the car on the right hand side, and a red marker around the car that is in the same lane as the driver. The coupling of the vehicle detection and lane tracking makes it possible to detect which vehicles are in the direct line of the driver, meaning that the system would know which vehicles it can discard in the calculation of whether the driver is driving too close, eliminating false readings. However, details of the distance limitations could not be found, leaving this as an unknown on how far in front his system could track. Linking back to the details of the paper written by Anselm and Anton (2005) where they conduct a similar project and state that detection and accuracy is increased with high resolution videos. Therefore an area to consider that it may be possible to achieve or exceed accurate measurements up to the DVLA stated 96 meters at 70 mph. Reading through OpenCV's documentation on video input, it does not specifically define if there is a maximum resolution supported by the library but the question was asked in the OpenCV community forum, with the top answer written by a reputable contributor stating that the video is not limited by size but the higher the resolution the more data that has to be processed in that frame (Puttemans, 2013). Leading to the problem of processing power required to do the calculations in real time.

OpenCV library is available on iOS, Windows Phone and Android which would make it perfect for a mobile application as there would be no requirement for external hardware and is not reliant on stereoscopic imagery so only one camera is needed. Nearly all smartphones have a camera and most are high resolution. However, on a mobile device processing power is limited. It is also noted that calculations are based around the height of which the camera is placed from the ground, angle of the camera pointing and the camera's specifications. This proposes a problem as the user would have to manually measure the height and input this into the application, although the angle could be calculated using the smartphone's in built accelerometers but not all smartphones have this feature, and cameras in smartphone's all have different specifications. So the application would either have to have a database of what specifications the camera has and automatically detect the make of phone and look this up, or reliance on the manual input of a user. Which presents a large problem as it would be hard for a user to find this out and if they input the incorrect details then the application would be significantly inaccurate in measurements or not work at all, completely failing its requirements as a system.

3.6.3 - ACTIVE

It was discussed earlier the details about measuring distance using stereoscopy, this was referred to passive as there was nothing that interacted with the environment. Active technology uses equipment that interacts with the environment, for example projection of light or lasers onto the environment that can then be measured in a way to determine the depth of the image. A device that uses active technology to measure depth is the Microsoft Xbox Kinect.

3.6.3.1 - MICROSOFT KINECT

The Microsoft Kinect devices uses a number of different pieces of hardware to capture video and depth. An article written by Zeng, 2012 explains and illustrates how the technology works: The Kinect has a standard RGB camera, an infrared camera and an Infrared projector. Using the Infrared projector, a structured dot pattern is projected into the environment, the infrared camera can see the infrared dots (see figure 13). As the projection and the camera are placed at a certain distance, as the dots are projected, the slight angle means that the further the depth the shape of the dots changes. The dot pattern is unique so the Kinect hardware can detect the unique patterns for that certain area and as the geometry between the infrared projector and camera is known, triangulation can be used to determine the depth of areas in the image. Combining this with the colour camera, a depth map can be calculated for the image (Zeng, 2012). See figure 12 for an illustrated image of the Kinect hardware

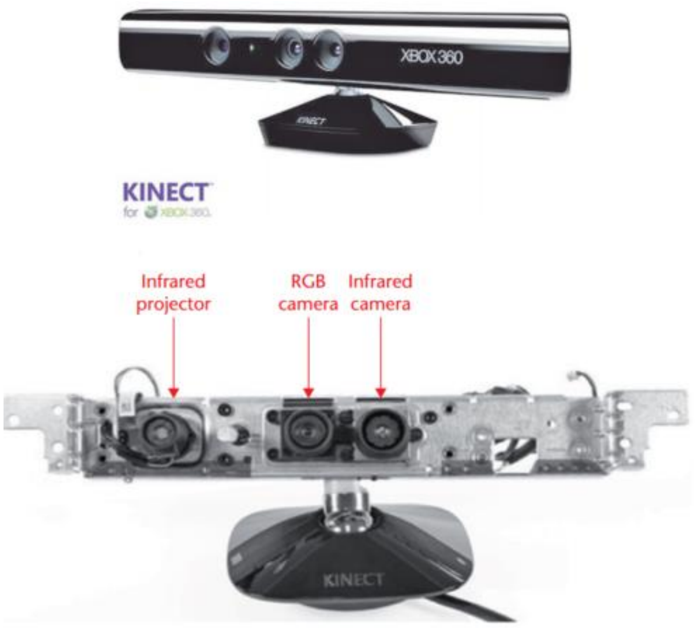


Figure 12 – Microsoft Kinect teardown showing the infrared projector, RGB camera and infrared camera (source: Zeng, 2012)

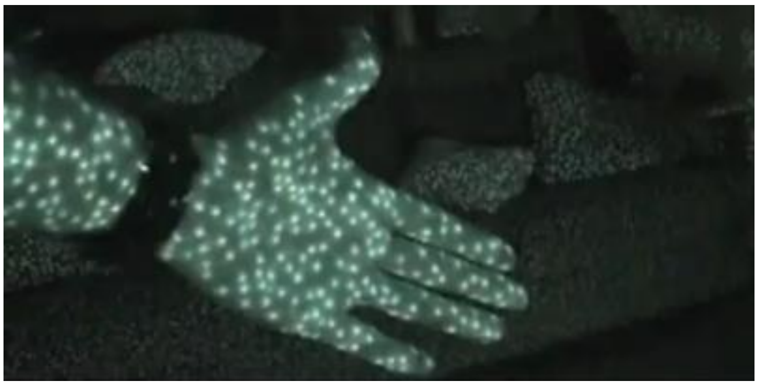


Figure 13 – Example of the structured light projected by a Microsoft Kinect device (source: Taylor, 2011)

Microsoft made available a fully packaged software development kit (SDK) for windows, allowing developers to create programs on a windows platform to utilise the Kinect hardware (Microsoft, no date). There are no official ways to directly interface a Kinect with a mobile device. However, research indicates that people have been able accomplish this with android using many hacks, workarounds and custom code (Cook, 2011), (Bauer, 2011). It is also looks rudimental and requires hardware to interface to the mobile device. A far from ideal solution.

The idea behind using this technology would be to mount it to a vehicle as such that it is facing outwards towards the road ahead. Then using the technology to measure the distance in real-time of objects in front, as the Kinect hardware can do this 30 frames per second (Zeng, 2012), real-time calculations would be very quick. However, there are many limitations with this technology: as it uses infrared light to project onto the environment accuracy is severely diminished under a high light environment. As intense light washes out the infrared projection and the Kinect cannot accurately measure depth. Microsoft themselves give out warnings that sunlight makes it hard to track movements (Microsoft, no date), rendering this impractical for daytime usage, in which most users would be driving in light conditions. Also using an infrared light projection limits how far the projection will travel. Similar to a low power torch shining into the distance there becomes a point where the light will not reach. The same principal applies to the Kinect technology, therefore Microsoft state that the maximum distance a Kinect can detect depth is up to 4 meters (Microsoft, no date), this is too short for the application in what is needed in this project. It does bare thought to whether this technology can be expanded and be applied to this project though. The possibility of higher powered infrared lighting to reach a greater distance could help, however this would require a complete overhaul of the technology and software used in the Kinect to achieve this, which is out of scope for this project.

3.6.3.2 - LASER RANGE FINDER

Another active distance measuring technology is the use of lasers. Lasers were invented in the late 1950's, and function by emitting photons (light particles) in a line to create a concentrated light beam (Woodford, 2015). With it being a concentrated beam, dependant on the power, lasers can travel long distances. Laser range finders measure distance by emitting a laser and measuring the time it takes for it to be reflected off the object and back to the receiver (Zant, 2013). Using the Time-of-Flight Principle, where a known speed of the signal being sent can be measured for its time to come back (TeraRanger, no date). Laser range finders are common place in many industries, construction use them to measure buildings and rooms. The military use them for measuring distances to places, and also used on weapons to aid in long distance combat.

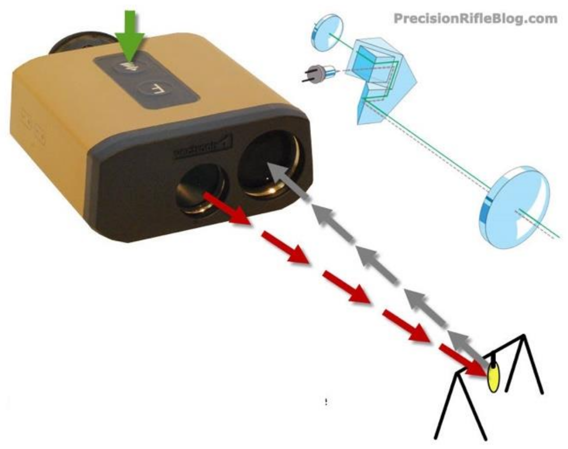


Figure 14 – Illustration of a laser range finder, red arrows indicate the emitted laser, grey arrows indicate the reflection back to the device (source: Zant, 2013)

There has been much advancement in laser technology, noted with the upcoming technology for autonomous cars. The google autonomous car uses a special laser range finder device to create a 3d map of its surroundings, the device, a Velodyne 64-beam laser, is mounted on the roof of the car and can map a 360-degree map of the environment (Guizzo, 2011). Consisting of 64 lasers on a spinning platform that can rotate up to 900rpm, creating a map of the environment up to 15 times a second (Velodyne, 2014). This data is used detect terrain and objects around the vehicle and compute how to drive the vehicle autonomously (Guizzo, 2011). The maximum distance this device can track is 120 meters (Velodyne, 2014), which is more than sufficient for the project needs of the DVLA documented stopping distance of 93 meters at 70mph.

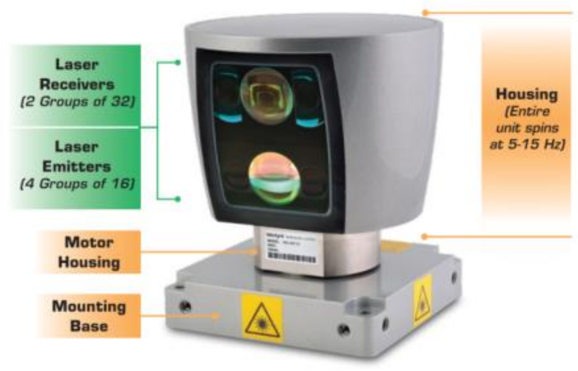

Laser Receivers (2 Groups of 32)
Laser Emitters (4 Groups of 16)
Motor Housing
Mounting Base
Housing (Entire unit spins at 5-15 Hz)

Figure 15 – Picture of the Velodyne HDL-64E (source: Velodyne, 2015)

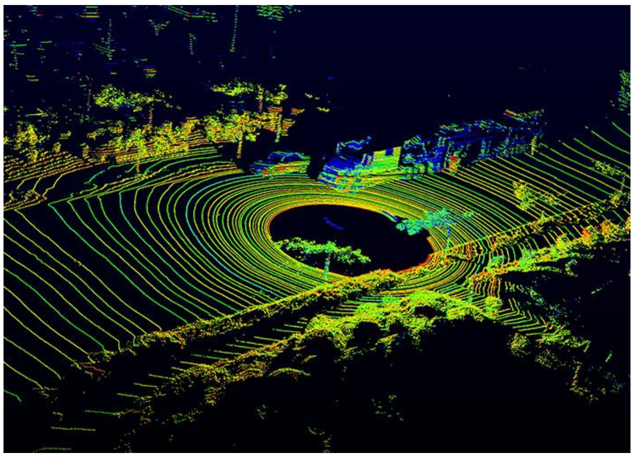


Figure 16 – example of raw data output from the HDL-64E (source: Velodyne, 2015)

This technology could be applied to this project to enable distance tracking for vehicles in front of the driver. This device produces a 360 degree 3D map of the surroundings of which for this project only a certain field of view (FOV) facing outwards from the front of the vehicle is the only concern. Similar to the video technology discussed earlier with OpenCV lane tracking technology would be needed to calculate which vehicles are actually a concern when driving and to eradicate false readings that could display when vehicles in other lanes pass by. Much work, complicated calculations and programming would be needed to turn the data into usable information in which to enable an application to determine the distance of objects in front of the driver. Also the price of this equipment is 70 thousand dollars (Deyle, 2009), which is out of scope for this project as the project is focused on delivering a technology using affordable consumer grade equipment.

3.6.3.2.1 - CONSUMER GRADE

Undergoing research on laser range finders uncovers a wide range of devices. Dependant on the rage required, the on board technology, the prices range considerably. From under £10 to over £500 (Amazon, no date). Many of these devices only allow range finding on-board the device with a display screen or offer recording capability for later review. This leads to the question of how can the data be made available to a mobile device? Researching many laser range finders brings to light that some manufacturers are equipping the devices with Bluetooth (Laser Technology Inc, no date) (Bosch, no date). With the ability to link to smartphones to utilise the laser data in applications. Notably these manufactures use their own proprietary applications for the smartphone to communicate with the device. Being able to get access to the raw data from the device to use it within the application of this project would be difficult, if not impossible. Either reverse engineering the device to determine the protocols available or by seeing if the manufacturer has any technical documentation available to tap into the Bluetooth stream. First of which is probably against the terms of the manufacturer of the device, secondly the manufacturer will highly likely not give out details of the workings as it is marketed for use with their own software.

Nearly all smartphones have Bluetooth so having an external device that can be linked over Bluetooth makes it for a very suitable approach. This leads onto the findings of a blog written by a man named Andrew Fuller, an engineer and a robotics hobbyist who is experimenting in remote-controlled robotics (Fuller, 2014).

In Fuller's blog post written in 2013 he explains how to get serial data out of a relatively cheap consumer grade laser range finder. He describes a device, the UT390B from Uni-T, having a debugging port on the device itself. The blog details the process of gaining access to the port on the device and the various pins that output the data. Using an Arduino device to read the incoming serial data. An Arduino is an open source electronics platform, with multiple connections available that can send and receive data to a number of electronics like lights, motors, LED's, LCD Screens and many more (Arduino, no date). It has its own Arduino programming language that is designed to not be difficult, easily understood and learnt (Arduino, no date). Arduino has become quite popular with electronics hobbyists building their own projects. The device can be connected to a PC using USB and can output information to the PC. However, programs can be written to the Arduino that can run in its own enclosed environment without the need for PC interaction.

Figure 17 – Uni-T Laser Range Finder

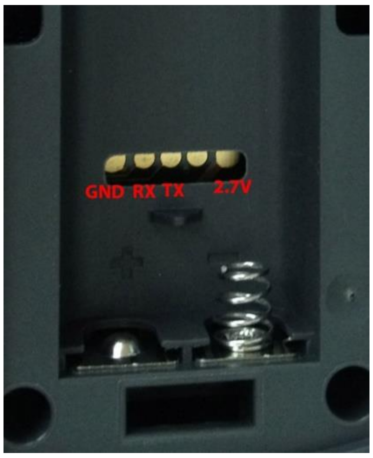


Figure 18 – Device with diagnostic port

Figure 19 – An example of the Arduino Uno board (source: Arduino.cc)

Fuller also has available the code he used on the Arduino to parse the incoming serial data from the device. This is as far as he takes his project, he shows the readings of the device being read into a PC or an LCD Screen connected to the Arduino device. This leads onto the question of being able to get this data to a smartphone over Bluetooth.

3.6.3.2.2 - ARDUINO

Arduino offers many add-on modules designed to work with the main board itself. One of these is a Bluetooth module. These can be attached to the board and offers a Bluetooth serial communication interface. Bluetooth serial communication has been part of the Bluetooth specification since development of version 1 emulating the standardised RS-232 serial interface (Bluetooth SIG, 1999), (TalTech, 2013). Having a standard serial interface opens communication possibilities for many devices - no special hardware or software is required to interpret signals. This is the main design motivation for Bluetooth, to enable multiple manufactures to design solutions that can communicate with each other in a standard way. Windows Phone and Android both support Bluetooth Serial Profile (SPP) for communication (Microsoft MSDN, no date), (Android Developers, no date). However, iOS does not support the SPP natively, but can be worked around using Bluetooth Low Energy 4.0 (BLE) which basically enables app specific communication profiles (Apple, 2015), or being part of Apple's MFI program, which means a special MFI chip is needed to use old communication profiles such as SPP (Apple Inc, no date). Adding further complexity, but plenty of opportunity to implement an application in either of iOS, Windows Phone or Android devices, these will be discussed later in the chapter for a suitable implementation environment.

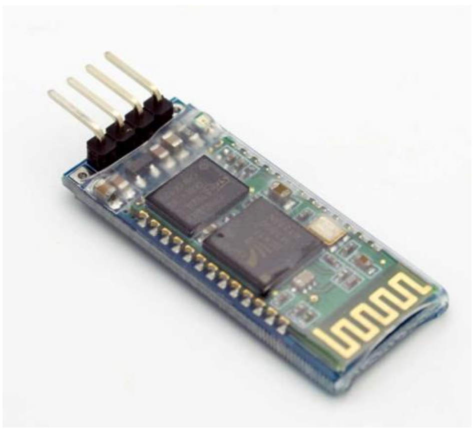


Figure 20 – Bluetooth module for Arduino (source: http://www.vetco.net)

3.6.4 - MOBILE DEVICE IMPLEMEMTATION

According to the International Data Corporation (IDC) below is the statistics on the current market share for smartphones

Period	Android	iOS	Windows Phone	BlackBerry OS	Others
2015Q2	82.8%	13.9%	2.6%	0.3%	0.4%

(IDC, no date)

Worldwide, Android is the clear leader with iOS following, then Windows phone and BlackBerry OS. As discussed earlier with the implementation of distance tracking technology using the open source library OpenCV and the camera on the smartphone, OpenCV is officially available for iOS, Android and Windows Phone but no mention of BlackBerry OS (OpenCV, no-date). This discounts using BlackBerry OS if implementation was using OpenCV.

3.6.4.1 - EXTERNAL CONNECTIVITY

If implementing using an external technology, like the Arduino discussed earlier. Communication is over Bluetooth using the SPP which has native support in Blackberry OS (BlackBerry, 2015), Windows Phone and Android. iOS is limited at this point. Natively iOS doesn't support SPP unless being part of Apple's MFI program which requires a separate authentication chip to be used with Bluetooth devices. Something which organisations and hardware manufactures can only apply for and incurs costs and extra complexity. However, a blog post by Owen Brown details a tutorial on how to communicate with an Arduino in iOS over Bluetooth using Bluetooth Low Energy 4.0 (BLE). Apple allows greater access using BLE compared to older Bluetooth technologies (Brown, 2014). This does result in a more specialised Bluetooth module being needed for the Arduino, at twice the cost of a standard module.

3.6.4.2 - PROGRAMMING LANGUAGE

3.6.4.2.1 - BLACKBERRY

Based on BlackBerry's most recent operating system, BlackBerry 10 offers a wide range of development languages to produce an application. Published on the BlackBerry developer site (BlackBerry, 2015). They offer higher level programming using HTML 5 and JavaScript to produce a web application that will run as a native app on the device. Also Adobe AIR, Adobe AIR is a runtime designed to package the same code into native applications for multiple devices like Windows, Mac, android, iOS (Adobe, no date). Applications can also be created natively, this is using C, C++ and QML languages. However, when searching through the Bluetooth documentation available for the API's, HTML and Adobe AIR only has access to simple file sharing profiles. The only documentation for getting access to the Serial Port Profile needed to communicate with an external device such as the Arduino is by using the Native development languages for BlackBerry OS (BlackBerry, 2014).

The integrated development environment (IDE) for blackberry OS is Momentics IDE. It is marketed similar to Eclipse which has been a popular Java IDE since 2001 (The Eclipse Foundation, no date). However, as it is C and C++, there is also a plugin for Microsoft Visual Studio IDE to develop for blackberry (BlackBerry, no-date). Visual Studio is an extensible IDE with a long list of features that can be used to develop Microsoft Windows programmes, Windows Phone and Websites, with a very large user base and active community of development. However only available on a windows platform.

From the research undertaken, evidence shows that the blackberry community is very small comparatively to iOS and android development, API documentation also looks minimal. This correlates to the market share statistics shown earlier as such a small market share means less developers and communities. This poses an issue in development, many developers rely on good documentation and communities to develop good software.

3.6.4.2.2 - ANDROID

Natively Android development is done in Java and XML (Sims, 2014). However, like Blackberry OS, apps can be made using HTML and JavaScript with 3rd party tools like PhoneGap, which is an open source framework for building cross platform applications, combining native API's translated to HTML and JavaScript which one can develop once and work cross platform (PhoneGap, no date). Again, these higher level style of programming for the devices do not include support for Bluetooth SPP, but with some research does reveal an open source project that is a plugin for PhoneGap to enable Bluetooth serial communication on iOS, Android and Windows Phone (Coleman, 2015).

To develop for Android, up until recently the Java IDE Eclipse was used with android plugins (Android, no date). Eclipse being a very well known and used IDE in the community. The modern way of Android development has changed to Android's own custom IDE called Android Studio. This is based on the IntelliJ IDEA IDE, another well known and feature rich Java IDE (Android, no date) (JetBrains INC, no date). Android studio has been developed to give a better and more stable environment to develop Android apps, which when researching Eclipse and android surfaces issues with development which lead many communities to disliking Eclipse (Rajput, 2015). Both Eclipse and Android Studio is available on Windows and Mac platforms.

Android provides extensive documentation on API's available, tutorials to help development and has a strong community on many forums.

3.6.4.2.3 - IOS

Primarily iOS applications are developed using Objective-C but in 2014 Apple launched a new programming language called Swift. Which is designed to be easier to read and develop with (Apple Inc, 2015) Swift can work along side Objective-C in development.

Development for iOS can only be done in Apple's IDE Xcode, a feature rich IDE with many features to aid development. Which is only available on a Mac platform (Apple Inc, no date).

iOS development also has a very strong community and extensive documentation provided by Apple.

3.6.4.2.4 - WINDOWS PHONE

Windows phone development is done in C# for programming and XAML for layout (Microsoft MSDN, 2014).

To develop for Windows Phone, the Visual Studio IDE must be used which is only available on a windows platform.

There is good API documentation provided by Microsoft and a large community of developers from what has been observed during research.

3.6.5 - CONCLUSION

Researching software and hardware solutions to implement in this project has revealed many solutions available. It is noted that the most ideal solution would be contained within the single device as the smartphone itself. An application that can use the existing phone hardware to measure distance. From researching and discussing the OpenCV library, with the examples shown of other peoples work on vehicle detection using video, this looks like a viable solution to be able to accomplish the needs of the project. However, much time and technical skill is needed to successfully achieve this. The use of a video stream and OpenCV with lane detection technology requires extensive programming and testing and also looks complex. In addition, the OpenCV library is completely new to the project executioner and would require extensive time learning and practising the code to implement in this project. It may be possible but time constraints limit the amount of learning and execution time available to implement a solution. Furthermore, the technology relies on known characteristics like height, angle and the camera specification of the mobile device, in a world where there are thousands of mobile devices all with different camera specifications and relying on the user to input correct measurements concludes that OpenCV is not the correct implementation for this project.

An alternative to having the solution completely contained on the user's device itself would be to have an external device fitted to the car coupled with an application on the user's phone with a connection between the two devices. This has multiple benefits: one benefit is that there is no reliance on knowing characteristics of the user's smartphone height, angle and camera etc. This then removes the uncertainties and thus makes for a more accurate solution. The idea behind the solution would be to have the external device mounted to the car, requiring no extra user interaction once fitted, fit once and leave. This sounds complicated for a user to undergo themselves but if the solution was to be implemented in the real market, professional fitters would be used to make sure that the device is fitted accurately. Like companies already do now with the telematics solutions available for car insurance, the company fits a monitoring box into the user's vehicle to monitor driving aspects. Making it a very suitable and viable solution as very similar procedures already exist in the market today. This would leave only the user to connect to the device with their smartphone once they enter the vehicle.

Another benefit of separating the solution into two devices is that the external distance measurement tool can be made like an interface, with a standard communication protocol being used, separating the concern of compatibility with the smartphone device. This gives the ability for the smartphone solution to be implemented on any platform, it is already noted that Android, Windows Phone and iOS have abilities for Bluetooth serial communication so the external device could be used with either platform, expanding on market availability and user reach.

From the research gathered it is clear that there are some highly effective range finding tools available. The Xbox Kinect looks like a very good solution for detecting depth, it is not focused in 1 line of sight and allows for a wide view of the scenery that can measure distance on multiple points, however limitations of this device in that it can only detect up to 4 metres and reduced performance in light environments and the addition of the issue of interfacing wirelessly means that this is not a viable solution unless the technology was expanded using more powerful projections and sensors. Of which is out of scope for this project.

In contrast to measuring depth using a projection and a camera like the Xbox Kinect is using a different approach using lasers. Lasers are more powerful and thus more accurately measured. Notably the Velodyne 64-beam laser discussed earlier looks like an outstanding approach to map the surroundings of a vehicle and to be able to detect distances up to 120 meters, exceeding the requirements of the project in relation to the DVLA stopping distances. However, the unit costs 70 thousand dollars which is not ideal for this project.

Looking for a more consumer grade application revealed the use of the handheld laser distance measurement tool the UNI-T UT390B coupled with the use of an Arduino device to interpret the serial output of the laser device. The solution would be to have the laser device mounted to a vehicle pointing outwards from the front, therefore pointing straight towards the potential vehicle in front. With the Arduino device interpreting the serial data from the laser device and relaying that over Bluetooth to the smartphone where the application would read the data. However, this does have limitations. The UNI-T has a range limit up to 45m, far from the DVLA maximum distance of 96 meters required to stop at 70mph. Also a laser beam is straight, which means that the device is only accurate and functional when the vehicle is in a straight line of sight of the vehicle in front (See figure 14 and 15)

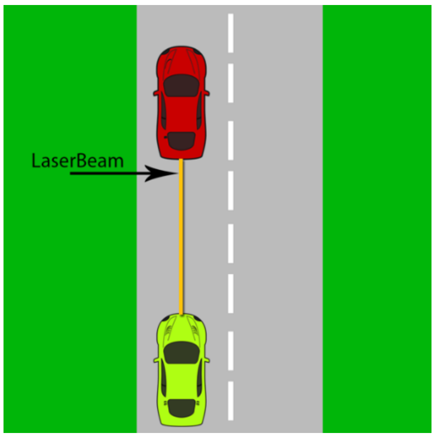


Figure 21 – Illustration of Laser beam with direct line of sight

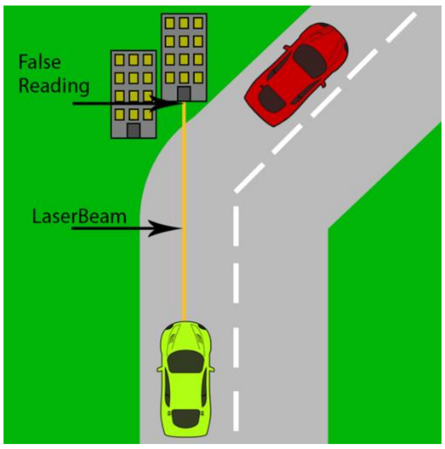


Figure 22 – Illustration of the effects of a corner and producing false readings.

However, the corner issue is something that could be mitigated by taking averages to predict valid values over a period of time or only taking values when the car is straight, which could be achieved via GPS technology on the smartphone where a prediction of a straight road can initiate a reading, or use of accelerometer data to determine if a vehicle is going around a corner. Of which these mitigations need to be researched, designed and tested in future stages of the project. As for the maximum distance limitation, this is device specific. Many other devices on the market offer higher ranges, but also cost for the devices are much higher. Due to monetary limitations for this project, it is acceptable to use the UNI-T device as a means to achieve a proof of concept. Further work can be undertaken after the project is succeeded to expand on the technology available.

In conclusion, the best option is to use the UNI-T Laser range finder device coupled with an Arduino Bluetooth device, to enable real time reliable distance measurements that can be used in a smartphone application.

With the hardware implementation justified the software implementation needs to be considered. Due to time constraints and limitations of this project, for proof of concept, only one mobile device platform will be used to implement the application. To support the decision on the chosen platform to develop the application a decision matrix:

The following is a weighted decision matrix. There are weightings on what are the more important aspects of deciding on the programming environment. The IDE, Current Knowledge and the Bluetooth API's are considered to have more weighting as the IDE plays an important role in development, the functionality of code completion, code analysis and ease of use can determine how well an application is developed. Current knowledge also seen as a higher weighting because if the project executioner does not have any current knowledge then they would have to start from the basics to learn how to implement aspects of development, thus increasing development time and possible compromise on quality. Lastly, the device has to connect to the Arduino via Bluetooth then the availability and extendibility of the Bluetooth API's are weighted higher.

Scores range from -5 to 5

Table 1 - Mobile Device Decision Matrix

Decision Factors		**Android**	**iOS**	**Blackberry**	**Windows Phone**
Criteria	**Wt.**	**1**	**2**	**3**	**4**
IDE	2.0	3	2	2	3
Current Knowledge	2.0	1	-5	-5	4
Bluetooth API's	2.0	5	0	4	4
UI Design	1.0	2	1	2	3
Ease of Learning	1.0	2	2	2	3

Criteria	**Definition**
IDE	How good is the Interactive Development environment?
Current Knowledge	How knowledgeable of the language. Any previous experience?
Bluetooth API's	How extensive are the Bluetooth API's on the platform? Based on reviewing current API documentation for Bluetooth connectivity and other sources, which refer to the use of Bluetooth connectivity with Arduino devices.
UI Design	How easy is it to build UI on the platform?
Ease of Learning	How quickly can the language be learnt and understood?

Speed of Development	1.0	2	2	2	3
Support	1.0	4	4	0	1
Weighted Scores		28.0	3.0	8.0	**32.0**

Speed of Development	How quickly can you develop working prototypes?
Support	How good is the support community at showing knowledgeable examples of code to help build a solution? Based on searching community websites like stack overflow for level of public interaction.

Having reviewed the numbers, it shows the winner of the decision matrix as Windows Phone. Having previous experience with C# and the Visual Studio IDE gives excellent benefit for understanding code and increasing speed of development, time of which is a significant constraint in this project. Researching Bluetooth API's and finding community answers on how to implement an Arduino with a Windows Phone application shows that it is significantly easier to do so compared to iOS, however android scored better on this section. Correlating to the market share statistics, research and observations shows that the Windows Development community is significantly smaller compared to Android and iOS, but the other sections scores and weighting compensates for this and concludes that Windows Phone is the choses implementation environment.

To conclude the chosen implementation for this project will be two devices: firstly, a UNI-T UT390B attached to an Arduino via the serial output. The Arduino will process the serial data and make the data available on a Bluetooth interface. Secondly, a Windows Phone with an application that will connect over Bluetooth to the Arduino. The application will process the data stream and react accordingly.

Figure 23 illustrates the chosen 2-tier implementation for this project:

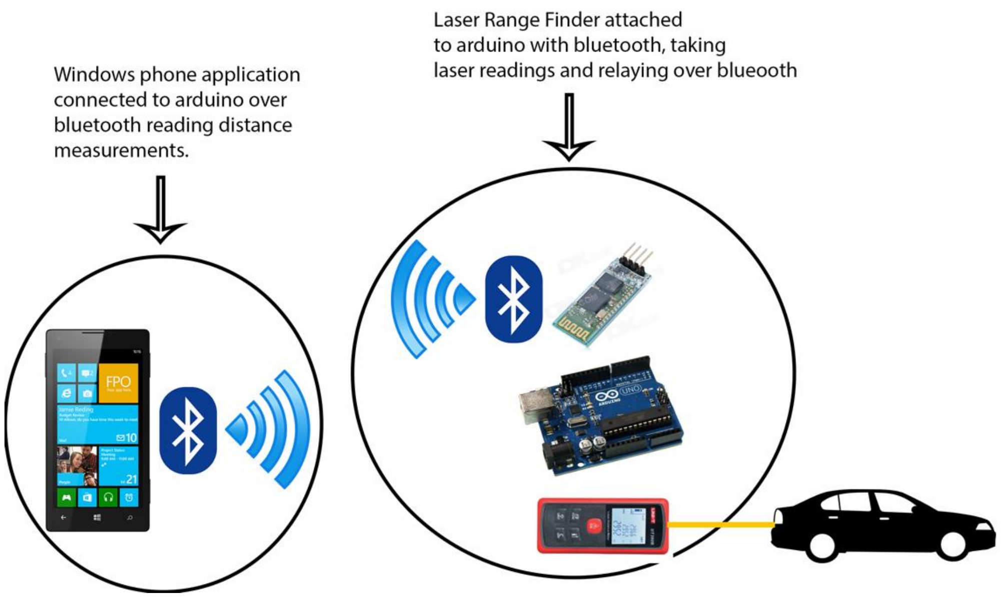

Windows phone application connected to arduino over bluetooth reading distance measurements.
Laser Range Finder attached to arduino with bluetooth, taking laser readings and relaying over blueooth

Figure 23 – Diagram illustrating the two chosen implementations.

3.7 – TESTING STRATEGY

The following section will highlight testing techniques and strategies that can be used in execution of this project. Also review details of the testing guidelines proposed by the DSDM Methodology. Any descriptions; key characteristics and pros and cons will be discussed where relevant.

To conclude, an overall testing strategy will be designed to support the project.

3.7.1 - DSDM & TESTING

The chosen methodology to follow for this project is a hybrid approach, following most characteristics from DSDM. The DSDM Consortium defines in chapter 23 of the 2008 DSDM Atern handbook a detailed description of guidelines on testing while implementing the methodology. One of the key principles is to 'Fail Fast', the idea that the earlier a defect is found and fixed the less it costs (DSDM Consortium, no date). With this principle in mind they lay out the following guidelines:

- Collaborative testing – collaborate with stakeholders while testing to increase productivity.
- Repeatable Tests – as DSDM is based on iterative development tests need to be repeatable so that they can be run as required at each iteration to ensure the system is still working.
- End-to-End experience – demonstrations, modelling and testing user experience from early stages will allow changes to maximize the end to end experience of the product.

- Independent testing – tested by someone other than its creator. This ensures that understanding of a requirement is tested. Active involvement of the business helps with ensuring an independent perspective is applied.

- Prioritised tests – prioritise tests because it may not be possible to test everything delivered in a development Timebox. Each test prioritization will use MoSCoW rules. Tests should be aligned to a requirement which in turn has MoSCoW rules.

- Test-Driven development – tests are designed before the product is created, ensuring acceptance criteria is confirmed before effort is wasted creating something incorrect.

- Risk-Based Testing – emphasis on testing high risk areas compared to lower risk items of low value if they do not work. Requirements should be risk assessed against the impact and probability of a failure, this will give prioritisation to the tests that should be run.

(DSDM Consortium, no date)

As this project is undergone solely then collaborative and independent testing and becomes irrelevant, however it is something to have in mind. It is common for developers to be single minded when developing a system resulting in functionality that might be deemed correct by the person who developed it but does not satisfy the business case. This is why multiple users are involved in testing to get a better overall idea of the success of the system. In this case of the project being executed solely, one must be mindful of testing objectively in the perspective of the relevant user to gain a better more accurate understanding if the system tests are successful.

3.7.2 - TESTING TECHNIQUES

In software development development there are many techniques used to test different aspects of the product being developed. Many of which are combined into an overall testing strategy to accommodate all the needs of testing the product.

Taken from Software Testing Help (2015), multiple testing types for software testing are outlined:

Table 2 - Testing Types (Software Testing Help, 2015)

Type	Description
Black box testing	Testing of the system as a whole without internal design considered, based on requirements and functionality

White box testing	Internal logic of the system tested. Based on coverage of code, branches, paths conditions.
Unit testing	Individual components of the software, usually conducted by the developer as detailed knowledge of the internal coded required.
Incremental integration testing	Continuous testing of the application as new functionality is added, bottom up approach.
Integration testing	Testing of integrated modules combined functionality. Can be code modules, individual applications, or client and server applications.
Functional testing	Black box type testing against functional requirements defined for the system
System testing	Whole system tested as per overall requirements.
End-to-end testing	Mimics real world usage, testing like a user
Usability testing	UI, application flow, navigation tested

Another technique to use in the testing process is the idea of full traceability. This is where each business requirement is mapped to a functional requirement which is mapped to test cases and results. Showing full traceability of the tests back to the business requirements. This is often documented in a Traceability matrix, a spreadsheet that clearly maps the requirements down to the tests (Software Testing Help, 2015).

However, Hazrati (2008) states interesting points questioning the need for a traceability matrix. He discusses it evoking a strong response from agile communities. Citing many credible articles and well known people in the industry. In one example he cites a point from a Scrum development group saying that the particular projected that was undertaken, the suitability of a traceability tool was completed concluding that the human labour cost was staggeringly high that there was no benefit. Another particular extract: "Traceability matrix would only be required when there is a danger of losing corporate memory that is genuinely valuable. If that is not the case then traceability is available in many forms like conversations, stories, strategic themes, histories, logs, journals, source code, automated tests, design documents, daily scrums, email etc." Here he is stating that traceability matrix is only of value if knowledge was possible to be lost, saying that there are many other forms of traceability that can be used. The following sections of this chapter will discuss the different techniques to be used, documenting testing of the artefact which will demonstrate where traceability can be obtained. In addition, the project is solely executed, knowledge is not in danger of being lost, compared to big team projects where people may leave.

UI testing can be a grey area with personal opinion on what looks aesthetic and intuitive related to a particular user. However, smartphone companies: Apple, Android, Microsoft and Blackberry set out different guidelines to what the application should look like for their platform. Guidelines include, layout, colours, fonts, navigation styles etc. With certain guidelines to follow, apps for the platforms have a house style feel to them, although functionality is fundamentally different for different apps, navigation style, headings, fonts and colours are very similar. Allowing users of the platform to become familiar with gestures, positioning of elements and overall style, giving the user a better experience as they know what and where to expect elements. Naturally, adhering to design standards yields more accurate less error prone UI that is intuitive and easy to use. However, the different elements need to be tested so that they function as intended. For example, certain elements hidden or changed dependant on certain conditions. There are 2 ways this can be done: manually by running the application and testing the UI physically; or automatic testing, where test scripts can be written that will execute UI inputs programmatically like a user would be interacting with the system then output a set of results indicating pass and failures of the tests.

3.7.3 - PROPOSED TESTING STRATEGY

The following is the testing strategy that will be undertaken during this project. It will follow the guidelines set out by the DSDM methodology as close as possible. For reasons outlined previous, collaborative and independent testing cannot be undertaken, but the viewpoint of the target users will be a conscious thought throughout so that testing is as fair and unbiased as possible.

3.7.3.1 - REQUIREMENTS TESTING

The most important testing that can quantify the success or failure of the project. As an agile process is being followed, and from the research found on the suitability of traceability matrix it is concluded that a traceability matrix will not be used in this project. Resources are under constraint for this project and there are other testing techniques going to be used that can provide a good level of traceability without the need for extra labour to maintain such document.

However, requirements still need to be tested to make sure the project delivers to specification. For this a document will be produced and completed after the project is deemed complete. Listing all the requirements set out for the project. Documenting the tests undertaken and the results.

See Appendix C.1 for the requirements test template

3.7.3.2 - TIMEBOX TESTING

DSDM is an iterative development methodology. Using Timeboxes splits the workload into manageable chunks of work that will be delivered within a set time frame. A document will be used for each Timebox to outline the required functionality to be achieved, this will record if the Timebox session is successful and outline if there are issues with the deliverables of the Timebox that need to be rectified.

A Timebox might be used to gain some preliminary functionality as a proof of concept or a base that will later be adapted in other increments to satisfy an overall requirement of the project. So testing of Timebox requirements may form the bases of requirements testing later. It will also aid in visualising the success of the project as the iterations are executed.

See Appendix C.2 for the template for a Timebox test.

3.7.3.3 - INTEGRATION TESTING

The project consists of delivering 2 artefacts that work together as one application: The external hardware item and the mobile application for interaction with the user. There are going to be many different modules that need integration testing. Some functionality of both artefacts can be tested independently from each other but the system as a whole will be integrated tested to ensure the programming modules within the mobile application itself work together but also the communication between the external device is tested.

See Appendix C.4 for the integration testing template.

3.7.3.4 - UI TESTING

As the chosen platform is Microsoft Windows Mobile. Each screen of the application will be evaluated for adhering to the UI standards set out by Microsoft.

Similar to unit testing Microsoft offer a UI testing solution for UI interaction, allowing the developer to assert outcomes of different UI elements, navigation and data when actions are applied (Microsoft MSDN, 2013).

Through each iterative development stage, UI test scripts will be written where a test case is deemed appropriate to allow for automation of testing of UI elements. This will provide a good strategy as throughout the development iterations the tests can be run to make sure elements of the system have not broken in other places where functionality may have been changed. When the system is complete, a full suite of tests will have been completed that tests the UI extensively. Problems may be encountered where a UI test might not be possible to conduct programmatically, if this happens a manual test will be conducted and documented.

3.7.3.5 - END-TO-END TESTING

End to end testing involves testing the program in conditions as close to how a real world user would use the system. There is a limited amount of testing that can be carried out programmatically and to guarantee the system is free of defects, the system must be end-to-end tested to ensure the high level of the system meets the requirements. It is difficult to write end-to-end test cases that can be executed pragmatically. Unit tests, integration, and automated UI testing may be possible to get as close to end-to-end testing as possible, but some issues only arise when undergoing real world testing.

To end-to-end test the system, the external device and mobile application will be taken into a real world environment and used in its intended way. A testing template document will be available to record any anomalies that are observed and steps taken to reproduce the issue. This will give a list of all bugs found and ways to reproduce the problem that can be looked into being fixed.

See Appendix C.3 for the bug reporting template.

3.7.3.6 - UNIT TESTING

Unit testing is to be conducted throughout the project. As DSDM states to use Test Driven Development where you right tests first and develop functionality to pass the tests. Unit tests are a component of this. In addition, this type of testing brings into the KISS principle of development. Keep It Simple Stupid. Meaning breaking down code into simple sub tasks that can be individually tested, make sure methods are small and only solve one problem not multiple use cases. Resulting in maintainable, easy to read code that will be successful in delivering a working artefact (Hanik, no date).

As the project consists of 2 separate devices each with their own development language and environment a separate strategy is needed to test both aspects.

3.7.3.6.1 - MOBILE APPLICATION

During the development of the mobile application, test driven development will be adhered to. The chosen software implementation environment: Windows Phone developed in Visual Studio, has built in functionality to aid in test driven development. The suite allows the developer to create a test project and access specific libraries that can be used to test methods and class behaviour. For methods and classes, tests will be written to assert the required functionality, then development will be undertaken to satisfy the tests. This gives the advantage of being able to test all aspects of code, tests can be re-run at any time to make sure that other changes during development that break other parts are picked up quickly and rectified.

3.7.3.6.2 - ARDUINO DEVICE

Development in Arduino is more simplified, only a limited set of C and C++ instructions are available on the platform, many libraries are emitted due to processing and memory limitations, some of which are considered out of scope for what an average Arduino user will need (Westfw, 2014). In addition, the Arduino software does not provide any unit testing libraries or ability to create unit tests with verbose GUIs which are common in other languages. Arduino is considered a trial and error development. For this reason, Unit testing will not be used but the higher level testing documentation will be used for testing, documentation and confirming functionality as discussed earlier.

CHAPTER 4 – ANALYSIS AND DESIGN

4.1 – ANALYIS AND DESIGN

Research has been completed and decisions made on the hardware and software to use for implementing the project. This section consists of analysis and design for the proposed system to be implemented.

For this project, the use of the UML tools along with other diagrams and descriptions have been used to document the analysis and design of the system. UML is considered an object-orientated tool which consists of behaviour and structure diagrams: Behaviour modelling demonstrates dynamic behaviour that changes over time; Structure modelling shows the static structure and relations with other parts (uml-diagrams.org).

Modelling the system is conceptual. Taken from current knowledge and research means that modelling is what would logically and conceptually see the system to be. Therefore, after development has finished the resulting system architecture could be different to what was already modelled. However, modelling the system before development helps scaffold the system and is used along side the requirements capture process to fully realise all requirements for the system.

For the UML modelling 2 tools was chosen, the use of an online UML diagram tool called Creatley and software called ArgoUML. These was chosen as they offered a comprehensive set of tools to produce the diagrams required. ArgoUML is UML 1.4 compliant (CollabNet, 2001) and both tools are free, compared to other tools available that require payment, no extra cost is incurred on the project.

For this project, a detailed use case diagram with supporting descriptions and activity diagrams have been produced as documentation for the behaviour of the system. Use cases describe a set of actions the system performs in collaboration with an actor (user). The activity diagrams illustrate in greater detail than the use cases the processes that the system undertakes at a lower level.

See appendix D for Use Case Diagram and Descriptions, appendix E for Activity Diagrams.

A class diagram and component diagram have been produced to document the structure of the system. The component diagram is used to conceptually visualise the different components of the system. As there will be 2 separate products being developed, each with its own internal logical components in the software, a component diagram shows dependencies, interfaces and communication with each component. The class diagram represents the conceptual classes that will exist in the Windows Phone application. The attributes, methods relationships, associations etc. required for the system.

Throughout the modelling process, all diagrams, pseudo code, UI diagrams and the requirements capture for the system was all undertaken in parallel. Modelling, UI design and requirements capture is all complimentary of each other while the process takes place.

See appendix F for the application Class Diagram, appendix G for the entire system component diagram.

4.2 -HARDWARE DESIGN

It has been chosen to use an Arduino micro controller with a Bluetooth module and a UNi-T UT390B Laser range finder. From the research already gathered from the blog post by Fuller (2013) where he uses an Arduino and the UNI-T laser range finder to parse distance readings and output them to a LCD screen. His work will be used during this design process and then expanded to achieve the relevant data that can be sent over the Bluetooth module.

The Laser range finder outputs data over a serial connection and the Bluetooth module sends and receives data over a serial connection also. In this case 2 serial connections are needed on the Arduino device. There are many Arduino boards available that offer more or less features dependent on the application. In this design the Arduino UNO will be used. The UNO offers 14 digital input-output pins but only 1 hardware serial connection, which allows for serial data to be processed through a separate on-board chip. Allowing for other tasks to be completed on the Arduino's main processor (Arduino, 2016). However, Arduino offer a built in library called SoftwareSerial, that will allow the other digital pins to be emulated as a hardware serial connection. In this design, the hardware serial pins will be used to connect to the laser range finder, and 2 other digital pins will be used for a software serial connection to the Bluetooth module.

4.2.1 – ARDUINO PHYSICAL DESIGN

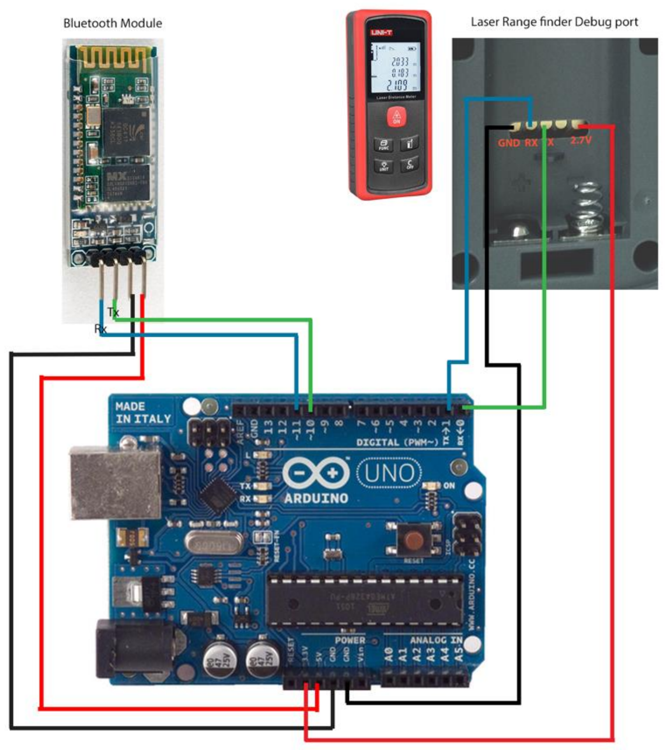


Figure 24 – Diagram showing Arduino, Bluetooth and laser range finder connection design.

Figure 24 is a diagram illustrating the cable connection design for the hardware. The laser range finder is powered by 2.7v as indicated by Fuller on his blog post. The Arduino board has 2 power supplies, 3.3v and 5v. It is shown that the laser device will work with a supplied 3.3v. Therefore, the 3.3v is connected to the power pin of the laser range finder, with the ground connecting to the ground pin of the Arduino. The laser Transmission connection goes to the Receive pin of the hardware serial on the Arduino, and the receive pin on the laser range finder is connected to the transmission pin of the Arduino.

The Bluetooth module is branded as a HC-06 device. This is powered from 3v - 6v, therefore the power pin will connect to the 5v pin on the Arduino as it is free. Ground pin to the Arduino ground pin. The Transmit pin on the module will go to pin 10 on the Arduino, and the receive pin going to pin 11 on the Arduino.

4.2.2 – ARDUINO SOFTWARE DESIGN

The Arduino software is developed in the Arduino IDE where the language C is used. The term sketch is used when developing for Arduino. This term is used to identify a single program that is uploaded to the device to be run. The concept of an Arduino program has 2 parts, a setup and a loop. These 2 methods are required in every sketch otherwise the IDE will not allow the sketch to be uploaded to the device (Arduino. 2016). The setup method is executed once when the device is powered on, and the loop method is a constant loop that is run on the device. It is noted that there is no concept of event driven programming, all processes are carried out within a loop. Therefore, carful design consideration need to be considered. As it is C language library classes that are written in C can be imported and used, this is how the supplied library's work.

As the UNI-T laser range finder device has been used by some hobbyists online, someone by the username of 'erniejunior' created a library for use with the Arduino and it is on Git Hub for anyone for free usage. The library abstracts the low level code needed for sending and receiving to the laser device into a class that can be imported into the sketch and then simple methods can be called to trigger a measurement from the device and return the reading.

In addition to the range finder library previously mentioned the SoftwareSerial library that comes shipped with the Arduino IDE will be used for communicating to the Bluetooth module. This will enable the use of sending serial data to the Bluetooth module via 2 digital pins that are not the hardware serial interface. The Bluetooth module itself doesn't need any additional software. The module itself is an abstraction from the Arduino. All the module is design to do is send and receive serial data from the pins it has. When the module is given power it will automatically become visible to other Bluetooth devices that can connect to them, it is the device that is connecting responsibility to create the serial port Bluetooth connection. The module will just relay data that it is sent on the pins to the connected Bluetooth device. This means that no further software design is needed for the Bluetooth module other than setting up the SoftwareSerial input and output.

4.2.2.1 - PSEUDO CODE

The pseudo code for the Arduino sketch as follows:

```
#include <UT390B.h> -- Include the library for the Laser measure device
#include <SoftwareSerial.h> -- Include the software serial library

UT390B laser(HardwareSerial); -- create instance of Laser measurement class
SoftwareSerial BT(10, 11); -- create instance of the software serial class

Void setup() – executed once when Arduino is booted
{
        SoftwareSerial.begin(9600); -- setup serial communication at 9600 rate
        HardwareSerial.begin(9600); -- setup serial communication at 9600 rate
}

void loop()
{
        IF SoftwareSerial HAS Value – If the software serial has incoming data, e.g. the phone device
has sent data to the Bluetooth module
                THEN
                READ VALUE into Variable INPUT – Read the value into a variable
                IF INPUT = “Measure” – if the variable = Measure then the other device has
requested a measurement
                        THEN
                        SEND Measurement Command to laser – send measurement request to
laser
                        WAIT Until Measurement is Returned – wait until value is returned
                        READ Measurement into Variable OUTPUT – Read measurement into variable
                        SEND OUTPUT to SoftwareSerial – Send value back to software serial interface, which will send over Bluetooth
}
```

4.2.3 - ANALYSIS ALIGNMENT

The design of the Sketch aligns to use cases from the analysis stage. The use cases state that the mobile phone application will poll the Bluetooth device for distance data (see use cases: Poll for Distance, Receive Poll Request). Then wait to receive data back over Bluetooth. The design of the Arduino sketch will be that the Software Serial (Bluetooth module) will constantly check for any incoming messages. If a message is received and matches the string “Measure” then the Arduino will then Poll the laser range finder over the hardware serial interface to initiated a measurement, it will then wait for the measurement to return from the range finder and return the value back over the Software Serial which the Bluetooth module will relay back to the phone.

4.3 - UI DESIGN

Design considerations of the UI are important as this can determine if the user will like the application. If they don’t then they are unlikely to want the application or use it. The main issue around UI design is Usability. Defined as 5 quality components:

- Learnability – How easy it is to accomplish tasks the first time they encounter the design
- Efficiency – once users have learnt, how quickly can tasks be performed
- Memorability – when users return to the design after a period of inactivity, how easy can they re-establish proficiency
- Errors – how many errors do users make, how easy is recovering from errors
- Satisfaction – how pleasant is the design

(Nielsen, 2012)

The application is going to be developed as a Windows Phone 8.1 application and the user interface for Windows Phone is relatively strict. Microsoft design all the built in UI elements to their standard. Using XAML language to scaffold UI elements on a particular screen. They all have a simple look and feel that is similar and recognizable across different elements. Microsoft also supply UI guidelines that advise on layout, commands, positioning etc. https://developer.microsoft.com/en-us/windows/design/layout

Previous experience working with XAML for a Windows Phone UI shows that if you use the controls as they are intended without modification to the visual aspects then they will naturally adhere to the guidelines that Microsoft publish. However, being conscious of layout and positioning is needed to be fully compliant. By using the available UI elements, it not only conforms to the guidelines set by Microsoft but also instils familiarity to its users, further satisfying the usability standard.

The overall UI appearance for the application will be based on the XAML element called a Hub. This used to be called a panorama control in versions prior to Windows Phone 8.1. The Hub control makes the UI one continuous horizontal panorama, with multiple sections with different elements on. This has been chosen because it allows all the important elements of the UI to be in one continuous screen, with navigation made easy with a single finger swipe left or right to the next sections. For an application that is going to be used to monitor driving, most likely the user will have their device mounted on the window or at some accessible fixed point in the vehicle. All the important UI aspects, like the starting and stopping of the monitoring, the monitoring screen where it shows, current speed, distance and safe indicator are all easily accessibly via swiping. Eliminating click through, small menu options for settings and having to navigate backwards, reduces the need for interaction by the user, therefore increasing the safety of the usage of the application in a driving situation.

See appendix H for all the UI screen designs for the application.

4.4 – REQUIREMENTS

DSDM employs the usage of MoSCoW prioritisation for requirements, ‘Must, Should, Could and Wont have’ prioritisation will be applied to all the requirements captured. Taking into consideration the complexity and the timeframe available for completion into account to produce a set of requirements that will be full filled during implementation.

See appendix I for the detailed Requirements List

4.5 – TIMEBOXES

After the requirements capture, to provide structure of the implementation of the artefact and aligning with the use of Timeboxes within the DSDM methodology. 9 Timeboxes templates have been created to document the required functionality that will be achieved in each timebox. See appendix J for all 9 timeboxes. The templates also served as part of the testing process for testing each timebox therefore the testing criteria and results will be contained within the tables.

For Timebox number 6 the start and end date was not documented as this timebox documented the ‘Could’ functionality of the artefact and would be attempted if time allowed from completing other timeboxes early.

CHAPTER 5 – IMPLEMENTATION

5.1 - IMPLEMENTATION

Prior to starting the implementation, the Arduino device, laser range finder and Bluetooth was purchased ready for use. The Arduino was purchased as part of a starter kit that came with a breadboard and wires for ease of development. http://www.amazon.co.uk/Arduino-Starter-Kit-UNO-Board/dp/B009UKZV0A/ref=sr_1_3?ie=UTF8&qid=1460544182&sr=8-3&keywords=arduino+starter

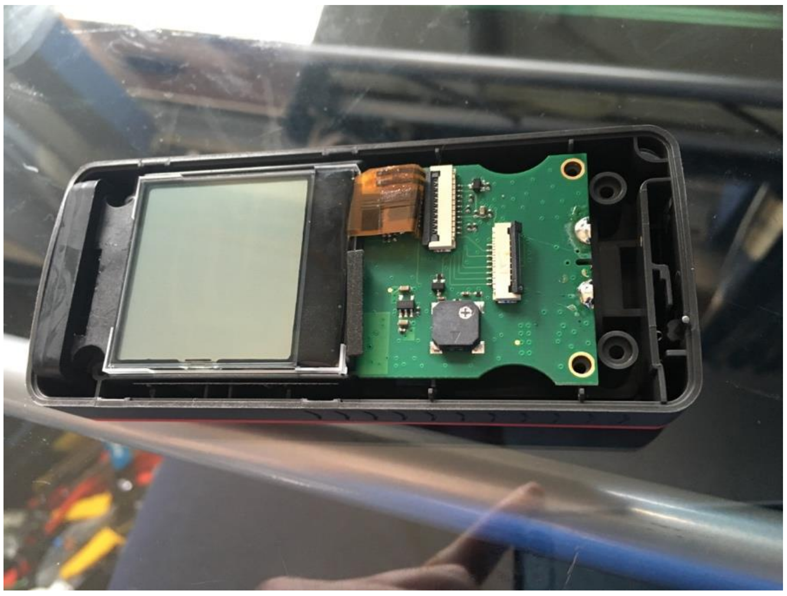
5.2 - TIMEBOX 1 - DEVELOP HARDWARE DEVICE.

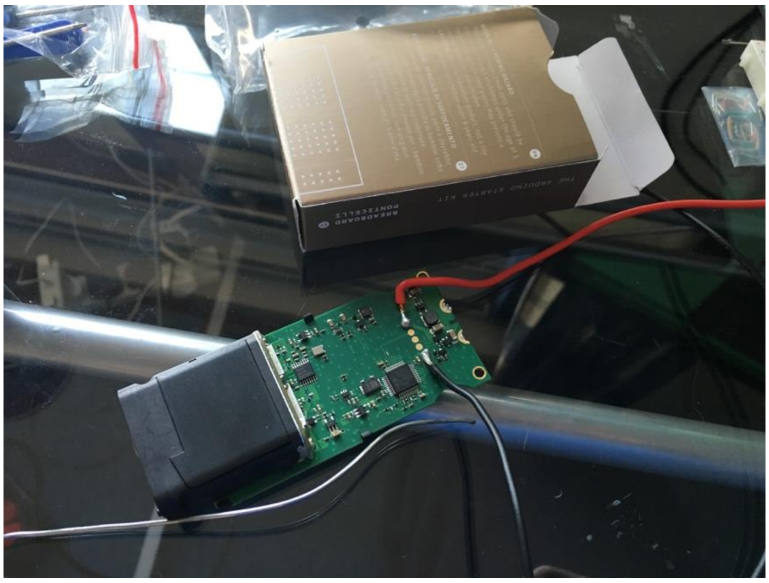
Implementation started with Timebox 1. The deliverables for this Timebox are to have the laser range finder and Arduino communication completed. As an abstraction layer from the application itself, the Arduino will receive communication over Bluetooth to initiate commands and also send results back over Bluetooth. The communication protocol is Bluetooth Serial, which is a standard, therefore making the device work universally across many devices. Therefore, it meant that the hardware can be developed without the need for a windows phone application to test with. The hardware was tested by using a Laptop with Bluetooth. Using a simple program called 'serialTools' which allowed a Bluetooth Serial connection to the Arduino's Bluetooth module, the laptop can then act as if it was the windows phone application, by initiating commands and receiving commands that are read back on screen.

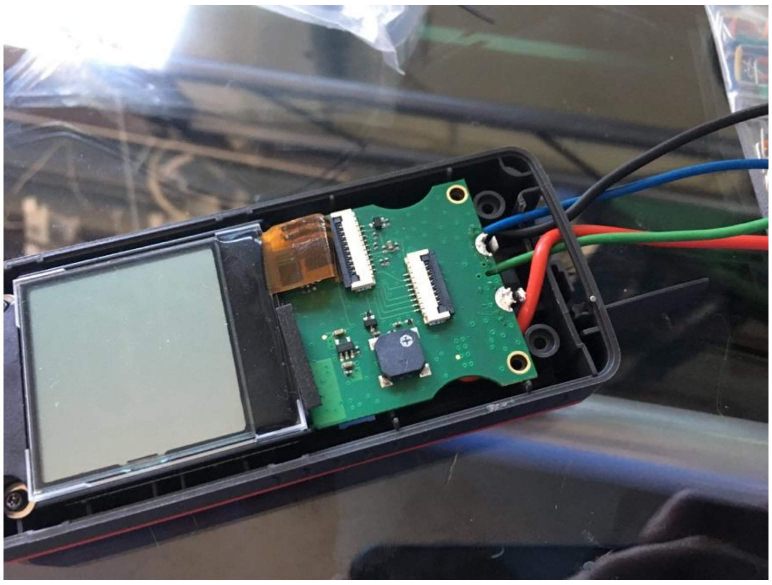
As stated in the design documentation, the laser range finder debug port is to be connected to the power, ground, transmit and receive pins on the hardware serial port of the Arduino. This involved taking the laser range finder apart and soldering wires on the required pins. Using the guide that Fuller had written on his blog that shows the disassembly process and the pin-out diagram (Fuller, 2013). This process was completed and then the process of connecting the Bluetooth module was undertaken, Using the breadboard as part of the starter kit the Bluetooth module was placed on, the breadboard consists of hot swappable pins making it easy for prototyping and developing. The required pins were connected to the Arduino as per the design documentation.

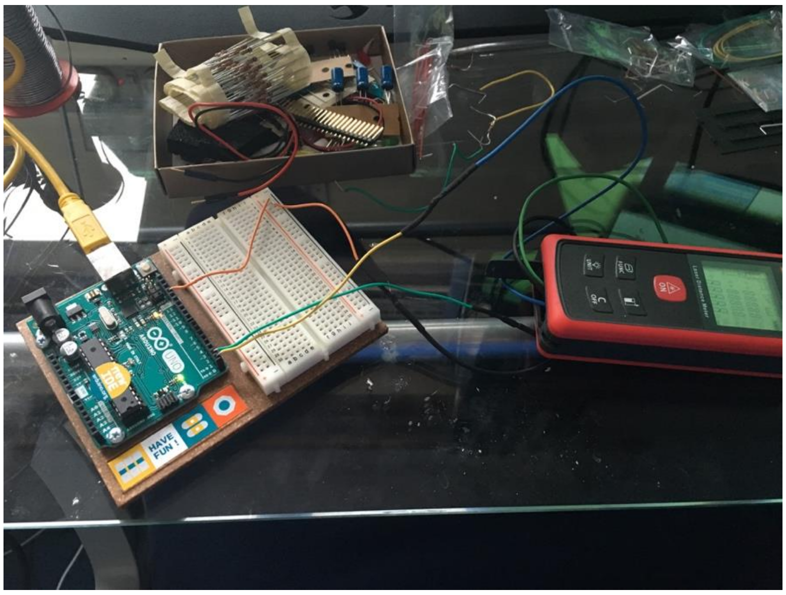
Pictures of process as follows:

CAUTION
LASER RADIATION
DO NOT STARE INTO BEAM
MAX OUTPUT<1mW
WAVELENGTH 620-690nm
CLASS 2 LASER PRODUCT
BATTERY 2*1.5V(AAA)









With the hardware attached the next process was to start developing the Arduino sketch that will run and communicate with the laser range finder and the Bluetooth module. The analysis and design mapped out the conceptual pseudo code which was followed. Many issues halted development during this stage. The Conceptual idea was to receive a request over the Bluetooth module to initiate the reading from the laser range finder. This was achieved but was noted that there was latency with the laser range finder starting a reading and returning the reading back. Over 1 second which was unacceptable according to the analysis as this was required every second. However, the laser range finder has a quick fire mode. That is also available for use over its debug port. As pointed out in the comments section of Fuller's blog. Once initiated the range finder will take as many measurements in a constant stream. Approximately 2.5/3 per second. This makes it possible to get more accurate data of the distance tracking as it can be updated quicker with less latency. Furthermore, this meant a slight change in design and requirements. Instead of the application initiating a distance request every 1 second, the application will send a request to the Arduino to initiate the burst fire mode of the laser range finder device and then constantly listen for the results that is being produced and then relay the onto the Bluetooth serial port, so the phone can pick them up. Effectively swapping the flow of communication. The application will just listen for incoming distance readings constantly as they arrive after the burst mode has been initiated instead of asking every time.

This presented another hurdle of development. The laser range finder constantly sends out serial data when in this mode. The serial data is not just a straight forward reading like '4.5m' it consisted of a sequence of characters with the distance reading embedded inside the sequence. For e.g (*#008430004500#) would be a reading of 4.5 meters. The nature of the Arduino's serial port has a 64-byte buffer, and that has to be read from to get data. As once the laser is initiated with the burst mode. The serial stream is constantly being sent to multiple times per second. This required the Arduino to read from the serial buffer efficiently so all readings were read and to decipher the actual distance reading out of the sequence of characters that are produced and then send the cleaned reading back over the Bluetooth serial port. The laser range finder reading library that was proposed to be used in the analysis stage was attempted to be used during development. However constant issues arose of the library not reading anything and overall wasn't working. The library consisted of a C header file that contained all the inner workings of the communication with the laser range finder. Within the Arduino sketch an instance of the library just needed to be instantiated and it had methods available to take readings and display the cleaned data. This was not working. To troubleshoot the library was disassembled and interpreted for what the inner workings was. This was incredibly time consuming as limited knowledge of the C language and how serial works on an Arduino took some time, researching on forums to gather knowledge so the code could be interpreted. From the outset the code looked as if it was working as intended. To troubleshoot even further, the library methods were extracted and dumped into the top level Arduino sketch, some modifications were required to convert the C header file language into the Arduino sketch language.

This allowed closer inspection of the workings of the code without the need to modify the library and instantiate it in the sketch where limited debugging was available that way. Multiple trial and error attempts changing the code eventually landed with a working sketch that was successful at reading from the laser range finder in burst mode and relaying the information back over Bluetooth. This was observed using the serial tools on the laptop and observing the incoming readings over the Bluetooth serial connection. The next stage was to implement the burst mode switch on and switch off commands from the Bluetooth serial port, this meets the requirements for the phone to initiate the start and stop of the monitoring. This was completed and the Timebox was now finished. The hardware aspect was now ready for communication with the phone and the application to be developed.

5.3 - TIMEBOX 2, 5 - SCAFFOLD WINDOWS PHONE APPLICATION, SETTINGS SCREEN LOGIC

Timebox 2 was to create and scaffold the Windows phone 8.1 application. This process was completed with very few hurdles. Scaffolding UI with windows applications is relatively easy and involves usage of the built in controls via XAML. Having previous experience in Website development using ASP.net the concept is very similar. There are many difference that had to be learnt on the way, which involved much time looking over stack overflow and MSDN documentation to find solutions. The concept with windows applications is the idea of data binding controls to data sources. This means that when the data changes the controls will update in real time. So concepts and usages had to be learnt while developing the UI but this allowed for a more stream lined application, for example, when a journey gets recorded it will instantly appear in the list of previous journey's that can be reviewed.

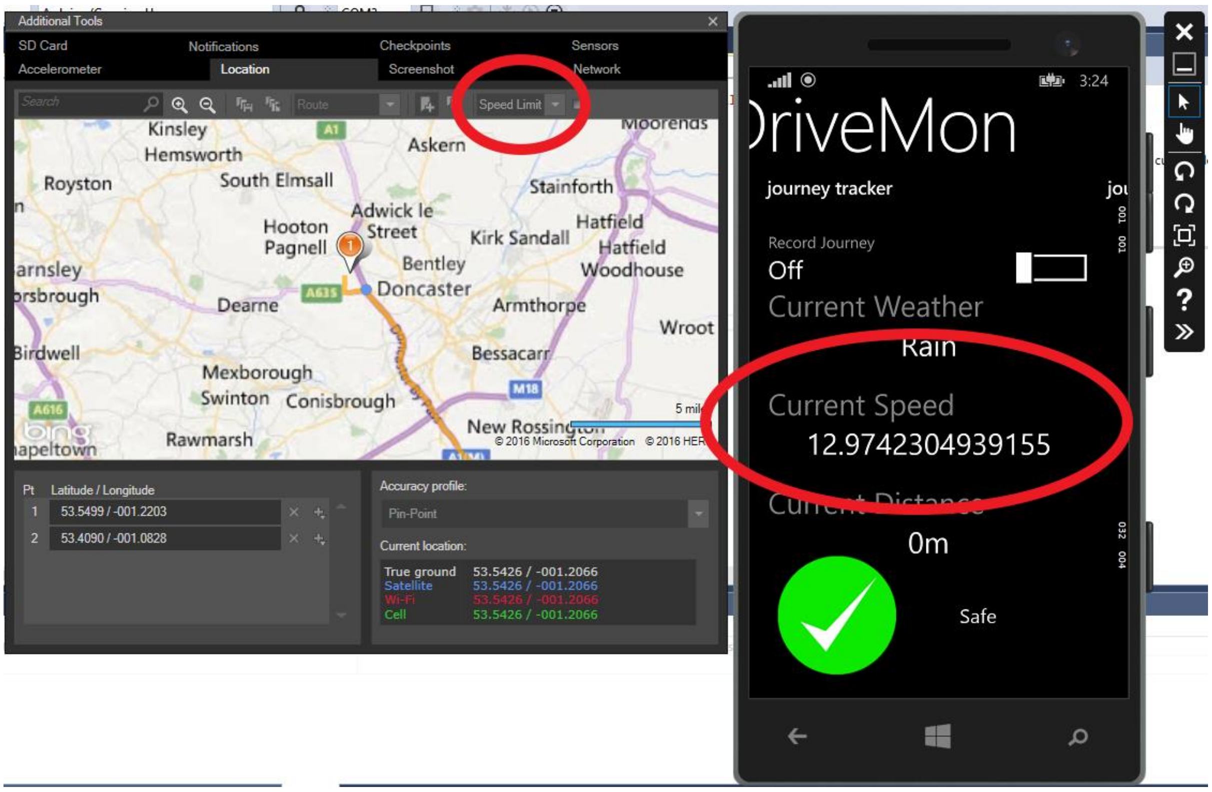
The decision was made to complete Timebox 5 at the same time as Timebox 2 because it made sense to develop the application settings UI and the logic at the same time as the settings are fundamentally used later in the application. There was a challenge present at this stage, how to persist the settings between application closure and opening. Luckily the windows phone 8.1 library has built in classes that allow storage of simple data in key value pairs. So after a little research and experimentation the settings aspect was developed. Concluding the completion of Timebox 2 and 5.

5.4 - TIMEBOX 3 – BUILD GPS LOGIC

With the main aspects of the UI scaffold and the settings logic developed attention was turned to the development of the GPS tracking logic. As per the analysis the conceptual system was going to poll for GPS data every second. When it came to developing the logic, it was realised that the GPS classes within the windows phone library allows the application to subscribe to a 'Location Changed' event that has adjustable precision. So the precision can be set to update as close to every second as possible. Leading to removing the need to poll every second, the GPS tracking logic can be executed within the event that gets raised by the phone itself and this will carry on for the entire life cycle of the application. At each raised event you get access to GPS coordinates and the speed value (in meters a second). This became an issue with development and testing. Microsoft offer an emulator to use while developing and is very resourceful, it allows accelerometer, sensor and GPS emulation. You can plot a route and then virtually follow that route, this was very useful debugging the GPS location. However, a very big issue was identified with using the emulator. The emulator allows you to simulate a journey at a chosen pace, there are 4 options, walking, speed limit, biking and fast. Allowing the developer to debug at different speeds. In the case of this mobile application, speed limit seemed suitable to pick to simulate a route. However, it doesn't matter which speed is picked, the speed reported in the application is always 12.97. At first it was thought that there was a problem with the code, but after some research had been done it was found that this is only an issue in the emulator, apparently it is a hardware only option.

Up to this point in development a physical device hadn't been sourced for testing. All testing and development was undergone using the emulator. So after coming across this issue a physical device was sought after to use for development and testing. A cheap windows phone was found on a buy and sell website and was purchased. This allowed the theory to be tested. The application was installed onto the device, mounted in the authors vehicle, and a test journey was undertaken, observing the screen as driving took place, the speed reading was changing regularly and was accurate compared to the speedometer of the vehicle. This proved the theory that the emulator was an issue. An issue that could prevent the pace of development as testing of the speed could only be done physically by driving. However, the speed reading can be easily spoofed for testing purposes as is something that will be discussed later.

Screenshot of the emulator reporting a solid reading for speed:



Pt | Latitude / Longitude
1 | 53.5499 / -001.2203
2 | 53.4090 / -001.0828

Accuracy profile: Pin-Point
Current location:
True ground 53.5426 / -001.2066
Satellite 53.5426 / -001.2066
Wi-Fi 53.5426 / -001.2066
Cell 53.5426 / -001.2066

As part of this Timebox development was to record the journey if the user toggled the record journey button. This presented another hurdle to overcome. The concept of data saving in windows phone 8.1 application is focused on flat file storage. During the analysis and design stage it was identified that there would be a database storage context. This was ignorantly based on old documentation that was found during research for windows phone 8, In 8.1 There is no built in database support. Instead the classes that present entities for data storage can be serialised into flat files, then de-serialised when needed. However as per the class diagram in the Analysis section the 2 classes of 'Journey' and 'Polled Data', where each Journey has a collection of Polled Data classes is used within the application. Modification to the way the data is stored had to be implemented different to the conceptual class diagram. Simply a collection class is created that holds a collection of journeys, which inside each journey a collection of polled data. Then using built in windows phone framework code implementations, the data can be loaded into the application at start-up via de-serialisation, and saved when a new journey is added by serialisation. Concluding the implementation of Timebox 3.

5.5 - TIMEBOX 4 – BLUETOOTH CONNECTION AND POLLLING LOGIC

Having completed Timebox 1 where the laser range finder and Arduino are all working with the Bluetooth serial communication, it was realised that the design of the application would have to be changed slightly. Instead of the application polling for a distance reading, the application now needs to initiate the start of the burst mode and then constantly listen for incoming serial messages.

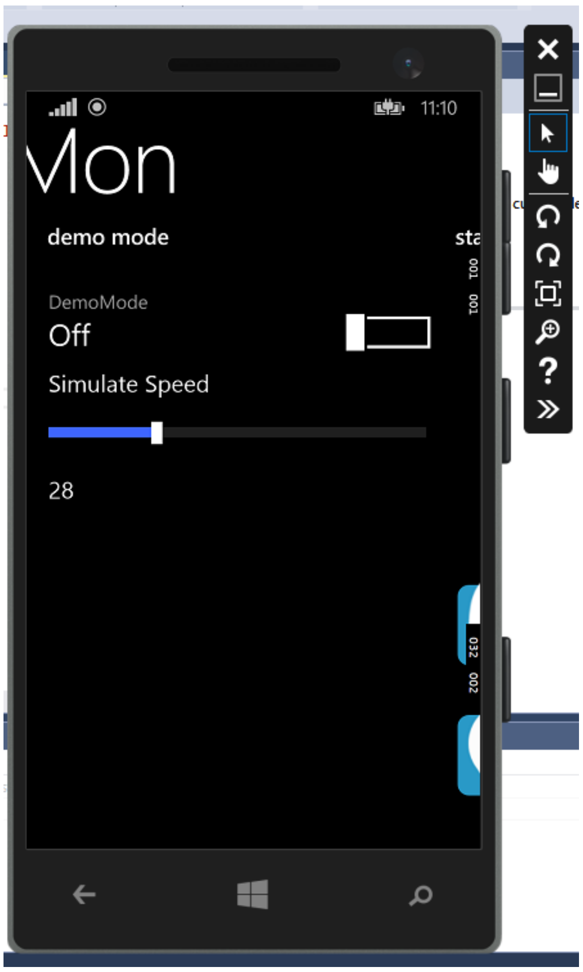
No knowledge of Bluetooth communication was known before commencing development of this Timebox, apart from windows phone does support serial port profile (SPP) as found out in the research. So the process for developing this section involved researching on MSDN and gathering ideas on the internet to find a solution. During this, a blog written by Joshua Drew was found that details his journey through developing a windows phone 8.1 application that can control a Robot that is made with an Arduino. He has is code library freely available for people to use and was downloaded and interpreted. The Bluetooth connection part of his library was used in the development of this phone application. Many tweaks were required for it to fully function to meet the requirements of this project however the connection and handling was completed by the class library which made the development process of this Timebox significantly quicker. The library fires an event every time a message is heard on the Bluetooth serial port. Therefore, in the main application execution class, the event had to be subscribed to and then this should get fired for every distance reading that comes through from the Arduino. However, under testing this method wasn't always being fired for every reading, it was irregular. The laser range finder could initiate readings 2-3 per second, but sometimes the event was fired every second, or not for 2 seconds but dumping all the data in one go, leaving the message being received in the event many lines long, instead of just 1 single line with 1 distance reading in it. Some time was spent trying to tweak and improve the reading of the data, so that the event gets fired more regularly. It seems to be that the way the serial port works is very similar to the Arduino, the concept of the Bluetooth serial port listening in the Bluetooth class library has a buffer, and every so often this gets read. Due to limiting time constraints, there wasn't any time left in the Timebox to consider finding and improving the issue. The data being received was accurate and usable but was received at irregular times, not every second as set out in the analysis stage. it is something that had to be taken into consideration when developing the monitoring logic later.

5.6 - TIMEBOX 6 – BUILD MONITORING LOGIC

At this stage, the GPS and Distance tracking metrics was working and ready to be incorporated into building the monitoring logic of the application. This requires incorporating the identified stopping distance formulae and using the known current speed and distance to calculate if the user is within a dangerous distance behind the vehicle in front.

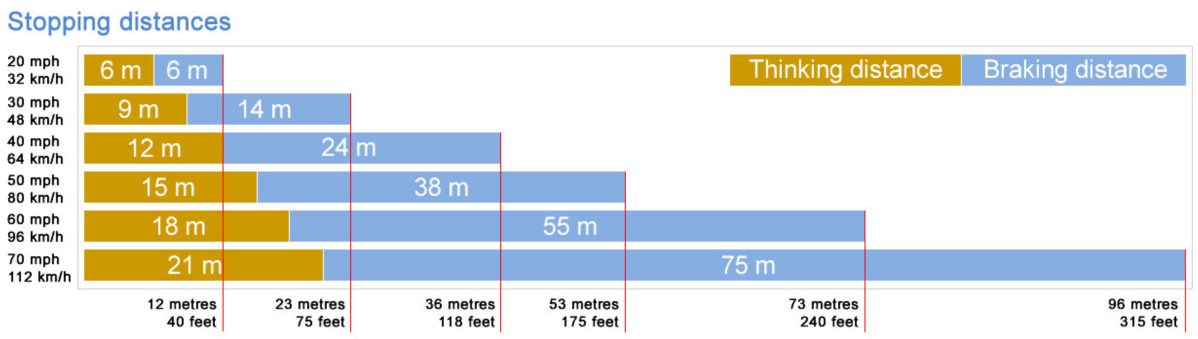
As the requirements dictate that there should be some form of smoothing built in to attempt to eliminate false and erratic readings it was decided that if the monitoring indicated that the user was in a dangerous level for more than 5 seconds then a warning will be shown. It sounds simple when describing the requirement but when it came to implementing in logic it was difficult. With Window Phone there is a class library that allows a developer to create timers, and execute code on a new thread when the timer expires. However, there needed to be multiple checks within the 5 second timer as if the user re-entered a safe distance, this timer needed to be reset. This was overcome by many trial and error tests using spoofed distance readings by just placing buttons on the UI and wiring them up to spoofing methods.

The spoofing mode was not something that was documented during the analysis and design but during development it became clear that it was not easy to test functionality in the way the application was intended as it would require actually attaching the devices to a vehicle and going for a physical drive. Which would have been very inefficient and would have led to costs incurring and a very lengthy development time. Using spoofed readings made it very easy to test as to do it in the context of the full application, this was done by creating another section on the panorama view with the ability to turn 'Demo' mode on and off, with a slider to spoof the speed in mph. See screenshot below of the demo mode:



With the spoofing of the speed available for testing, there was possibility of including the spoofing of the distance reading too. However, this would be like the speed and be set to a static number and would not reflect the way in which the readings would be received. It was observed that if you divide the standard stopping distance guidelines by a factor of 100 then all the guidelines fit onto a 1-meter ruler. See the image below:

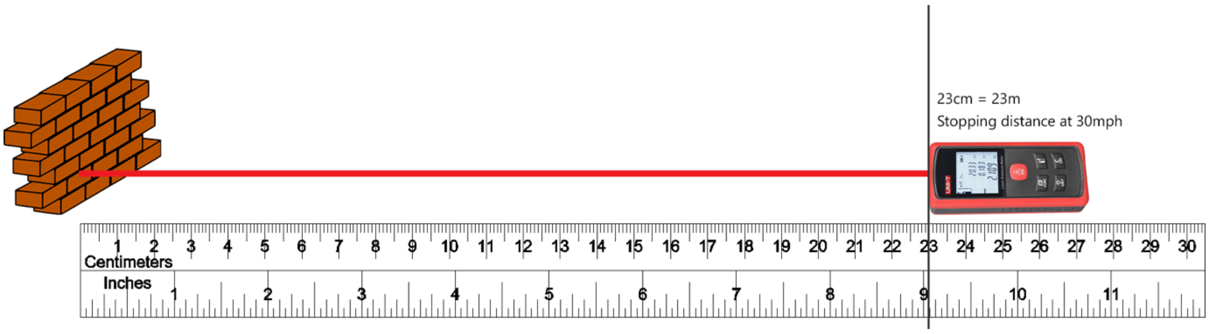
It is observed that at 30mph the stopping distance is 23 metres. Divided by 100 is 230 centimetres and at 60 mph this would be 960 centimetres.



http://www.drivingtesttips.biz/images/stopping-distances-graph.jpg

This lead to incorporating this into the demo mode. When switched on the distance formulae gets divided by 100 so that testing can be done within a 1 metre scale. Allowing for testing and development to be conducted as close to the real application as possible by using the laser range finder and the Arduino as it was intended to be but within the comfort of small desk area. Manipulation of the range finder can be done by hand to simulate distance changes as a car would get closer/further away from a vehicle in front.

Diagram illustrating the demo mode used for development and testing:



Refereeing back to the use of a timer mentioned previously there was another issue that was encountered. The development of Windows Phone applications relies on the use of asynchronous code, meaning that execution of code is threaded and allows other code to be performed while waiting for others to finish. For e.g., fetching of data while the rest of the application continues to function. The timer class has executable code that is run at the end of the timer period, which is ran on a separate thread. However, when trying to access UI elements on the thread created by the timer produces an exception due to the fact that access to the UI elements has its own specific thread to handle. Research found a solution from stack overflow that code had to be put in to grab a reference to the 'dispatcher' of the UI which allows access to UI elements during code execution.

After the hurdles were overcome the monitoring logic was built and the deliverables of the Timebox where complete.

5.7 - TIMEBOX 7 – BUILD REVIEW JOURNEY SECTION AND PAGE

With all the other functionality now complete the reviewing of the journeys was left to implement. This is a section of the panoramic view that lists all the recorded journeys and each item can be clicked through into a separate screen that gives more detail. The section of the panorama was already scaffolded during Timebox 2 therefore it was just a case of coding the data binding to show the list of recorded journeys, this didn't take long. Prioritisation was given to completing deliverables 3,4,5,6 over 1 and 2 as this was seen to be the most time consuming and deliverables 1 and 2 was not seen as important as all they do is present extra data to the user for convenience and not add functional quality to the application.

Each journey item can be clicked and takes the user to a separate page. This page was also already scaffolded with the Map control and the list ready to be implemented. Development of the map control was difficult and took much time. It required reading the Microsoft MSDN documentation and reading various stack overflow answers to help find a solution. For each journey there was a list of coordinates recorded (every 1 second) and translating these into a plotted journey resulted in many small iterations of trial and error. At first it was thought that it could only be done using Microsoft's Bing maps API that can be used to get generated journeys plans. Essentially you can give a start and end point and a journey would be generated back to the device. This would not work as the journey would be completely different to the actual journey that was recorded. Later a solution was found using a PolyLine class in C# that allowed a list of coordinates to be drawn as a line on the map. This satisfied the requirement well.

The next stage was to implement the interactivity of the map and plot red dots where incidents occurred and to pop up a detail bubble once clicked. This was easy to implement as the Map control can have children of drawable objects that can be placed at co-ordinates on the map.

At the end of the Timebox the review journey section was mostly completed, the Total mileage and Last journey mileage was not implemented due to time shortages but these was seen as desirable features for the user and the fundamental functionality was completed as priority.

5.8 - TIMEBOX – 8 IMPLEMENT S PRIORITY REQUIREMENTS

At this stage all the 'Must Have' Priority requirements are completed and this Timebox was intended for any requirements that was categorised into 'Should have' to be developed time permitting.

Part of this Timebox was to develop a logo. This was completed and implemented in the application.

The other requirements involved including weather as another metric for calculation of the stopping distance formula. Including automatic weather grabbing once the user starts monitoring. This was achieved by using a free API called OpenWeatherMap. It allows the device to send the coordinates and received back weather information. This is permitting if the user has access to the internet on there device. Otherwise it would prompt them to select the weather that they observe. If it is damp or raining, then stopping distance doubles. As documented in the research. So it is important to gain weather data for more accurate monitoring and warning for the user. It was easy to modify the monitoring logic as a simple check for if the weather was wet then it would just double the distance reading.

Unfortunately, the time allocated for this Timebox had run out. The start time for this Timebox was already late due to Timebox 7 running over. The requirement that specifies prompting the user to input the weather was not completed. It was the decision to complete the automatic weather and get the application stable without errors and if weather data was not available then the application would just default to the standard stopping distance formula.

CHAPTER 6 – TESTING

6.1 - REQUIREMENTS TESTING

The requirements set out in the analysis and design was tested and documented via a table containing a list of all the tests conducted to satisfy the requirements listing with a test procedure description, expected results and pass/fail/comments. See appendix K

Some requirements were not fully met or was deemed a failure:
Test number 14, matching requirement 5.2 which should display the distance readings updated every second was a requirements that wasn't fully met. Due to the way the Bluetooth messages were handled using a buffer, and the smoothing introduced to mitigate false positives, every second update was not achieved. However, the functionality achieved proved to be sufficient.

Some of the 'Should' and 'Could' priority requirements were not implemented. Leaving 5.8, 6.2, 7, 11.7, 9.2, 9.2 12, 12.1, 12.2, 12.3, 12.4, 12.4, 12.5, 12.6, 12.7, 12.8 of the requirements not being fulfilled. This was due to time constraints but these were the requirements that were not necessary for the main functionality to fulfil the proof of concept.

6.2 – FUNCTIONAL TESTING

Much of the system was tested ad-hoc. Formal automated unit tests were not completed for the mobile application; due to lack of technical skill of test driven development it was deemed to be a time consuming process to implement and as a result could jeopardise the development of a working system. Therefore, the decision to not include them was made. In contrast, the entire system was constantly unit tested informally and undocumented. The development and iteration process involved writing some code for a piece of functionality, running it and evaluating the results in a constant cycle.

As functional testing was difficult. It would involve mounting the device and physically driving and observing the outcome every time new functionality was added, at cost of time and expense. As documented in the implementation testing was possible on a 1m scale. Saving time, allowing for the system to be tested more frequently, resulting in a better developed product.

Final functional testing was completed in a real world environment. See the attached video in the appendix (k.1) for a recorded physical demonstration of the system in action. The testing was conducted on a quiet estate with the help of another driver that drove at 20mph in front of the vehicle that had the system attached to. The resulting real world testing found out that the system worked up to 20mph/15m distance. More than that and the system was not reliable or could not get a reading at all. This is possibly due to the laser not being powerful enough and has resulted in an area for investigation and extension. However, the proof of concept is realised.

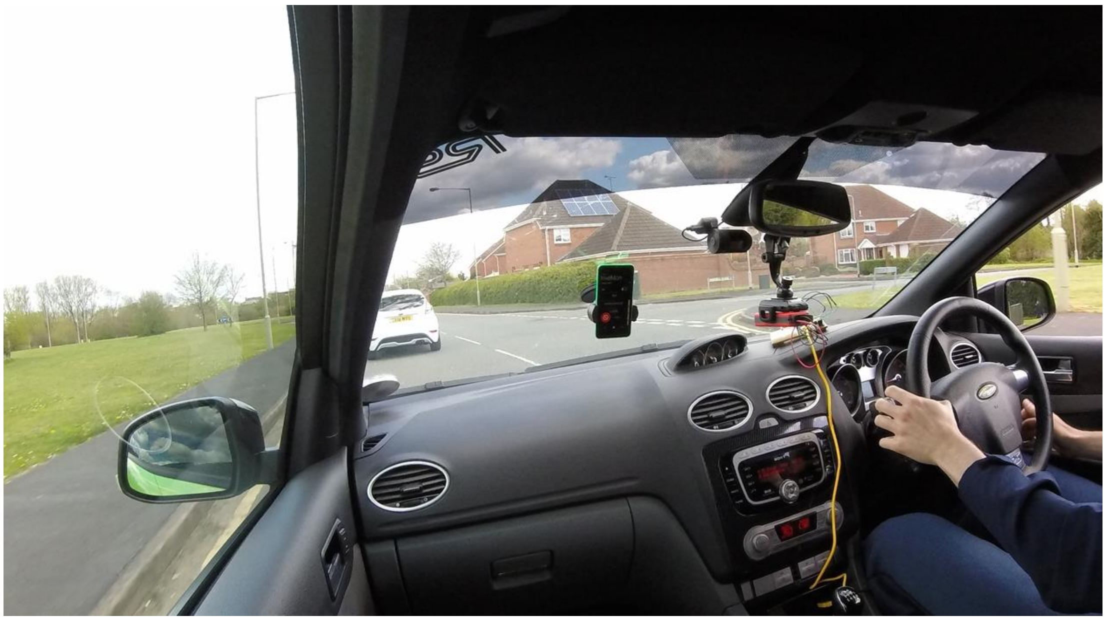


Figure 25 - System in action

CHAPTER 7 - CRITICAL EVALUATION

From the requirements testing and the physical functional testing shown by the video the concept of the project is shown to be successful. The hardware device was able to take distance measurement readings and relay this back to the phone as a metric for calculating how safe the driver is. In the current state it only works semi-reliable up to 20mph and in straight lines which is far from ideal as the stopping distance guidelines are given up to 70mph, but does leave the area open for extension to find a more reliable distance measuring source that will give greater distance and reliability. There were also many negatives and positives along the journey that shaped the project as the process went on.

7.1 - PROJECT MANAGEMENT

The management of the project proved to be the greatest learning experience. During the inception stage of the project, where the initial ideas were gathered and proposed, a rough Gantt chart was created containing the basic layout of what was thought to be the project structure. See appendix L.1 for the original Gantt chart. Sections analysis, design and implementation was included with little details in between of what was initially thought to be part of the process, drawn from only previous experience and knowledge of a project management process. As the initial stages of research underwent, starting with the methodology research it became clear that the project plan had to be amended and given more detail. As the chosen methodology was incorporating timeboxing and MoSCoW prioritisation, planning of the timeboxes and the timeframe given to them needed to be added to the project plan to make it clear what areas needed to be completed and started at certain times. In addition, UML was chosen to document the analysis and design of the project and as such the various diagram drawings were added into the project plan to visualise the time needed to create these.

An important aspect overlooked at the start of the project planning process was the additional time allocated to completing other academic work. The original plan (Appendix L.1) didn’t take into consideration the time needed to complete other academic assignments, as such too much time was allocated to aspects of the project which would have been hindered due to time constraints. In the revised plan (appendix L.2) task 19 has 21 days of allocated time for perusing other academic work. The additional details that were uncovered during the early research stage and the implementation of allocating time for other work gave greater visualisation of the time available for the areas of the project, making the revised plan much more valuable.

In review of the project plan, task 38 details 35 days allocated for testing but is ran in parallel with the implementation section. It was planned for unit testing to be completed throughout and with the iteration style, meant that testing could be done in parallel during development. However, the implementation of the artefact ran over the allocated time on the plan, leaving formalised end to end and requirements testing cut short and rushed. The consequence of this meant that the artefact has not been tested as widely as possible. It was uncovered very late during the physical testing in a real life environment that the gathering of distance data using the laser range finder is very temperamental, and does not sustain consecutive readings in order for the application to react appropriately. This could be down to the way the device was mounted and the angles etc. This was discovered too late for any rework to be applied to the artefact for improvement. If this had been tested in detail earlier on then sufficient fixes could have been sort to improve the artefact and conclude on a more reliable product. On the other hand, as a proof of concept the idea works, it has left the foundation of using consumer grade laser devices as a viable product to work for the requirements the application is intended for. More research and development work is needed to iron out the uncertainties to create a more reliable application.

Looking back at the project process, more time should have been allocated to implementation. In the revised plan it is shown that implementation wasn't to start until the 22nd February. The first project plan Implementation was to start on the 8th of February which is 2 weeks earlier than the revised including a 35 days' implementation section compared to 28 days in the revised plan. The later allocation was due to the research analysis and design process taking too long. This was partly because of the additional allocated time for other academic work and partly due to personal effort. Looking back there was a lack of time and effort put into the initial stages of the analysis and design, progress was slow and this resulted in the required documentation not being completed in time for implementation, therefore in the revised plan the implementation was cut short. This had a knock on effect where implementation ran over, hindering the testing stages.

However, it is noted that within the revised plan the contingency placed at the end of the project was increased to 16 days. What was previously mentioned before about cutting the implementation short wasn't entirely necessary and could have possibly been extended leaving a shorter contingency timeframe. However, when trying to stick to a timeframe on the Gantt chart it provides greater incentive to complete the activity. If the implementation was extended, shortening the contingency, then it is possible that the sense of urgency to complete the implementation task is less, resulting in a lackadaisical frame of mind that could lead to even less work being completed and possibility of there not being enough contingency and a project failure. The contingency was kept at 16 days for this reason. The contingency period has been used; the implementation of the application development overran due to complications within coding and finding solutions to problems. Pushing testing backward and the final stages to finalise the project. Nonetheless, without the contingency then it could be visualised as a possible project failure as time would have run out and the project not completed.

Reflecting upon the proposed testing strategy, this section detailed the use of documented automated unit testing to create repeatable tests in the aid of producing a more maintainable solution. From research it is clear that unit tests are beneficial to the development process and is a sound software engineering practice; however, time constraints and lack of skill lead to the project managed decision to not include these in development of the artefact. It is of the opinion that developing unit testing adds extra time onto development, although this could be argued by others. On the other hand, it could be said that unit tests were completed but in an un-formalised and undocumented manner as the development process involved writing little bits of code to try and achieve the required functionality, running it and evaluating the outcome. From undocumented research there are many articles explaining IT projects that fail due to lack of testing. It is learnt that testing is very important in development and needs to be given more effort and time. Therefore, this is an area that would be looked at in greater detail if the project was to be repeated

In contrast to the negative areas, the implementation of the DSDM MoSCoW prioritisation and Timeboxing proved to be a good technique and extremely beneficial. Setting out requirements and prioritising them allowed for controlled scope delivery. The judgement was made on what was thought to be possible to implement within the time constraints and this formed a greater process for delivering a working artefact. In addition, the Timeboxing technique allowed the separation of tasks leading to the artefact being developed with greater efficiency. Without these 2 techniques the lack of structure could have proved fatal to the project as some requirements could have had too much time spent on them and others neglected resulting in the artefact being improperly developed with nothing working correctly.

Another personal trait that proved to be useful is being honest and knowing when something is unknown and asking for help. Situations where finding out answers to issues can prove to be very time consuming and may not get the answer that was sought for in the first place. Asking for help from the supervisor or friends was very useful to get ideas on how to approach the problems faced. Additionally, the project supervisor conducted meetings in a group style and encouraged the use of a private Facebook group where our supervisory group peers could talk and ask questions. The use of social media was a great tool for getting peer feedback and to get answers from the supervisor which others could benefit from. Regular ideas, questions and ‘banter’ was exchanged in the group, including the analogy of comparing a successful project to the process of baking a good cake, which helped provide sparks for ideas and solutions to problems which worked great as part of the project process.

At first the use of UML diagrams for system documentation didn’t seem that it would be beneficial to the project, this was probably due to the combination of self confidence and skills in using UML. It is an area that I personally hadn’t had much exposure too. Having spent over a year in a professional environment on placement developing applications for a large company, they surprisingly did not use documentation for the design and analysis of their systems. Only the requirements capture and overall outline of what the application would do was given, leading to the development having no structure and being ad-hoc. Within the analysis and design of this project, the use of UML actually proved to be extremely beneficial. Even though the finished product doesn't fully align to what the documentation set out, due to the iteration style this was expected as problems are uncovered and knowledge of the chosen technology is greatened through the iteration process. The project itself had different areas (components) that can be developed independently; the UML documentation of this gave structure to what areas needed to be completed, and what functionality they needed. Consequently, they provided basis for the Timebox design.

The UI design of the application is an area that went very well. Microsoft supply Adobe documents that contain templates for the different controls of the Windows Phone interface. Having previous experience with Photoshop it wasn't difficult to produce the designs required. Comparing the UI designs to the finished product, the end result closely followed what the design originally entailed, with some exceptions of the drop down warnings.

7.2 - PROJECT EXTENSION

The mobile application itself is well developed and the concepts and techniques that underpin it are sound, the application is more reliant on the external hardware aspect itself producing better results for the application to be better. As it is an application designed to monitor the driving of a user, the other applications on the market all track accelerometer and cornering data to produce ratings for the driver, and/or for insurance companies to track user's safety. The scope of this project was to find out if distance tracking can be possible and as such the other areas like cornering and acceleration was deemed out of scope for development. Therefore, if the project was to be extended these extra metrics could be brought into the application and used to rate the user on their driving style, and/or provide statistics for insurance companies. Much research would have to be undertaken into algorithms that use the accelerometer and cornering data metrics as these are areas that do not have much exposure.

As the project is a proof of concept to see if it is possible to have a hardware solution made from relatively inexpensive consumer grade technology that someone or companies could find viable to purchase it has left the room for extension very open. The chosen technology for gathering distance data was done via a single laser, inherently laser beams are completely straight and any movement could throw inconstant readings. It was documented early on about this issue (see technology research) with a possible solution of developing smoothing algorithms to smooth and mitigate this problem. It became clear in testing that this is a very big issue. It makes tracking distance on bendy roads difficult. In straight lines the application proved that the idea does work. In a motorway scenario it could be said that the solution would work very well as motorway roads are quite straight and would make tracking distance easier.

Therefore, this provides a great deal of extension for the project. In the research stage it was documented about the use of video and tracking vehicles using the OpenCV technology, this was discounted due to it being out of scope for the project; however it could be a possibility to go back and see if this could be a viable solution to extend the project to create a better product.

It could also be considered to increase the number of laser devices attached to the vehicle, having some pointing at slightly separate angles (left and right) using algorithms to correlate the data could provide a more accurate experience.

It is also noted that the laser device chosen for implementation is a low cost item that is limited to a range of 50 meters, using a low power laser. Meaning that in the current form the project could not satisfy the guidelines up to 70mph due to distance limitation. In addition, the reliability diminishes the greater the distance away from the object. To extend the project, more research could be completed to find an alternative solution to the device chosen, something that outputs greater power that will track at a much higher distance. However, this might sacrifice the goal of the project of producing something using cheap consumer grade technology.

Another possible extension could be the use of ultrasound technologies, something that was touched upon during research but proved to be too expensive and out of scope is the use of ultrasound. Similar to how vehicle reversing sensors work, but only within 3-4 meters. It may be possible to extend this technology on a larger scale that would provide long distance tracking.

It is clear however that any technology used to gather distance data needs well developed algorithms around the technology to produce useful results. Due to the nature of how vehicles travel, and many obstacles presented around the idea, algorithms that smooth inaccuracies and broken results would need to be developed further.

The training mode that was documented as a 'could' piece of functionality is an area that would serve great purpose and needs to be considered for the project extension. The stopping distance calculations set out by the government are only average guidelines. This means that many vehicles will surpass or fail these guidelines. Making the application developed not accurate for all vehicles. Therefore, the project could be extended to include this training mode to allow the user to get a more accurate stopping distance calculation for their vehicle. However, this does produce other issues like how safe is the training mode, as it would require the user to effectively do an emergency stop, there are many variables in this procedure that can be unsafe, and costly to the driver as of wear and tear to the vehicle. In contrast if this was produced on a commercial level then it may be possible as a company with investment to conduct research into different vehicle makes and models and their typical stopping distances with original parts. Then the application could ask the user what vehicle they are driving and the user could select their vehicle and the application could adjust the calculation accordingly.

7.3 - IF THE PROJECT WAS TO BE REPEATED

There are many aspects to the project that could be improved if the project was to be repeated. Some points were touched upon in the project management section previously mentioned.

The main area that could be improved upon is the time management aspect. Furthermore, there are other aspects that link in with management of time. Namely having self-confidence, because of the size of the project and all the vast amount of areas that needed to be covered it became overwhelming. Resulting in a lack of self-confidence to complete anything. This then leads onto a lack of motivation and a lack of progress. It is a downward spiral. Having the self-confidence would have meant greater productivity in completing the work required. Further self-confidence comes with using other tools and techniques to structure the processes involved.

The initial project plan wasn't completed quickly enough and no tools were used to give visualisation of the project areas and as such resulted in the lack of time input to starting the research. If the project was to be repeated, then starting a big project needs structure from the beginning. Mind mapping is something that was brought in at latter stages of the project and considerably helped improve the process of gathering ideas and producing work. A tool like this should have been implemented at the start to drag all the confusion and mind mess out onto paper so it is visualised and structure can then be applied. Which would have boosted the confidence level and provide details on areas to work on.

The project methodology research was concentrated on too much. Comparing 5 methodologies in great detail lead to an abundance of content, reducing room for other important aspects of the project. Extra time has had to be applied to this section to condense the material. In contrast, this was one of the first aspects of the project undertaken, which is the first of its kind within the other academic work undertaken at university. Therefore, it was a learning experience in writing in strict academic style. It was of the advice of the project supervisor to go through this learning process of writing as much as the mind would allow, with the view to go back over later and condense the material. It's the learning process of being able to research for material and write about it. As more research was undertaken the amount of text written got better and more condensed, as the skill of concise writing developed. Therefore, with hindsight, if the project was to be repeated the time spent on researching methodologies would be less.

Implementation of target planning and reviewing is a project management process that would be implemented if the projected was repeated. Partly the reason why progress stalled was due to the lack of structure in milestones or targets. A lazy mind-set of 'it's ok, do it next week' resulted in poor performance in the first 3 months of the project. Implementing milestones and targets, daily or weekly could have prevented this and provided personal feedback on where targets were not met and how the issue could be resolved.

Prototyping the hardware implementation could have proved a benefit during the research, analysis and design. Having completed and tested the artefact, it is clear that the hardware distance measurement is a bottleneck for the success of the application. As previously mentioned the single laser beam is not ideal when it comes to cornering etc. If this was prototyped and tested in isolation without the development of the application, it could have provided foundation for further research into methods that could prove more accurate. In contrast this was a time constricted project and prototyping would have added extra time into the project, but if the management was better than this time would have been available.

BIBLIOGRAPHY

OpenCV (no date) *OpenCV - ABOUT*. Available at: http://opencv.org/about.html (Accessed: 22 October 2015).

Adobe (no date) *Adobe AIR*. Available at: https://get.adobe.com/air/ (Accessed: 6 November 2015).

Amazon (no date) *Amazon.co.uk: Laser rangefinders*. Available at: http://www.amazon.co.uk/s/ref=sr_st_price-desc-rank?keywords=laser+rangefinders&rh=i%3Aaps%2Ck%3Alaser+rangefinders&qid=1446039093&sort=price-desc-rank (Accessed: 28 October 2015a).

Amazon (no date) *GOWE 5 to 500M accuracy1MM Multifunction laser Rangefinder distance meter*. Available at: http://www.amazon.co.uk/diy/dp/B00PU7E1XU/ref=sr_1_8?ie=UTF8&qid=1446039096&sr=8-8&keywords=laser+rangefinders (Accessed: 28 October 2015b).

Android (no date) *ADT Plugin release notes*. Available at: http://developer.android.com/tools/sdk/eclipse-adt.html (Accessed: 6 November 2015).

Android Developers (no date) *BluetoothSocket - Android Developers*. Available at: http://developer.android.com/reference/android/bluetooth/BluetoothSocket.html (Accessed: 5 November 2015).

Anselm, H. and Anton, K. (2005) 'A vehicle detection system based on Haar and Triangle features', , pp. 261–266. doi: 10.1109/IVS.2009.5164288.

Apple (2015) *IOS: Supported Bluetooth profiles*. Available at: https://support.apple.com/en-gb/HT204387 (Accessed: 5 November 2015).

Apple Inc (no date) *MFi program - apple developer*. Available at: https://developer.apple.com/programs/mfi/ (Accessed: 5 November 2015a).

Apple Inc (no date) *Swift - apple developer*. Available at: https://developer.apple.com/swift/ (Accessed: 6 November 2015b).

Apple Inc (no date) *Xcode - what's new - apple developer*. Available at: https://developer.apple.com/xcode/ (Accessed: 6 November 2015c).

Arduino (2016) *ArduinoBoardUno*. Available at: https://www.arduino.cc/en/Main/ArduinoBoardUno (Accessed: 3 April 2016).

Arduino (no date) *Arduino*. Available at: https://www.arduino.cc/ (Accessed: 28 October 2015).

Aviva (2015) *Aviva drive*. Available at: http://www.aviva.co.uk/drive/ (Accessed: 27 November 2015).

Bauer, J. (2011) *Xbox Kinect working on Android OS thanks to developers' handy work*. Available at: http://www.talkandroid.com/30685-xbox-kinect-working-on-android-os-thanks-to-developers-handy-work/ (Accessed: 27 October 2015).

BlackBerry (2014) *Bluetooth library - blackBerry native*. Available at: https://developer.blackberry.com/native/reference/core/com.qnx.doc.bluetooth.lib_ref/topic/manual/c_stub_bluetooth_lib_intro.html (Accessed: 6 November 2015).

BlackBerry (2015a) *Bluetooth - blackBerry developer*. Available at: http://developer.blackberry.com/develop/supported_media/bb10_accessory_bluetooth.html (Accessed: 6 November 2015).

BlackBerry (2015b) *Using Bluetooth - blackBerry native*. Available at: https://developer.blackberry.com/native/documentation/device_comm/bluetooth/cascades_using_bluetooth.html (Accessed: 5 November 2015).

Bluetooth SIG (1999) *Specification of the Bluetooth System*. Available at: http://grouper.ieee.org/groups/802/15/Bluetooth/profile_10_b.pdf (Accessed: 5 November 2015).

Bosch (no date) *Bosch GLM 100 C professional – direct digital transfer of measuring results*. Available at: http://www.bosch-professional.com/static/specials/glm100c/gb/en/ (Accessed: 28 October 2015).

Brake (2014) *Risky tailgating and speeding rife on UK motorways*. Available at: http://www.brake.org.uk/news/1223-dl-motorways-may14 (Accessed: 27 November 2015).

Brown, O.L. (2014) 'Arduino Tutorial: Integrating Bluetooth LE and iOS with swift', November. Available at: http://www.raywenderlich.com/85900/arduino-tutorial-integrating-bluetooth-le-ios-swift (Accessed: 7 November 2015).

Business Case Studies LLP (no date) *Primary and secondary research - market research and consumer protection - food standards agency*. Available at: http://businesscasestudies.co.uk/food-standards-agency/market-research-and-consumer-protection/primary-and-secondary-research.html#axzz45z7E98ke (Accessed: 16 April 2016).

Coleman, D. (2015) *Don/BluetoothSerial*. Available at: https://github.com/don/BluetoothSerial (Accessed: 6 November 2015).

CollabNet (2001) *Argouml.Tigris.Org*. Available at: http://argouml.tigris.org/ (Accessed: 6 April 2016).

Confused.com (no date) *Telematics Explained - Confused.Com*. Available at: http://www.confused.com/car-insurance/specialist/black-box/telematics-explained (Accessed: 23 November 2015).

Cook, J. (2011) *Programming the Kinect to work with Android*. Available at: http://hackaday.com/2011/08/08/programming-the-kinect-to-work-with-android/ (Accessed: 27 October 2015).

Department for Transport (2015) *General rules, techniques and advice for all drivers and riders (103 to 158) - the highway code - guidance*. Available at: https://www.gov.uk/guidance/the-highway-code/general-rules-techniques-and-advice-for-all-drivers-and-riders-103-to-158#rule126 (Accessed: 27 November 2015).

Deyle, T. (2009) 'Velodyne HDL-64E laser Rangefinder (LIDAR) pseudo-disassembled', 3 March. Available at: http://www.hizook.com/blog/2009/01/04/velodyne-hdl-64e-laser-rangefinder-lidar-pseudo-disassembled (Accessed: 27 October 2015).

Driving Test Tips (2014) *Stopping distances*. Available at: http://www.drivingtesttips.biz/stopping-distances.html (Accessed: 27 November 2015).

DSDM Consortium (no date) *23. Testing - DSDM CONSORTIUM*. Available at: http://www.dsdm.org/content/23-testing (Accessed: 12 November 2015).

The Eclipse Foundation (no date) *About the eclipse foundation*. Available at: https://eclipse.org/org/ (Accessed: 6 November 2015).

erniejunior (2015) *Erniejunior/UT390B_Arduino_Library*. Available at: https://github.com/erniejunior/UT390B_Arduino_Library (Accessed: 3 April 2016).

Fuller, A. (2013) 'Arduino laser distance meter - solder and flux', *Uncategorized*, 27 July. Available at: http://blog.qartis.com/arduino-laser-distance-meter/ (Accessed: 28 October 2015).

Fuller, A. (2014) *About DORI*. Available at: http://dori.qartis.com/ (Accessed: 28 October 2015).

Google Scholar (no date) *Paul Viola - Google Scholar Citations*. Available at: https://scholar.google.com/citations?user=G2-nFaIAAAAJ (Accessed: 23 October 2015).

Guizzo, E. (2011) 'How Google's self-driving car works', 18 October. Available at: http://spectrum.ieee.org/automaton/robotics/artificial-intelligence/how-google-self-driving-car-works (Accessed: 27 October 2015).

Haar, A. (1910) 'Zur Theorie der orthogonalen Funktionensysteme', Springer Science + Business Media. pp. 331–371.

Hanik, F. (no date) *The kiss principle*. Available at: https://people.apache.org/~fhanik/kiss.html (Accessed: 20 November 2015).

Heeger, D. (2006) *Perception Lecture Notes: Depth, Size, and Shape*. Available at: http://www.cns.nyu.edu/~david/courses/perception/lecturenotes/depth/depth-size.html (Accessed: 22 October 2015).

IDC (no date) *IDC: Smartphone OS market share*. Available at: http://www.idc.com/prodserv/smartphone-os-market-share.jsp (Accessed: 5 November 2015).

JetBrains INC (no date) *IntelliJ IDEA — the most intelligent java IDE*. Available at: https://www.jetbrains.com/idea/ (Accessed: 6 November 2015).

JMW Solicitors LLP (no date) *How speeding, tailgating and unlicensed driving contribute to car accidents*. Available at: http://www.the-claim-solicitors.co.uk/car-accident/car-accident-reckless-driving.htm (Accessed: 27 November 2015).

Laser Technology Inc (no date) *Laser technology - TruPulse laser Rangefinder*. Available at: http://www.lasertech.com/TruPulse-Laser-Rangefinder.aspx (Accessed: 28 October 2015).

Law Offices of Michael Pines, APC (no date) *Top 25 causes of car accidents*. Available at: https://seriousaccidents.com/legal-advice/top-causes-of-car-accidents/ (Accessed: 27 November 2015).

Mahammed, M.A., Melhum, A.I. and Kotchery, F.A. (2013) *Object Distance Measurement by Stereo VISION*. Available at: http://www.warse.org/pdfs/2013/icctesp02.pdf (Accessed: 22 October 2015).

Mathworks (no date) *Vision.CascadeObjectDetector System object*. Available at: http://uk.mathworks.com/help/vision/ref/vision.cascadeobjectdetector-class.html (Accessed: 23 October 2015).

Microsoft (no date) *Kinect - windows app development*. Available at: https://dev.windows.com/en-us/kinect (Accessed: 27 October 2015a).

Microsoft (no date) *Lighting for Kinect | examples of good and bad lighting for Kinect*. Available at: https://support.xbox.com/en-GB/xbox-360/kinect/lighting (Accessed: 27 October 2015b).

Microsoft MSDN (2013) *Test Windows Phone 8.1 Apps with Coded UI Tests*. Available at: https://msdn.microsoft.com/en-us/library/dn747198.aspx (Accessed: 18 November 2015).

Microsoft MSDN (2014) *Getting started with developing for Windows Phone 8*. Available at: https://msdn.microsoft.com/en-us/library/windows/apps/ff402529(v=vs.105).aspx (Accessed: 6 November 2015).

Microsoft MSDN (no date) *Windows.Devices.Bluetooth.Rfcomm namespace - Windows app Development*. Available at: https://msdn.microsoft.com/en-us/library/windows/apps/windows.devices.bluetooth.rfcomm.aspx?f=255&MSPPError=-2147217396 (Accessed: 5 November 2015).

Mrovlje, J. and Vrančić, D. (2008) *Distance measuring based on stereoscopic pictures*. Available at: http://dsc.ijs.si/files/papers/S101%20Mrovlje.pdf (Accessed: 22 October 2015).

Nielsen, J. (2012) *Usability 101: Introduction to usability*. Available at: https://www.nngroup.com/articles/usability-101-introduction-to-usability/ (Accessed: 3 April 2016).

Nieto, M. (2010) *Lane tracking and vehicle tracking (rainy day)*. Available at: https://www.youtube.com/watch?v=JmxDIuCIIcg (Accessed: 23 October 2015).

Nieto, M. (2011) 'About me', *Marcos Nieto's Blog*, 13 November. Available at: https://marcosnietoblog.wordpress.com/about/ (Accessed: 23 October 2015).

PhoneGap (no date) *Supported features*. Available at: http://phonegap.com/about/ (Accessed: 6 November 2015).

Puttemans, S. (2013) 'Max camera resolution for opencv? - OpenCV Q&A Forum', May. Available at: http://answers.opencv.org/question/12944/max-camera-resolution-for-opencv/ (Accessed: 23 October 2015).

Rajput, M. (2015) 'Why Android studio is better for Android developers instead of eclipse - DZone mobile', May. Available at: https://dzone.com/articles/why-android-studio-better (Accessed: 6 November 2015).

Road Safety Foundation (2014) *REPORT EXTRACTS RELATING TO THE RECOMMENDATION FOR INSURANCE PREMIUM TAX RELIEF ON TELEMATICS MOTOR INSURANCE FOR YOUNG DRIVERS*. Available at: http://www.roadsafetyfoundation.org/media/30618/telematics_insurance_business_case.pdf (Accessed: 27 November 2015).

Rouse, M. (2011) *What is stereoscopy (stereoscopic imaging)? - Definition from WhatIs.Com*. Available at: http://whatis.techtarget.com/definition/stereoscopy-stereoscopic-imaging (Accessed: 22 October 2015).

Sims, G. (2014) 'I want to develop Android Apps - what languages should I learn?', *Android Development*, 10 June. Available at: http://www.androidauthority.com/want-develop-android-apps-languages-learn-391008/ (Accessed: 6 November 2015).

Software Testing Help (2015a) *Requirements Traceability matrix - creating process with sample template*. Available at: http://www.softwaretestinghelp.com/requirements-traceability-matrix/ (Accessed: 20 November 2015).

Software Testing Help (2015b) *Types of software testing and definitions of testing terms*. Available at: http://www.softwaretestinghelp.com/types-of-software-testing/ (Accessed: 13 November 2015).

TalTech (2013) *What is RS232 and serial communications?* Available at: http://www.taltech.com/datacollection/articles/serial_intro (Accessed: 5 November 2015).

Taylor, T. (2011) 'Kinect for robotics', 29 November. Available at: http://blogs.msdn.com/b/msroboticsstudio/archive/2011/11/29/kinect-for-robotics.aspx (Accessed: 27 October 2015).

Telematics.com (2015) *Telematics insurance, fleet Telematics, tracking & Infotainment*. Available at: http://www.telematics.com/ (Accessed: 23 November 2015).

TeraRanger (no date) *Time of flight principle, light vs sound*. Available at: http://www.teraranger.com/technology/time-of-flight-principle/ (Accessed: 27 October 2015).

Velodyne (2014) *High Definition Lidar*. Available at: http://velodynelidar.com/lidar/products/brochure/HDL-64E%20Data%20Sheet.pdf (Accessed: 27 October 2015).

Viola, P. and Jones, M. (2004) 'Rapid Object Detection Using a Boosted Cascade of Simple Features', IEEE Computer Society Conference on Computer Vision and Pattern Recognition: . Available at: http://www.merl.com/publications/TR2004-043 (Accessed: 23 October 2015).

Viola, P. and Jones, M.J. (2004) 'Robust Real-Time Face Detection', *International Journal of Computer Vision*, 57(2), pp. 137–154. doi: 10.1023/B:VISI.0000013087.49260.fb.

Volkswagen UK (no date) *Adaptive cruise control (ACC): Volkswagen UK*. Available at: http://www.volkswagen.co.uk/technology/adaptive-cruise-control-acc (Accessed: 27 November 2015).

westfw (2014) 'structures and object orientated help', *Arduino Forum > Using Arduino > Programming Questions > structures and object oriented programming help*, 6 January. Available at: http://forum.arduino.cc/index.php?topic=208779.0 (Accessed: 12 November 2015).

Woodford, C. (2015) 'How do lasers work? | who invented the laser?', 15 May. Available at: http://www.explainthatstuff.com/lasers.html (Accessed: 27 October 2015).

Wyse, S.E. (2011) *Difference between qualitative research vs. Quantitative research*. Available at: http://www.snapsurveys.com/blog/what-is-the-difference-between-qualitative-research-and-quantitative-research/ (Accessed: 16 April 2016).

Zant, C. (2013) ‘How do Rangefinders work?’, *Optics*, 29 October. Available at: http://precisionrifleblog.com/2013/10/29/how-do-rangefinders-work/ (Accessed: 27 October 2015).

Zeng, W. (2012) ‘Microsoft Kinect Sensor and Its Effect’, *Multimedia at Work*, .