School of Computer Science and Informatics
Cardiff University

Drone Transportation Protocol for High Traffic Areas

Author: Nathan Ahmad – C1248941

BSc Computer Science with Year in Industry

Supervisor: Stuart Allen

Moderator: George Theodorakopoulos

6th May, 2016

Table of Contents

Abstract

This project concerns developing an understanding of what is required to design a Drone Transportation Protocol that is autonomous, safe and efficient.

To achieve this goal, three main objectives have been set:

- Objective 1: Develop a Virtual Testing Platform that provides a means to test and validate different Drone Transportation Protocols. This will be used to understand how different route planning methods and evasive actions effects the efficiency of different Drone Transportation Protocols. The information gathered from the Virtual Testing Platform will be used to guide the development of Objectives 2 and 3.
- Objective 2: Develop a Communication Protocol that enables drones to be able to detect each other, share information about their position and direction, and then determine the appropriate avoidance action that should be taken.
- Objective 3: Develop Drone Control code that could execute on a physical drone. The Drone Control takes its direction from the initially planned route and the feedback from the Communication Protocol. The design of the Drone Control is informed by the performance metrics gathered from the Virtual Testing Platform.

Table of Figures

Table of Tables

1 Introduction

Drones, commonly referred to as UAV (Unmanned Aerial Vehicles) or RPAS (Remotely Piloted Aerial Systems), are aircraft that do not accommodate an on-board human pilot. Early drones were prohibitively expensive and provided no autonomous control, therefore requiring a remote human pilot. As such, their main usage was to be found in military applications, such as surveillance and defence. In recent years the price of drones has plummeted and technology has continued to get more sophisticated. As a result of this, owning a drone is now possible for the average consumer, and although the majority of drones still require a (remote) human pilot, it is not uncommon for consumer models to have semi-autonomous control, such as auto take-off and landing, or the ability to follow a mobile GPS device.

Figure 1 – A selection of consumer level drones

However, the growth of drone ownership has brought with it issues. Drones operate using propellers and can reach speeds of up to 55mph, which means they are very capable of causing damage to property or risk to human life. Unlike the automotive industry, the regulations governing the usage of drones is general and often do not reflect modern drone usage, and the infrastructure to monitor and license ownership is virtually non-existent. This has led to many reported cases where untrained or irresponsible drone owners have caused serious incidents. The most serious reported incident has been of a drone colliding with a British Airways passenger Airbus A320 plane carrying 132 passengers as it came into land at Heathrow [1]. The Civil Aviation Authority (CAA) says that in 2014 there were 9 reported near misses between drones and aircraft, but this rose to 40 in 2015 [2].

If left unaddressed, the issues caused by untrained and irresponsible drone owners will only continue to become increasingly common. There are currently calls for the government to tighten rules on drone ownership in the UK, and in the United States there is a Federal Aviation Administration (FAA) sponsored awareness campaign for recreational and business users of drones [3]. However, an innovative and complimentary solution to these would be to automate the control and operation of drones. This solution would remove the responsibility from untrained drone owners onto the automated systems that govern the operation of the drones, which ultimately lies with the people or organisations that develop these autonomous drone control systems. This would have the benefit of making drone activity safer (autonomous systems could be built with safety being a primary specification), drone licensing cheaper and easier (autonomous systems would be tested, validated and licensed rather than all the thousands of individuals that use the systems) and rapidly expand the potential advantages that drones could add to civil and business life.

Current regulations in the UK dictate that drones must be kept within Line-of-Sight (LoS), which is taken to mean 500 metres horizontally [4] (A more detailed examination of UK regulations will be included in Section § 2.1.1). Autonomous control of drones would remove this requirement, meaning that drones could be used safely across a whole city area. This would allow drones to carry out tasks such as delivery [5], emergency first response or even network infrastructure [6], which would in turn have the effect of reducing road traffic, increasing public safety and improving the quality of public services. Autonomous drone operation is an active area of research for many companies including Amazon, Facebook and Google, which will likely produce innovative and far-reaching uses for this technology. As drones continue to get cheaper and more capable, and autonomous drone control is combined with machine learning, the potential for growth is truly phenomenal and could lead to an unprecedented increase in industrial productivity.

The industry-wide activity surrounding drones makes it a particularly interesting subject of study at the moment. This, combined with the outdated regulatory framework surrounding drones (autonomous drones are not currently recognised by the CAA), make this area of study not only interesting, but important. More work needs to be carried out to determine how autonomous drone operation can be safely achieved, and conversations need to be started about how drones should be regulated and to what level we should allow drones to play a part in our society.

1.1 Project Overview

The proposal of this project is to understand the key factors and considerations required when implementing a Drone Transportation Protocol (DTP) that is optimised for high traffic areas. By exploring how different protocols compare against each other, a core aim of the project is to develop an informed recommendation for a safe, efficient and autonomous DTP. The implemented protocol facilitates collaborative Sense and Avoid (SAA) technology, which requires Vehicle-to-Vehicle (V2V) wireless communication between drones that are in flight.

When creating a DTP, there are three aspects that need to be developed:

- A Virtual Testing Platform, to allow testing and quantifying the safety and efficiency of different protocols in a safe environment.
- A Communication Protocol, the wireless communication between multiple drones to allow for safe, autonomous movement.
- The Drone Control, code that operates the actual movement of a drone, utilising feedback from the Communication Protocol when necessary.

In this project, these areas are explored and brought together to create a proof of concept for what a DTP could achieve and how to approach the task of creating this type of technology.

1.2 Project Goals

Throughout this project it is vital to build an understanding of what constitutes a safe and efficient autonomous Drone Transportation Protocol. Using this understanding, the goal is to deliver a partially implemented DTP, an implemented Virtual Testing Platform and a Communication Protocol.

To achieve this, it is important to follow the three main objectives which were set out in the initial project plan [Appendix A]. Each objective has primary goals and secondary goals. Primary goals should be considered as core requirements for the project, whereas secondary goals are included for consideration and explored where possible throughout the project. Below is a recap of these initial objectives set out in the Initial Project Plan.

1.2.1 Project Objectives

Objective 1: Virtual Testing Platform

Develop a Virtual Drone Testing platform, that will allow testing and validation of different transportation protocols.

- Primary Aims:
  - Develop a simplified model that can gather performance data on different implementations of Drone Transport Protocols.
  - Develop a model that can test the communication protocol against failure.
- Secondary Aims:
  - Demonstrate the implemented drone protocol running within the virtualised testing platform. i.e. the testing platform will be able to emulate the code written to run on the drone.

Objective 2: Communication Protocol

Develop a Communication protocol that will enable autonomous collision pre-emption. This protocol will be part of the larger DTP.

- Primary Aims:
  - Identify a viable technology for wireless communication between drones which has suitable characteristics, such as acceptable range of communication and power consumption.

  - Implement a communication protocol that determines appropriate action required for Vehicle-to-Vehicle Sense-and-Avoid collision pre-emption and does not result in deadlock.

- Secondary Aims:
    - The protocol should be able to resolve situations when it encounters drones running unknown protocols, while considering the correct level of 'politeness' for the protocol to adopt.
    - Explore safeguards that can be taken against malicious drones that attempt to obstruct or redirect the drone's journey in some way.
    - Appropriately respond to 'emergency service' requests.

Objective 3: Drone Control

Develop the code that controls the physical movement of the drone, which adjusts the drone's route based on feedback from the Communication Protocol when necessary.

- Primary Aims:
    - Identify safe operating limits, such as maximum speed and minimum proximity of drones.
    - Implement non-cooperative route planning techniques that utilise game theory to minimise collisions of single destination drones (i.e. drones that start at point A and finish their journey at point B).
    - Implement a program that uses a planned route and feedback from the Communication Protocol to control and direct a drone's physical movements.
- Secondary Aims:
    - Facilitate multi-destination drones and non-linear journeys (e.g. the journey of a surveying drone).

1.2.2 Scope

The creation of a DTP is a broad subject area that could be taken to a deep degree of complexity. As previously mentioned, some of the most well equipped technology companies are currently working on complete solutions to this problem in specific use case areas. Due to this, it is important to limit the scope of this project to ensure that there is focus and clarity on the core problems being solved.

- The project only concerns drones travelling between fixed points, such as the movement of a delivery drone. It does not concern drones following moving points, such a surveillance drone following a moving vehicle, or drones carry out tasks, such as agricultural crop spraying. The reason that travelling between fixed points is the chosen application of the DTP in this project is because it provides a basic function of how autonomous drones can operate, on-top of which more complex uses could be built.
- In the absence of British guidelines governing the use and limitations of autonomous drones, this project will be based on the model suggested by Amazon. Amazon's drone regulation guidelines will be detailed in section § 2.2.1.
- The project did not explore any non-collaborative Sense and Avoid technology, which would allow drones to avoid non-communicative objects, such as birds, buildings or traditional aerial vehicles (hot air balloons, planes etc.). The reason that this decision was taken was to contain the problem within drone operations, as Non-

Collaborative Sense and Avoid technology would require a huge amount of visual computing, which would require a large project in its own right.

- A demonstrable physical drone is not a deliverable for this project. The project will be based entirely around software implementations and virtual demonstrations. The reason for this is because testing and validating the movements of many drones across a large area is infeasible.

1.3 Approach

The approach to this project will be to split it into its three main parts: Control; Communication; and a Virtual Testing Platform. By splitting the project into its core components it is easier to define the problems that they pose and the tasks required to solve them.

Where possible, existing open source software that can help in achieving the project goals will be utilised. Using open source code is a common-practice way to quickly build software with complex tools, rather than spending time rebuilding existing solutions. Open source software is often used in professional environments as it allows organisation to benefit from the knowledge and work of others, and often allows individuals to become experts in that open source software and contribute improvements back to the community.

1.3.1 Open Source software

3D Robotics is a company that develops drone hardware and software for consumer and commercial use. They also develop and maintain the DroneKit [7] suite of open source drone tools.

DroneKit includes:

- DroneKit-Python, a toolset for easily programming a Drone Flight Controller using Python.
- DroneKit-SITL, a simulation platform designed to work with DroneKit-Python code.
- DroneKit-Cloud, a live telemetry storage solution that can be accessed with a RESTful API.
- DroneKit-Android, for developing Android apps that can communicate with and control drones.

This project makes extensive use of DroneKit-Python and DroneKit-SITL.

DroneKit-Python

DroneKit-Python [8] is an open source Python-based toolset that provides an SDK for developing applications for drones. DroneKit-Python provides an abstracted way for an on-board computer, such as a Raspberry Pi, to connect to the flight controller and send movement instructions. This can be used to provide reactive instruction to the drone mid-flight based on input from other components, such as on-board Bluetooth.

Within this project, DroneKit-Python will be utilised when creating the Control aspect of the DTP. The benefit of using this is that it allows for a simple, robust and well documented way to program the movements of the drone. Programming directly onto a Drone Flight Controller is more complex and requires a higher level of knowledge, so using DroneKit's tools makes it easier to implement the project and easier for others to understand and utilise the code from with project.

DroneKit-SITL

DroneKit also provides an integrated drone simulation platform, DroneKit-SITL [9] (DroneKit Software in the Loop). This platform provides a way for DroneKit-Python programs to easily integrate with ArduPilot's Software-in-the-Loop [10] drone simulation platform. DroneKit-SITL creates a very detailed virtual environment with simulated virtual drones, and it provides a GUI to make it easy to track the simulated drones.
This project explores DroneKit-SITL as a virtual testing platform and examines its benefits and weaknesses. This will be covered in detail in Section §2.3.

1.3.2 Major steps

Below are the main steps that were undertaken in the approach to completing each section of the project.

Drone Control

DroneKit-Python was used when programming the Drone Control software, as it provides a high level Python interface for programming the movement of the drone. This will drastically simplify this part of the project as it reduces the minimum knowledge required for creating bespoke drone software.

The Drone Control part of the project consists of:

- Code that executes non-collaborative route planning.
- Code that incorporates the Communication Protocol into a cohesive system.
- Code that controls the movement of each drone, based on the initial planned route or input from communication with another drone.

The major steps that were required in achieving these are:

- Identifying the safe operating limits of a drone.
- Implementing a basic route planning method. This was used (in the virtual testing platform) as the base benchmark to see how other route planning methods could be improved.
    - Iterating on this step by planning, implementing and testing new route planning methods.
- Implementing the execution step: Controlling the drone from start to finish (in the virtual testing platform).
    - This required the Communication Protocol, to allow for avoidance action to be undertaken.
- Planning basic avoidance actions that are required when receiving input from the Communication Protocol.

Drone Communication

The Drone Communication part of the project consists of:
- The software that will allow drones to communicate with each other mid-flight.
- Identifying the type of SAA scenario that has been encountered.

The major steps that were required in achieving these are:
- Comparing wireless communication technologies to determine which is the most suitable for this project.
- Identifying all possible flight scenarios where SAA preventative action will be required.
- Identifying the live state information that drones are required to communicate to take effective preventative action.
- Designing the communication interaction that will take place in an SAA scenario.
- Implementing the Communication Protocol.

Virtual Testing Platform

The intention of this project was to use DroneKit-SITL's simulation platform to test how different protocols compare against each other. However, it turned out that there is a limitation of only simulating a single instance of a drone using DroneKit-SITL. This report will detail the findings and usage of DroneKit-SITL in more detail and the actions that were undertaken to resolve this in the Implementation Section § 4.1.1.
In the Initial Report, a backup plan was devised for the scenario that DroneKit-SITL would not be suitable for the VTP. The backup plan consisted of implementing a standalone simulation platform, which would be far more rudimental but still provide a means to gather the core metrics required to develop an informed understanding of the different protocols.

The Virtual Testing Platform for this project consists of:
- Creating a virtual environment where it will be possible to simulate different models of drones. Drones differ by their methods of route planning and SAA avoidance.
- Designing the virtual environment to carry out several tests, which are used to quantify the performance of different drone's attributes.

The major steps that were required in achieving these are:
- Experimenting to find the limitations of DroneKit-SITL.
- Implementing a separate virtual drone flight simulation environment.
    - Note that the newly implemented virtual environment does not provide a means for working drone code to be tested in it. The drone software that was designed and tested in the virtual environment needed to be re-written in Python to integrate with the DroneKit-Python framework.
- Designing and implementing tests that can determine when drones collide (in the virtual environment), and testing the performance of drones.

1.4 Summary of Outcomes

On completion of this project the deliverables include a Virtual Testing Platform, data that supports a recommendation for a safe and efficient DTP, and a partially implemented DTP that demonstrates the Communication Protocol in action, but would be unsuitable for fully operating a drone autonomously.
The DTP consists of the wireless Communication Protocol that enables drones to collaboratively Sense and Avoid each other during flight, and a basic level of Drone Control, that is utilised to demonstrate the Communication Protocol.
To measure the performance of the DTPs, the Virtual Testing Platform was developed which was used to run performance tests on various DTP designs to determine their strengths and weaknesses. The DTPs that were implemented in the Virtual Testing Platform carry out initial route planning and the control of the drones when SAA communications are received.

This project presented several key challenges, which required the goals to be reassessed and the approach to be adjusted. This report will examine these challenges in detail in the Implementation Section § 4. Some of the larger challenges include the complexities and limitation of incorporating open source software into this project, particularly regarding the Virtual Testing Platform. The depth of the problem being tackled here occasionally caused challenges, as it was always easy to see new and interesting avenues that could have been explored when designing a DTP, but that were not possible to pursue due to time constraints. Creating a system that was entirely software based and required no human interaction posed the challenge of how to appropriately demonstrate the work achieved in a comprehensive and compelling form.

The conclusion of this project is to make an informed recommendation for a safe, efficient and autonomous DTP. Detailing the steps that have been undertaken to gain this understanding will allow others to learn from this knowledge and hopefully utilise it in future projects of their own.

2 Background

Before delving into this project, it is beneficial to understand what the current landscape of drone usage looks like in the UK: How are drones currently used? What restrictions are in place? How is the wider industry approaching this problem and how will drone usage change in the future? In this section, the report will examine what assumptions will be made throughout the project regarding operational and legal limitations that will be imposed on the drones and the DTP.

Having this understanding will help to see how the approach taken in this project fits into the wider context of creating an autonomous DTP that is suitable for use in the UK, and why taking on this challenge is relevant and important.

2.1 Current State of the Drone Industry

Private drone ownership is growing in the UK, in thanks to the many competing drone manufacturers (including 3D Robotics, Parrot, DJI and others) which are creating a wide selection of consumer drones at increasingly low prices. It is now possible to purchase a very simple remote controlled drone from as little as £20 [11], and sophisticated drones with high quality features (such as semi-autonomous control, HD camera rigs and long battery life) for under £1000 [12]. Drones are most commonly used for recreational entertainment, but have also been used for photography, film-making and even amateur drone racing.

Organisations are starting to utilise drones in industrial settings, but these are most commonly used over private land. A good example of this is the agricultural industry, where drones played a large role at Cereals 2015, one of the UK's biggest annual agricultural events [13].

2.1.1 Current Regulations and Licensing

The Civil Aviation Authority (CAA) is the governing body which oversees UK flight regulation. The current regulations for drones under 20kg are covered in Articles 166 and 167 of the CAA [14].

They are summarised on the CAA's website as follows:

1. The operation must not endanger anyone or anything.
2. The aircraft must be kept within the visual line of sight (normally taken to be within 500 m horizontally and 400 ft. vertically) of its remote pilot (i.e. the 'person in charge' of it). Operations beyond these distances must be approved by the CAA (the basic premise being for the operator to prove that he/she can do this safely).
3. Small unmanned aircraft (irrespective of their mass) that are being used for surveillance purposes are subject to tighter restrictions with regard to the minimum distances that you can fly near people or properties that are not under your control. If you wish to fly within these minima, permission is required from the CAA before operations are commenced.

4. CAA permission is also required for all flights that are being conducted for aerial work (i.e. in very simple terms, you are getting paid for doing it).
5. The 'remote pilot' has the responsibility for satisfying him/herself that the flight can be conducted safely.

If these regulations are broken, then the operator can be liable to face criminal charges and a fine.

These regulations cover drone usage in the most basic manner (all rules being a variation on common sense) but they are no longer fit for purpose. Point 2 states that there would need to be a case-by-case review of any drone use outside of line of sight, which is a wholly unscalable solution. The regulations do not acknowledge that autonomous drone operation occurs at all, not to mention providing a legal framework for large numbers of autonomous drones to adhere to when operating in the same airspace. Another omission is any formal guidelines for conducting ‘aerial work’. Point 4 states that permission needs to be acquired under any circumstances that someone is getting paid to operate drones. This regulation is not suitable for a future where autonomous drones could be used to carry useful service tasks.

Due to the unsuitability of the current regulation, the project will assume that in the UK, Amazon’s suggested regulations and guidelines for drone operation are in place. These address autonomous control of drones and the possibility of drones safely sharing airspace. Amazon’s regulations are covered in detail in Section 2.2.1.

2.1.2 Future Drone Usage

The future of drone usage will likely have many varied applications. Some of the broader potential uses for drones can be forecast by the research being carried out by Facebook and Google in an effort to use drones to connect remote areas to the internet. Also, the first Drone Prix racing contest was held in Dubai in 2016 [15], this is expected to become an annual event.

Regarding drone usage that is more closely related to this project, several organisations are currently researching the area of autonomous drone operation in high traffic areas. Most notably, Amazon envisions that in the near future they will be able to use autonomous drones to deliver small packages, as can be seen in their promotional video [16].
This raises the question of how the problem being solved in this project fits into the future of drone, and why this project is being taken on when large organisations are working to solve it.

Motivations for this project and how it fits into future drone usage

Although Amazon and other organisation are working on solving the problem of autonomously controlling large numbers of drones, there is currently no ‘off-the-shelf’ solution to this problem. It is also incredibly unlikely that when these organisations do create a complete drone control system, that they will then sell or license it publicly. It is far more likely that this technology will be kept as proprietary software to serve their business needs, and therefore the details of their approach to this problem will remain unknown.

By tackling this project in an open manner, this technology and the approach taken to solve this problem will be available to be used by others to help guide them in solving this and other similar problems. In a broader sense, there is a hope that this project will contribute to the wider conversation regarding how autonomous drones should be incorporated into our society, and the wider effects this could have on socio-economic issues.

2.2 Assumptions

Autonomous Drone Transportation is a relatively new subject and there are no complete regulations that allow drones to reach their full autonomous potential, or guidelines that recommend how organisations use drones in a safe, fair and ethical manner.
Although the limits of what is consider safe are changing all the time, it is important to define what the current boundary of drone safety means in the context of this project, in regards to both the operational limits of drones and the minimum specification requirements. It is important to make these assumptions so that there is a common expectation of the standards that are in place throughout this project.

2.2.1 Legal requirements of Autonomous Drone Transportation

Amazon have released proposed regulations to guide the standards of autonomous drone operation. These guidelines currently hold no legal standing, but they provide a template that Amazon believes government regulators should follow. These guidelines are being used in this project because they represent the most comprehensive attempt to address suitable regulation for autonomous drone transportation. Amazon's proposal is covered in *"Determining Safe Access with a Best-Equipped, Best-Served Model for Small Unmanned Aircraft Systems"* [Appendix B] and *"Revising the Airspace Model for the Safe Integration of Small Unmanned Aircraft Systems"* [Appendix C].

The proposal covers two main areas:

1. How vertical airspace should be treated to enable the safest possible drone operation.
2. Different levels of drone classification which state the operational limits applied to drones in relation to the area type the drones are flying above.

Airspace Classification

Below is Amazon’s diagram detailing their suggested airspace design:

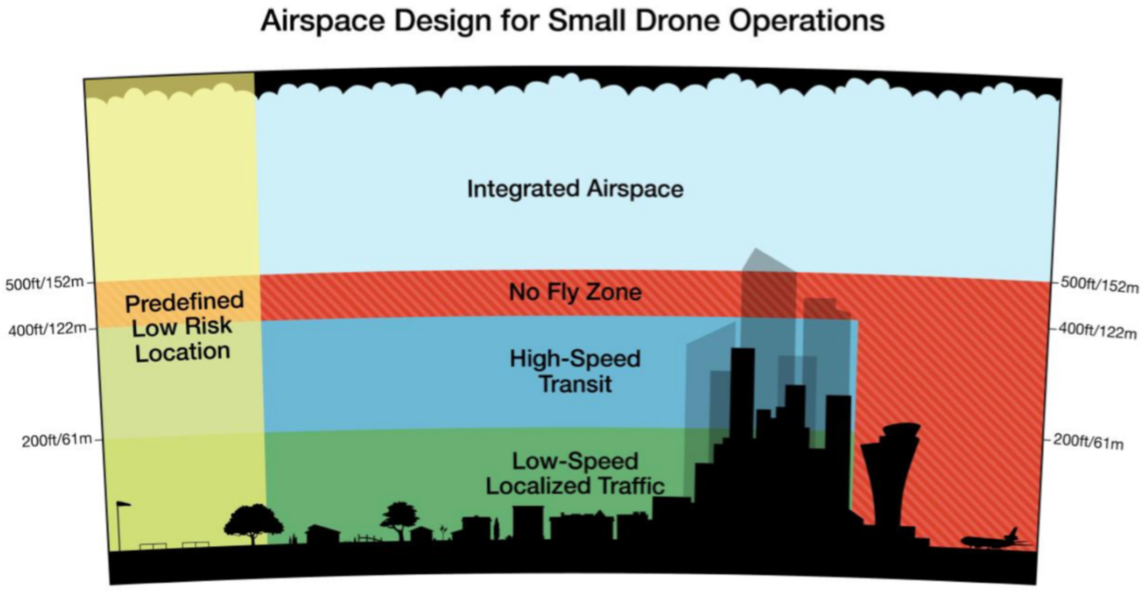


Figure 2 - Amazon's recommendation of how airspace should be segregated to accommodate different aircraft

Amazon suggest segregating airspace based on vertical height and ground area type:

- Height:
    - **Ground level to 200 feet** is reserved for Low-Speed localised traffic.
        - Lesser-equipped drones and non-transit operations (such as surveying or video/photography operations) are permitted in this airspace.
    - **200 feet to 400 feet** is reserved for High-Speed transit.
        - Well-equipped drones are permitted in this airspace.
    - **400+ feet** is a No Fly Zone for drones.
- Ground Area Type:
    - **Restricted Areas**, such as airports or government space, will not allow drone operation.
    - **Predefined Low Risk Zones**.
        - Areas where special restrictions are established in advance. These areas may be used for activities such as drone research or drone recreation.
    - **Urban, Suburban and Rural Areas** affect which classes of drones can operate in that area and the limits of their operations.

Drone Classification

Amazon suggest classifying drones into four categories based on their ability to automatically Sense and Avoid (SAA) obstacles:

- **Basic**
    - Radio controlled drones.
    - Zero SAA technology.

- **Good**
    - Ability to announce identity, location and activity to other drones via Vehicle-to-Vehicle (V2V) communication.
    - Ability to alert operator when manual action is needed.
        - Sense-only technology.
- **Better**
    - Autonomous control.
    - Capable of collaborative SAA via V2V communication.
        - Ability to autonomously avoid other drones and 'smart-enabled' obstacles.
- **Best**
    - Autonomous control.
    - Capable of collaborative SAA.
    - Capable of non-collaborative SAA.
        - Ability to autonomously avoid any other obstacles, e.g. birds.

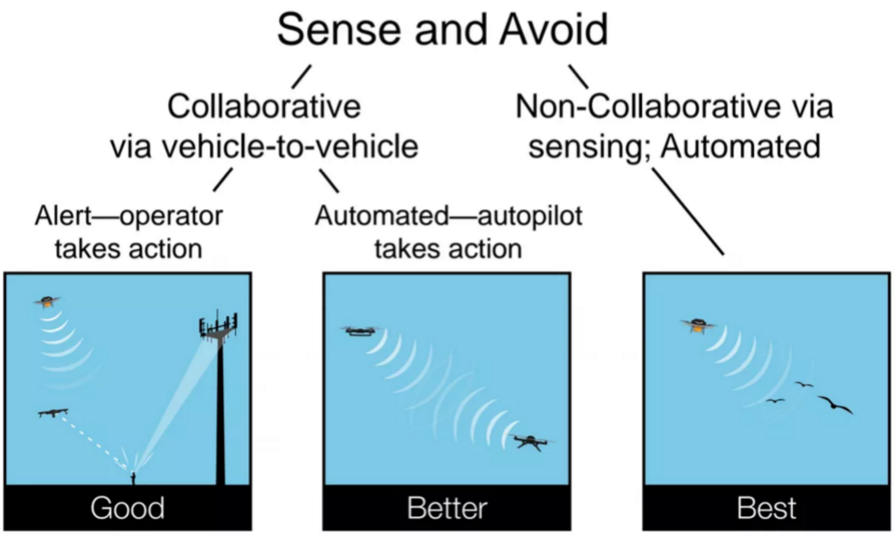

Sense and Avoid
Collaborative via vehicle-to-vehicle
Non-Collaborative via sensing; Automated
Alert—operator takes action
Automated—autopilot takes action
Good
Better
Best

Figure 3 - Different drone classification based on their SAA capability

Deviations from Amazon's guidelines

Amazon has suggested comprehensive limitations on which classification of drone is permitted to operate over different ground area types, however, the guidelines in this project will deviate from these in one major point. Amazon recommends that only the 'Best' class of drone should be able to operate completely independently of an operator (The 'Better' class must be within line of sight of an operator who could take control if needed).

For this project the assumption will be made that the 'Better' classification of drone is safe to operate in urban areas up to 400 feet outside of the line of sight of the operator.
The reason this decision is made it because creating a DTP that would fit into the 'Best' class would require non-collaborative SAA technology, which is out of scope for this project.

2.2.2 Ethical Issues

It is important to consider any ethical issues that surround developing an autonomous DTP. These issues are covered in detail in the Initial Report [Appendix A]. Below is an overview of the main concerns:

- Legal
    - The legal frameworks that govern the use of drones in the UK should provide fair and safe practices to follow, while not impeding the development of new drone technology without reasonable concern.
- Safety
    - Safety to human life and property should be seen as primary specification of any drone control system.
- Privacy
    - Drones have the capability of collecting data (and often require the collection of image data to operate effectively). The same regulations should apply to this data as any other collected data in the UK at the moment. It is important that only the minimum amount of data that is required is collected and that sensitive data is not stored long-term without a valid reason and the appropriate permissions.
- Licencing
    - Like Net Neutrality, the licencing of autonomous drone technology should be open to all and should not be swayed in favour of big businesses or private individuals.
- Jobs
    - Autonomous drones could have a large impact on job distribution and wealth creating. This is a deep and complex ethical issue with unclear socio-economic repercussions. Any organisation building autonomous DTPs should strongly consider what effects their software could have. They should also take on the responsibility of operating transparently and raising awareness of the effects of job automation and engage in the wider conversation about the effects this could have. The Atlantic article, *'A World Without* Work' [17], contains an interesting discussion of this topic.

2.2.3 Operational Limits

It is important to define the safe operational limits that are applied to drones flying in high traffic areas. These limitations need to be realistic (i.e. within the limitations of current drone technology), and find the right balance between restricting the capability of the drones and ensuring that they have a high enough technical ability to deal with any scenarios that they may encounter.

Specification of Drone Model

Throughout this project the assumed model of drone that this DTP is design to run on is 3D Robotics' Solo drone. The reason that this model of drone has been chosen is because it is a well-equipped model (it has feature such as GPS and a second on-board computer that is separate from the Flight Controller, which reduces the likelihood of system failure during flight) and it is supported by DroneKit, which is used as a core component of the implementation of the DTP. The Solo should be considered to have the minimum required specifications for any drone using the DTP in this project.

The full specification of the Solo can be seen on 3D Robotics website [18]. The key information is detailed below.

Flight time: 25 mins (20 mins with payload)
Max payload: 420 g
Max speed: 89kph (25 m/s)
Max ascent/decent speed: 10 m/s in stabilize mode
Weight 1.5 kg

Operational Restrictions of DTP

To define safe operating restrictions for the DTP first the expected uses must be defined. For this project the assumption is made that the DTP is designed to work running on drones in the city of Cardiff. Cardiff has been chosen because it represents a reasonable sized metropolitan area within the UK.
The Cardiff urban area is approximately 75 km^2 and has a maximum diameter of close to 15 km. This information was gathered using Google Earth.

Speed

The worst case drone journey that could occur in Cardiff will be used as the basis for deciding the operational restrictions. Consider a journey with the origin and destination locations at their furthest points in Cardiff where the drone is required to return to its original location and it travels at the maximum flying height of 400ft(152m). This journey would consist of the following steps:

120 metre ascent.
15 km journey to destination.
120 metre decent.
[Carry out task, such as dropping off package]
120 metre ascent.
15 km journey to origin.
120 metre decent.
[End journey]

The journey time would be approximately 21 minutes. This is calculated as:

480 metres of ascent/decent at 10 m/s takes < 1 minute
30 km horizontal travel at 25 m/s takes 20 minutes

Using these calculations, a drone operating at its maximum speed could travel between any two destinations in Cardiff within the drone battery life time span. Therefore, the maximum capacity of the drones will be the operational speed limits (25 m/s horizontally and 10 m/s vertically).

Proximity

Drone can safely operate within close proximity (< 1 metre) to each other when moving in a predictable manner [19]. In this project the area that is being considered is a city space, which is very large. Therefore, space is not at a premium and the proximity limitation can be very generous.

In the confines of this project, the safe operating proximity of two or more drones will be 5 metres or more.

The 'Communication Range - Processing Time – Speed' Triangle

The wireless communication range, processing time and speed of a drone are inextricably linked, as shown in this thought experiment.

Two drones are travelling towards each other at velocity *v* and they must sense each other and react before they get within 10 metres of each other. The distance between each drones' original position and the position that is 10 metres from the other drone shall be denoted with *x*, and the time it takes shall be denoted with Δt.

This can be displayed by the following diagram:

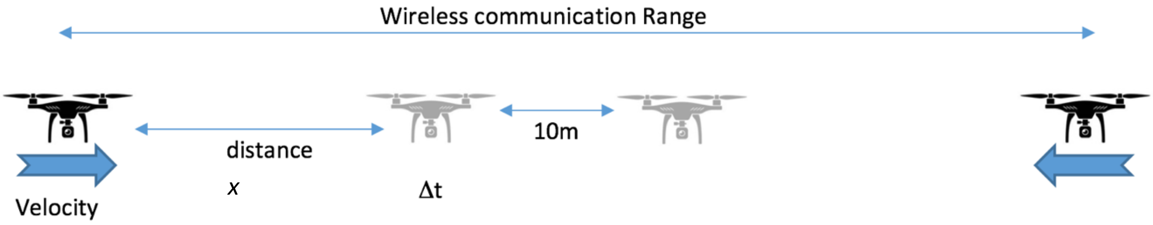


Figure 4 - Diagram showing justification for wireless communication range

The relationship between speed, range and processing time can be written as:

$$Range = Speed \text{ x } Processing\ Time$$

Range, the distance that the wireless signal can be sent, is measured in metres. It can also be written as $2x + 10$, where x is as stated above.

Speed, the speed the drone is moving, is measured in metres per second.

Processing Time, the time taken for a drone to detect and react to another drone, is measured in seconds. It must be equal or less than Δt to ensure that drones do not get too close to each other.

As can be seen in the equation, if any value changes it effects the other values in the equation. This equation will be utilised when selecting an appropriate wireless communication technology in Section § 3.3.2.

2.3 DroneKit

A core part of this project is built using the DroneKit open source toolset. For this reason, it is important to detail what DroneKit consists of, the benefits it provides to the project and the limitations it has.

The full documentation of DroneKit can be found here: http://dronekit.io/

DroneKit builds on existing open source systems to provide a simple Python interface for interacting with drones without requiring the knowledge of the other systems.

DroneKit utilises:

- ArduPilot http://ardupilot.org/ardupilot/index.html
    - ArduPilot is the flight control software that runs on PixHawk and APM hardware (these are the two most common drone control computers). As DroneKit is built on ArduPilot it means that it already works with the majority of drones available today.
- MAVLink http://qgroundcontrol.org/mavlink/start
    - MAVLink stands for Micro Air Vehicle Communication Protocol. This open source software allows drones to remotely connect to ground stations which can be used to provide flight instructions.
- Software in the Loop (SITL) http://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html
    - Software in the Loop is a virtual platform created by ArduPilot to test code that has been written to control a drone in a virtual testing environment. It is a complex system which allows users to simulate drone flights in very high detail, even being able to simulate things such as wind speed and direction.

DroneKit has been used in this project because it is a very robust multi-platform solution. DroneKit is actively being developed by 3DRobotics (3DR) which makes it a safe choice as new features and fixes are regularly issued when bugs are detected. 3DR are aware that DroneKit is a widely used platform and are therefore cautious of making backwards-breaking changes.
DroneKit works on a wide range of hardware, so it is a good choice as it means the code produced in this project will be compatible with that hardware. The compatibility of DroneKit can be seen here: http://python.dronekit.io/about/overview.html#compatibility

A key limitation of DroneKit is that the DroneKit-SITL package only supports simulating a single instance of a drone. This issue was discovered during the project and the report examines the solution taken to resolving this in the Implementation Section § 4.1.1.

2.4 Tracking Project Progress

Throughout this project the work carried out has been tracked in a transparent way so that it is possible for anyone to see the progress and contributions that have been made.

Trello
Trello has been used to keep track of the backlog of tasks. This is a common technique used in agile development methodologies which makes it easy to see the tasks that are currently being worked on. Most importantly, it provides an easy way to select and focus on new tasks when choosing the next area to work on.
The Trello board can be found at the following link: https://trello.com/b/x1KTZVBE

GitHub

A public repository of all the code created for this project is being hosted on GitHub. GitHub provides a fantastic way to share open source code with the community and it is a very popular service used by large and small teams. GitHub makes it easy for others to examine and contribute to the code if they wish.

GitHub also provides a very easy way to distribute code between development environments. This has been particularly useful for editing code in a GUI environment in Mac OS X, and then pulling it onto Raspberry Pis that do not have the Gnome desktop interface installed.

The GitHub repository can be found at the following link:
https://github.com/NathanAhmad/DroneTransportationProtocol.git

OneNote

OneNote has been used as a digital notebook to keep all kinds of notes during meetings, information lectures and planning sessions. OneNote's versatility makes it very suitable for keeping a range of notes, as it is easy to create text, task lists, annotatable images and other forms of media. It also has a reliable search function which makes it easy to find previously created notes.

The OneNote notebook can be found at the following link: Final Year Dissertation (Web view https://onedrive.live.com/edit.aspx/OneNote/Final Year Dissertation?cid=4fe2163107cc80d6&id=documents)

2.5 Research Questions

Aim:

The aim of this project is to understand the requirements of creating a DTP and be able to recommend a protocol that would provide a safe, autonomous and efficient drone transportation protocol that it suitable for a high traffic area.

Research Questions:

In order to achieve this goal, it is important to define appropriate performance metrics for the different DTPs. Having appropriately defined performance metrics makes it possible to define how different DTP compare to one another.

To define and test the performance metrics of DTPs it is necessary to design and implement a Virtual Testing Platform that is able to simulate drones running different protocols.

This project brings together a testing platform and examples of different DTP into a robust software package. Using this software package, it is possible to identify a safe and efficient DTP.

3 Specification and Design

The Specification and Design section of this report will discuss any decisions that have been made during the design process and justify why these decisions were made. Ultimately, the aim of this section is to provide a clear plan of how this project has been designed and how the completion of this project was planned to be executed.

The overall goal of this project is to be able to make an informed recommendation on the key factors that would be required for a safe, autonomous and efficient DTP. From this information a partially working proof-of-concept DTP that incorporates both Drone Control and a Communication Protocol has been developed.
To achieve this goal, three core components must be developed: creating the **Virtual Testing Platform** (VTP), and the DTP that consists of **Drone Control** and a **Communication Protocol**.

Using the Objectives from Section 1.2.1, a MoSCoW analysis has been carried out for each of the three major components to determine their business requirements. A MoSCoW analysis defines the requirements and clearly prioritises them by splitting them into four categories: Must have, Should have, Could have and Won't have.
Defining the requirements of the system helps to focus the direction of the project, track the progress and measure the success of the outcome.

3.1 System Overview

3.1.1 Major Components

Virtual Testing Platform

The VTP is responsible for simulating how different DTPs function when hundreds of drones running them are operating in the same space. It is used to generate performance metrics of the different DTPs in a safe virtual environment. The information generated was then used to guide the design of a feasible Drone Transportation Protocol.

Throughout this report, the term 'Drone Transportation Protocol' (abbreviated to DTP) can be used to describe two things. The DTP can be used to describe the complete package of code that runs on a drone and provides the Communication Protocol and Drone Control functionality. It can also refer to the code that was written specifically to function in the revised (Java-based) VTP. This will be noted as 'test Drone Transportation Protocols' (or tDTP) when the distinction is necessary. Test DTPs are written in Java and would not function at all on a physical drone. There purposes is to emulate the design choices that could be applied to DTP in a way that is possible to run and test in the revised VTP.

Initially, the VTP was planned to run and test DTPs, but as the design changed to being a Java-based application, the need for tDTPs arose.

Communication Protocol

The Communication Protocol is responsible for detecting other drones and determining the position of those drones in relation to its own. It then hands this information over to the Drone Control component which uses it to determine what action to take.

The Communication Protocol makes up half of each DTP (Drone Control is the other component), but unlike the Drone Control aspect, every DTP shares the same Communication Protocol.

Drone Control

The Drone Control is the second major component of each DTP. However, unlike the Communication Protocol (which runs identically on every DTP), DTPs are differentiated by how their Drone Control operates.
Drone Control is responsible for planning the initial journey route (based on the drone's origin and destination positions) and determining the action to take when another drone is encountered (based on information it receives from the Communication Protocol).

3.1.2 Information Feedback Influence

Below is a high level view of how the information that was gathered from carrying out different tasks was used to provide feedback and influence the choices made regarding other tasks.
The purpose of this diagram is to give the reader an understanding of how the components of the the system relate to each other, and the general thought process that when into decision making.

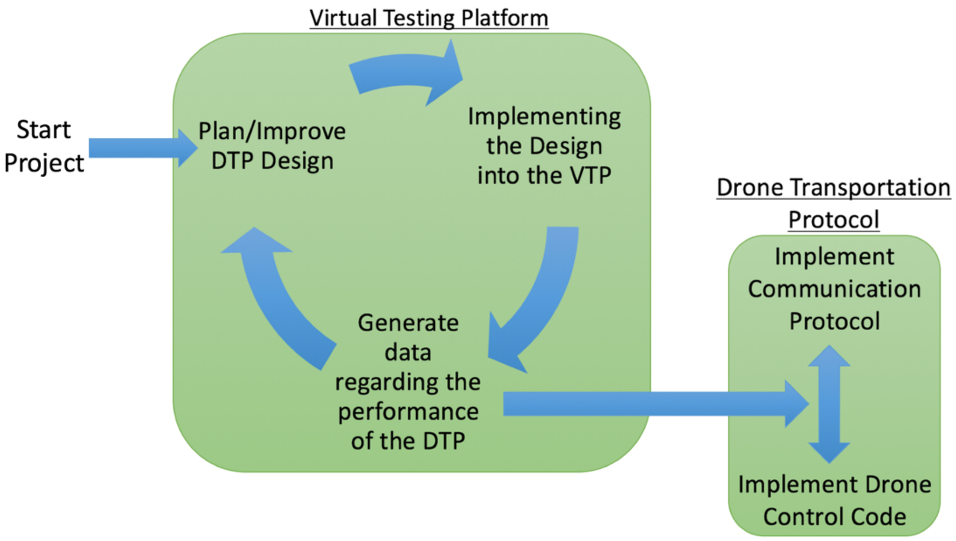

Virtual Testing Platform
Start Project
Plan/Improve DTP Design
Implementing the Design into the VTP
Generate data regarding the performance of the DTP
Drone Transportation Protocol
Implement Communication Protocol
Implement Drone Control Code

Figure 5 - The flow of information across the project

Legend

Plain Text…………………..The plain text represents tasks that were carried out during the project.

Blue Arrows…………….The blue arrows represent the influence that information gained from one task had on another.

Green boxes……………The green boxes represent the collective software systems. The system is labelled above each box in underlined text.

3.2 Virtual Testing Platform

The Virtual Testing Platform is a system that has been developed for this project that can simulate different implementations of Drone Transportation Protocols and generate performance metrics about them. The VTP has been used to guide the development of different DTPs and quantify their performance.

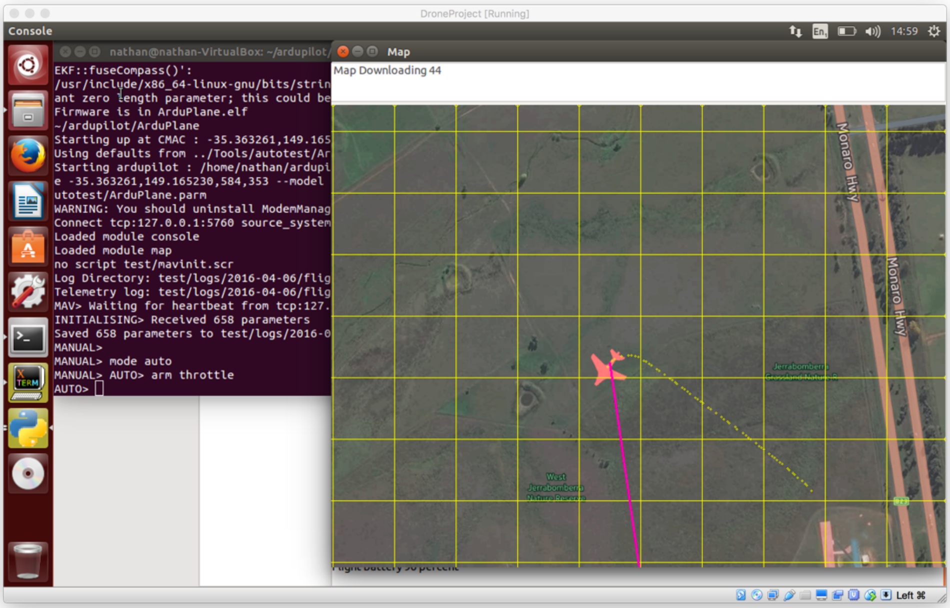
3.2.1 Requirements

- Must Have:
    - The capability to simulate different DTP (both the Communication Protocol and the Drone Control aspects).
    - The capability to generate performance metrics of the different DTP by running tests on them.
- Should Have:
    - The ability to run and test the code that is executable on a physical drone.
    - Be designed in a professional manner so that it is easy to integrate new tDTPs and Test classes as the need for them arrives.
- Could Have:
    - A visualisation tools so that tests can be displayed in a visual manner to convey how tests are progressing.
    - A tool that automates the process of consuming the data generated from the tests and producing performance metrics/insights.
- Won’t Have:
    - A full GUI that allows for selecting DTPs or Test suites to run. This is not an end-user feature so this kind of interface is not necessary.

3.2.2 Initial Design

As DroneKit-SITL already provides a robust and feature-fill platform for simulating individual drones it was the logical platform to utilise while building the VTP. DroneKit-SITL makes it easy to simulate a drone and interact with it using DroneKit-Python, so by building the VTP using DroneKit-SITL it would have the advantage that the code that was written to be tested in the VTP could be ported directly into the DTP with minimal alterations.

To create the VTP using DroneKit-SITL it needed to be extended in two ways: Allowing multiple drones to be simulated concurrently; and implementing tests on these simulated drones.



```
EKF::fuseCompass()':
/usr/include/x86_64-linux-gnu/bits/strin
ant zero length parameter; this could be
Firmware is in ArduPlane.elf
~/ardupilot/ArduPlane
Starting up at CMAC : -35.363261,149.165
Using defaults from ../Tools/autotest/Ar
Starting ardupilot : /home/nathan/ardupi
e -35.363261,149.165230,584,353 --model
utotest/ArduPlane.parm
WARNING: You should uninstall ModemManag
Connect tcp:127.0.0.1:5760 source_system
Loaded module console
Loaded module map
no script test/mavinit.scr
Log Directory: test/logs/2016-04-06/flig
Telemetry log: test/logs/2016-04-06/flig
MAV> Waiting for heartbeat from tcp:127.
INITIALISING> Received 658 parameters
Saved 658 parameters to test/logs/2016-0
MANUAL>
MANUAL> mode auto
MANUAL> AUTO> arm throttle
AUTO> 
```

Figure 6 - Image of DroneKit-SITL being run in a virtual Linux machine

As development started on the VTP it became apparent that DroneKit-SITL was not suited to simulating multiple drones concurrently. The reason for this is because each time a virtual drone was emulated in the software it was assigned system resources, such as ports, which caused conflicts when multiple drones were emulated. After exploring this issue and trialling different fixes without success the decision was made to follow the alternate route, which had been proposed in the initial project plan [Appendix A], to scale back the complexity of the VTP but develop it without the use of DroneKit-SITL.

3.2.3 Revised Design

The revised design for the VTP was to build it from scratch so that it no longer utilises DroneKit-SITL. Now that the VTP was being developed separately from DroneKit-SITL, the DTPs that were being tested would need to be implemented as test versions, which just represent the design choices of that DTP (how it plans its initial route and how it reacts to other drones). These test DTPs were used to generate the performance data, but could no longer be used as code that would work on a physical drone.
The effect that this had on the project was that the final deliverable of an implemented DTP (that was set in the Initial Plan) would have to be scaled back drastically, as the work being carried out on the VTP no longer could contribute directly to the code required for the DTP. The scaled back DTP only has enough Drone Control aspects to be able to demonstrate the Communication Protocol effectively.

Creating a completely separate VTP had benefits. There was much greater control over the whole system which made it easier to design in the features that needed to be examined and not include the more complex simulations. A good example of this would be how in DroneKit-SITL wind speed and direction can be simulated to see how this effects drone flight. In the VTP developed for this project the simulated environment was very abstracted. This simplification made it much easier to see and understand what was occurring inside the simulation because only the core aspects were being simulated, which helped when identifying why collisions occurred.

System design

The VTP was developed in Java (Java SE 8). The decision to use Java was based on the developer's knowledge and familiarity with Java. Java is a statically typed language, which can make finding bugs easier during compilation. The optimisation that occurs during compilation is also beneficial for systems that are simulating large amounts of information. Also, Java provides useful class behaviours, such as abstract and interface classes, that are particularly useful when implementing many classes that share a lot of commonality (like the different Drone Transportation Protocols). C# was also considered as the programming language for this project as it offers similar features, such as garbage collection and compilation optimisation. However, C# is far less flexible in terms of where it can be developed and run. As it is based on .NET it is difficult to run C# code outside of the Windows environment. For these reason Java was selected as the programming language to be used for the VTP.

It was important to create a system that is designed to easily allow different Drone and Test classes to be integrated as and when they are created, but keeps the core components of how the simulated world operates.

The VTP is organised into packages of similar classes. The report will explain what the classes in each package represent and what their responsibilities are. A UML class diagram is included for each package. See below for the UML legend, which uses IntelliJ's UML design.

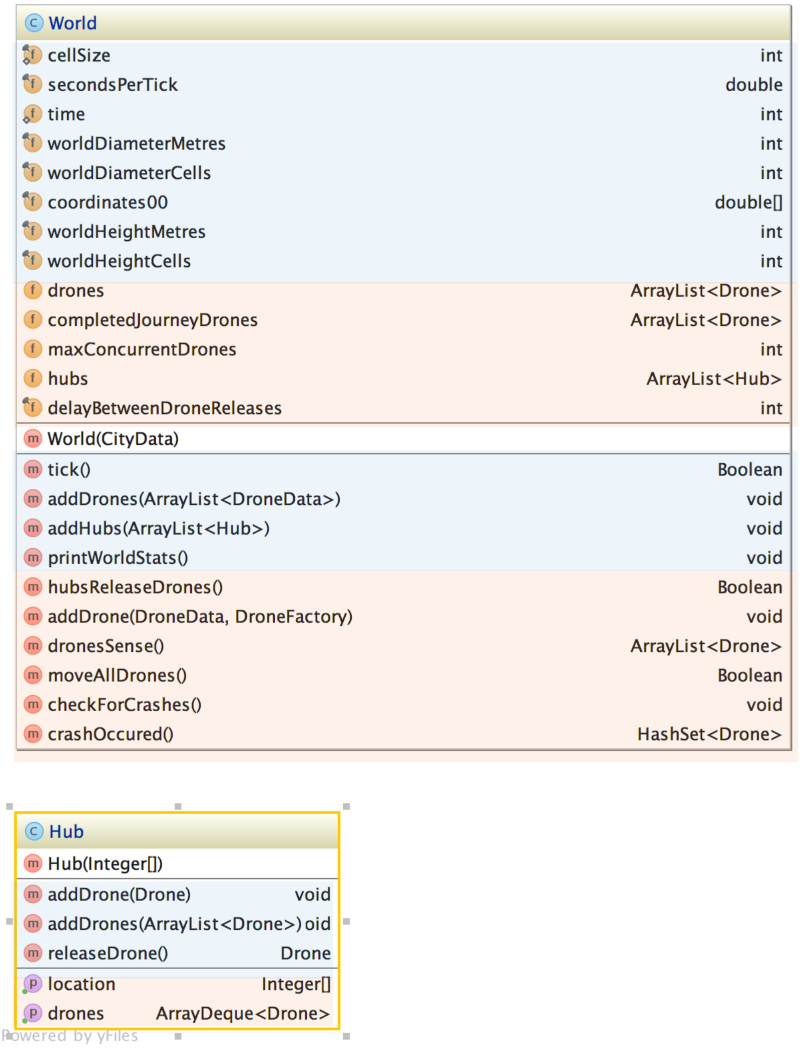
Table 1 - Meaning of IntelliJ UML symbols.

Symbol	Meaning
C	Concrete class
C	Abstract class
E	Enum
⚡	Exception
f	Field
f	Final field
f	Static field
m	Method
m	Abstract method
m	Static method
p	Property

🟦	Blue shaded text is Public
🟩	Green shaded text is Protected
🟥	Red shaded text is Private

World

The World package contains classes that represent real world entities. It was designed so that the World class would have access to all of the drones and any general information that was required to monitor them (such as the dimensions of the space being simulated and the number of 'ticks', or units of time within the simulation, that had been processed). This design allowed drone monitoring to be handled in a top-down fashion within the World class. Although in a real situation, drone monitoring would have to be carried out on individual drones and then aggregated at a later date (if gathering this data was required), this design made it possible to easily get an understanding of the state of the whole simulation. This is necessary to monitor how the drones are performing.



World	
cellSize	int
secondsPerTick	double
time	int
worldDiameterMetres	int
worldDiameterCells	int
coordinates00	double[]
worldHeightMetres	int
worldHeightCells	int
drones	ArrayList<Drone>
completedJourneyDrones	ArrayList<Drone>
maxConcurrentDrones	int
hubs	ArrayList<Hub>
delayBetweenDroneReleases	int
World(CityData)	
tick()	Boolean
addDrones(ArrayList<DroneData>)	void
addHubs(ArrayList<Hub>)	void
printWorldStats()	void
hubsReleaseDrones()	Boolean
addDrone(DroneData, DroneFactory)	void
dronesSense()	ArrayList<Drone>
moveAllDrones()	Boolean
checkForCrashes()	void
crashOccured()	HashSet<Drone>

Hub	
Hub(Integer[])	
addDrone(Drone)	void
addDrones(ArrayList<Drone>)	oid
releaseDrone()	Drone
location	Integer[]
drones	ArrayDeque<Drone>

Figure 7 - Class diagram of world package

World class

The World class holds the information regarding the area that drones can exist in. This class contains all instantiated drones, which is suitable for tracking their state as the simulation runs.

- Represents:
    - This class represents the space that all drones must operate in.
    - Space is represented as a collections of cells, with each cell representing a 5 metre by 5 metre cube.
    - This size was chosen because 5 metres is the minimum proximity of drones. This makes it easy to detect when drones have violated the proximity limitation.
- Responsibilities:
    - This class is responsible for storing the position of all drones currently embarked on a journey.
    - This class is responsible for the simulated Communication Protocol, the process of identifying if two or more drones are within range of each other. This was designed this way because this class contains the positions of all the drones, so it is the most suitable component of the system to carry out the process intensive task of identifying nearby drones. This task utilises Quad-Trees to search the space for drones in close proximity.

Hub class

The Hub class stores drones that have not yet started their journey. Hubs exist to account for areas where a high density of drones should not be considered unsafe.

- Represents:
    - This class represents a location where many drones would be stored at the same time.
    - This class is required because of the special properties of Hubs: multiple drones can be in the same space without violating the proximity limitation.
- Responsibilities:
    - Storing drones and staggering their start journey time.

Drones

The Drone package contains all the classes that represent different models of drone or that are drone related, such as factory classes. Drones differ by having different route planning methods and different reactions to SAA scenarios. The rationale behind designing the Drone package in this way was to provide code that made it easy to implement new tDTPs while reducing the chances of breaking core functionality of how a concrete drone class should operate.

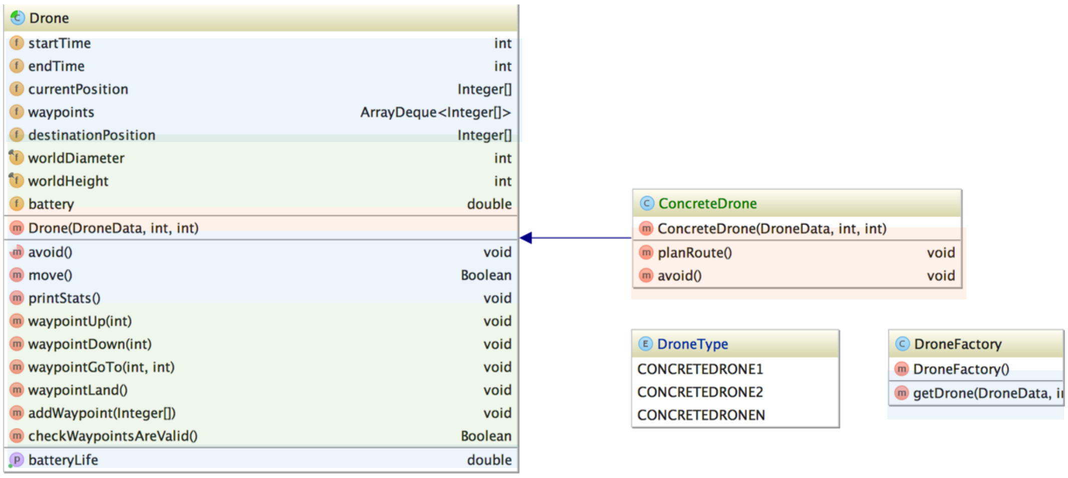


Figure 8 - Class diagram of drone package

Drone abstract class

The Drone abstract class provides the shared codebase that all drone subclasses must extend. It also provides an interface for the World class to follow when interacting with Drone classes.

- Represents:
  - This class represents the core mechanics that are shared by all drones, such as storing routes and moving though the world.
- Responsibilities:
  - This class has only one abstract method, avoid(), which concrete drone models must implement as different drones have different reactions to SAA scenarios.
  - It uses protected methods to provide functionality to the classes that extend it.

ConcreteDrone class

This is a concrete implementation of the Abstract Drone class.

- Represents:
  - This class represents the different aspects that various DTP have: Their method of route planning, and their reaction to SAA scenarios.
- Responsibilities:
  - Each implementation of a ConreteDrone class must provide a different routePlanning method and avoid method that can be tested.

DroneFactory class

- Represents:
  - This is an implementation of the Factory Design Pattern.
- Responsibilities:
  - Ensure that generating drones is always handled correctly.

DroneType enum

- Represents:
    - The different possible types of drones that can exist.

Tests

The test classes exist to monitor different aspects of simulations when different tDTPs are used. These are used to generate the performance data regarding different types of drones. The test package was designed with a similar rationale to the drone package. The abstract Test class provides the core functionality for all test so that new concrete test classes can be implemented without the need to consider the core mechanics.

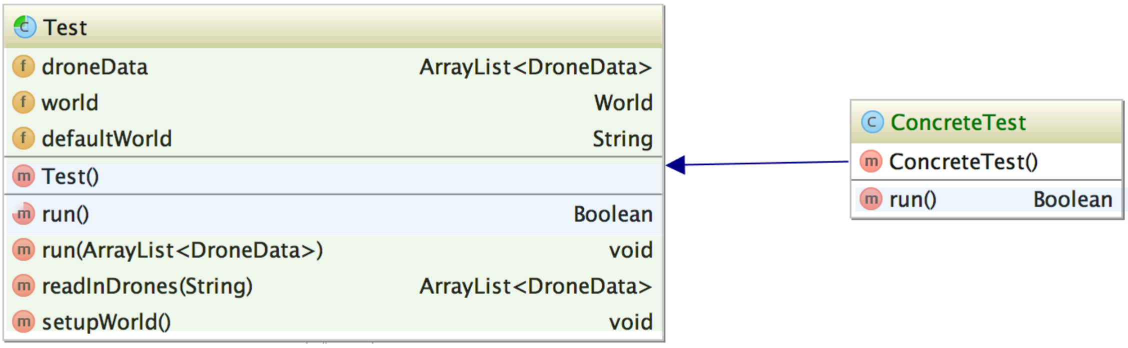


Figure 9 - Class diagram of test package

Test abstract class

Similar to the Abstract Drone class, this provides a shared code bases for all test subclasses to follow.

- Represents:
    - A single test of a simulations. Each test generates an instance of the World class to carry out the test in.
- Responsibilities:
    - Implement any shared code that the concrete test classes utilise.

ConcreteTest class

A concrete implementation of the abstract Test class.

- Represents:
    - A concrete instance of a test.
- Responsibilities:
    - Each test must monitor some different aspects of the the simulation to generate metrics regarding the drone performance.

Helpers

This package contains miscellaneous classes that assist in the running of the program, such as CSV parsers and a class that logs text to an output file.

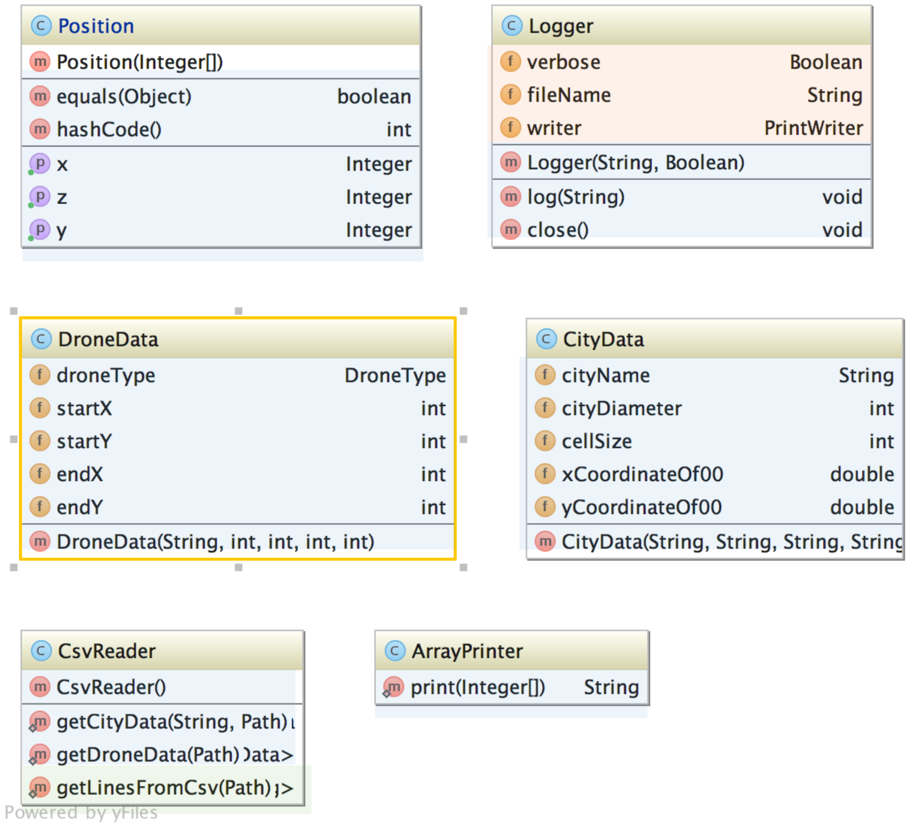


Figure 10 - Class diagram of helper package

Position Class

- Represents:
    - This class represents the position of a drone.
- Responsibilities:
    - It must implement equals() and hashCode() methods to allow position values to be compared easily, when being used as the key in a HashMap object.

Logger class

- Responsibilities:
    - Provides an easy way to to write text to an output file.

DroneData class

- Represents:
    - It holds key data about how a drone should be initialised.

CityData class

- Represents:
    - It holds key data about how a city should be represented in an instance of a World object.

CsvReader class

- Responsibilities:

    - Provides useful methods for parsing CSV files.

ArrayPrinter class

- Responsibilities:
    - Provides a useful method for printing String arrays to Console.

Exceptions

This package contains exceptions that relate to drones and are specific to this project, all of which are self-explanatory: InvalidWaypointException, OutOfBatteryException and DroneCrashException.

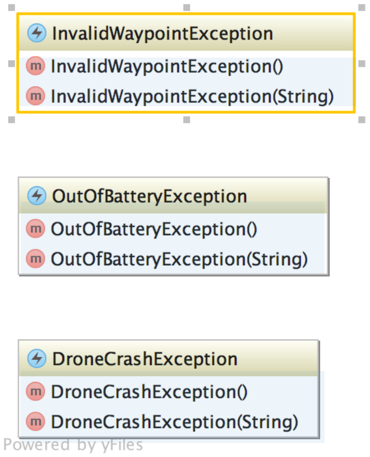


Figure 11 - Class diagram of Exceptions package

3.3 Communication Protocol

The communication protocol is the system that allows drones to detect and react to each other. It makes up a key part of the DTP (the other key part being Drone Control, see Section § 3.4). Without the communication protocol the DTP is still able to function but loses all SAA capabilities.

3.3.1 Requirements

- Must Have:
    - Allow drones to detect other drones remotely.
    - Have a large enough range to react to drones before getting dangerously close (considered to be < 5 metres, as defined in Section § 2.2.3).
    - Find an acceptable balance between power consumption and range.
- Should Have:
    - Ability to compute how to respond to approaching drones and pass this information over to Drone Control component.
- Could Have:
    - Ability to resolve situations by collaborating with other drones.

    - Explore safeguards that can be taken against malicious drones that attempt to interfere with the drone's journey.
    - Ability to recognise and respond to "Emergency Service" drones, that have priority of movement.
- Won't Have:
    - Ability to sense drones that are not running the known Communication Protocol.

3.3.2 Research

To implement the Communication Protocol, it was first necessary to select a suitable wireless communication technology to use, considering the environment that the technology will be used in. Then it was important to decide what information needed to be transferred in order for drones to react appropriately.

Wireless technology comparison

The choice of wireless technology used was an important decision within this project as it affects how the Communication Protocol can interact with the rest of the system. Important considerations that needed to be taken into account include:

- Does the wireless technology have a suitable power consumption?
    - This is important as it will be battery powered, and if the technology has a very high power consumption it will reduce the flight time that the battery can provide.
- Does the wireless technology have a suitable range?
    - As mentioned in Section 2.2.3, the range of the wireless signal effects both the speed that drones can travel at and the processing speed that is required for the on-board computer. A larger range is better.
- How easy will it be to program code that interacts with the wireless technology?
    - This is important because the Communication Protocol and the Drone Control code will both combine to form the DTP. As Drone Control will utilise DroneKit-Python, a Python module for interacting with the wireless technology will be very beneficial.
- How cheap/easy it it to integrate the technology into a drone?
    - This is important because the Solo drone model only has a closed 3DR Link Wi-Fi network, which cannot be accessed, so the wireless technology used would need to added via the on-board companion computer.

Technology	Bluetooth Classic (Class 1)	Bluetooth Low Energy	ZigBee	Wi-Fi (n)
Real world data throughput (Mbps)	1	0.27	0.2	150
Max. Range (Metres)	100	50	100	100-250
Battery Life	Days	Month-Years	Months-Years	Hours
Network Size	7	Undefined	64,000+	255

Available Python libraries	PyBluez [20]	PyBluez[ble]	Unavailable	wifi [21]

Table 2 - Wireless technology comparison. Data gathered from FCC Technology Comparison. See Appendix D.

From this comparison the conclusion is to use Bluetooth Classic. The reason for this choice is that Bluetooth has a good balance of range and battery life, and Bluetooth hardware is very cheap (a Bluetooth USB dongle can be bought for under £5 [22]). It is also supported by PyBlueZ, which is robust and actively developed, as can be seen on its GitHub page [23].

SAA Scenarios

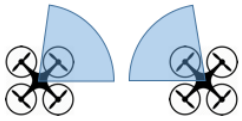
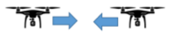
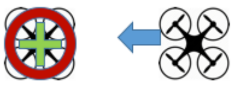
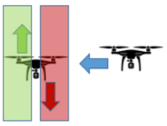
Below are the details of the different scenarios in which drones are near enough to detect each other and therefore are required to take Sense and Avoid (SAA) action.
Each scenario detailed below will only consider two drones. As it is possible that more than two drones could be involved in an SAA scenario, these will be considered as compounds of the scenarios given below.
For example, if we have two drones in a Head-on-Head situation, and a third drone coming in at a right angle, this could be considered at RightAngleSAA(Drone1, HeadOnHeadSAA(Drone2, Drone3)).

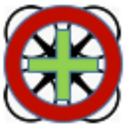
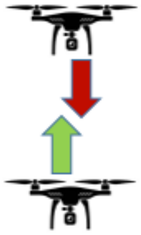
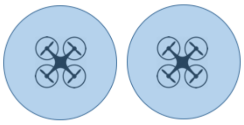
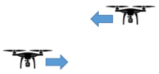
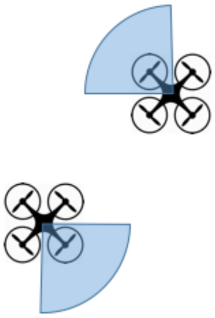
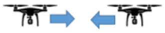
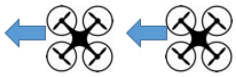
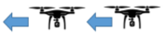
Legend

✚ Indicates the drone is moving vertically up

○ Indicates the drone is moving vertically down

◔ Blue full or partial circles represent that the drone can be travelling in any direction represented from the centre of the circle to the edge.

SAA Scenario Name	Description	Birds Eye view	Side View
Intersect collision	When two drones are heading towards each other.		
Cross lateral collision	When a drone travelling horizontally will collide with a drone travelling vertically.		

Vertical collision	When a drone travelling upwards will collide with a drone travelling downwards.		
Vertically Near (No Danger, no action required)	When 2 drones are on different vertical planes but both are travelling horizontally.		
Passing (No Danger, no action required)	When 2 drones are nearby and on the same vertical plane, but have no danger of intersecting each other's path		
In convoy (No Danger, no action required)	When drones are travelling in the same directions, so will be near each other but have no danger of colliding.		

3.3.3 Design

The Communication Protocol is designed to be the same for all different DTPs. Its job is to detect other drones and determine the SAA scenario it has encountered. It is then the Drone Control component's responsibility to act upon this information.

The Communication interaction will occur in several steps.

1. While a drone is on a journey it will constantly try to discover other devices.
2. When another drone is discovered the connect procedure will occur.
3. When the drones connect it will use its current position and the position of the waypoint it is heading to to determine its trajectory.
4. It will send its trajectory to the other drone.
5. It will expect to receive this pieces of information from the other drone.
6. From these two trajectories, it will determine which SAA scenario it has encountered.
7. An SAA scenario Enum will be passed to the Drone Control component.

8. It is now the responsibility of the Drone Control component to determine how to react.

Testing the Communication Protocol

It is important that the Communication Protocol is properly tested. This poses a challenge because testing with physical drones would be impractical and unsafe, but the VTP does not provide a means to execute runnable version of the DTP.

The solution to this problem is to test the Communication Protocol using two separate instances of DroneKit-SITL virtual environments that are running on two separate computers but which connect using physical Bluetooth dongles. This provides a way to test actual Bluetooth connections, and run DroneKit-Python code to ensure it would work as expected when executed on a drone, but in a virtualised and completely safe environment.

3.4 Drone Control

The Drone Control component is the second key component of the DTP. This controls the physical movement of the drones and provides the software that differentiate DTPs from each other. The features that differentiate DTPs are how it initially plans its route from origin to destination and how it reacts to other drones in SAA scenarios.

The Communication Protocol is responsible for detecting other drones and determining the SAA scenario that it has encountered, but the Drone Control is responsible to determining the following action that the drone takes.

3.4.1 Requirements

- Must Have:
    - Adhere to the safe operating limits defined in Section § 2.2.3.
    - Determine an initial route through non-cooperative planning. I.e., selecting a route that has a minimal chance of requiring SAA without collaborating with others.
    - Respond to feedback from the Communication Protocol to take evasive action, when necessary.
- Should Have:
    - Ability to carry out multi-destination journeys.
- Could Have:
    - Ability to collaboratively plan initial route, in attempt to see how this improves journey time.
- Won't Have:
    - A testing framework that allows for testing multiple physical drones. This is considered infeasible and would require the use of many drones over a large space.

3.4.2 Design

The change of design to the VTP meant that the Drone Control would have needed to be designed and implemented twice to achieve the original objectives: once for the VTP as a class that demonstrates the design decisions of the DTP; and a second time using DroneKit-Python, which would be executable on a drone. This led to a decision being made that the executable DTP implementation would be drastically scaled back to provide only enough functionality to demonstrate the Communication Protocol in action.

Below are descriptions for different DTPs. As these were developed and tested in the VTP, the performance data that was generated informed what changes to make when designing new DTPs. The overall rationale for designing the different types of drone was to initially start with the Basic Drone what would have 'common sense' rules that get it from A to B, then to slowly add new design decisions to each iteration of tDTPs to see how they compare to the previous one. When improvements could be seen, the design changes were carried forward into the next iteration of drone designs.

Basic Drone

The Basic Drone plans its route by going up to 200 feet then going directly towards its destination and landing.
When it encounters another drone it moves to a random height, then continues its journey.

Random Height Drone

The Random Height Drone plans its route by going up to a random height then going directly towards its destination and landing.
When it encounters another drone it moves to a random height, then continues its journey.

Hub Drone

The Hub Drone operates exactly like the Random Height Drone, but after reaching its destination it is required to return to its original location.

Rude Drone

The Rude Drone plans its route by going up to a random height then going directly towards its destination and landing.
When it encounters another drone it stays stationary for 5 seconds then check to see if it is clear to move. If not it moves to a random height, then continues its journey.

Polite Drone

The Polite Drone plans its route by going up to a random height then going directly towards its destination and landing.
When it encounters another drone it moves backwards for between 1-10 seconds and then continues its journey.

4 Implementation

The Implementation section of this report will discuss how the designed project software was carried through to full implementation. It will also look at the problems that arose during the implementation and how these were overcome. When developing software, a seemingly disproportionate amount of project time can be taken up with encountering, identifying and resolve unexpected issues, so this section aims to explain where a large amount of the project time was spent.

This section will be split into the implementation of the Virtual Testing Platform and the Drone Transportation Protocol, detailing the critical or interesting aspects of these systems, the problems that were encountered and how and to what extent they were overcome.

4.1 Implementing the Virtual Testing Platform

The Virtual Testing Platform is the system that allows different DTPs to be tested and validated. Below the report will go into the details of how the original development of the VTP was attempted, the issues that arose from this and how the redesigned VTP was eventually undertaken.

4.1.1 DroneKit-SITL

The initial design for the Virtual Testing Platform relied heavily on utilising DroneKit-SITL. DroneKit-SITL provides a way to simulate drones to a very high degree of detail. For example, drones simulated in DroneKit-SITL will keep track of their exact battery life and can monitor the voltage being drawn from the battery during different manoeuvres. DroneKit-Python can be used to connect to a DroneKit-SITL drone and give it flight instructions. Initially, code was developed that would instantiate a drone in DroneKit-SITL and give that drone instructions to fly up to 100 feet, then travel the distance of roughly a kilometre and land again. This was designed as a 'hello world' program that was capable of displaying a basic understanding of the programming library being used. This can be seen in the file 'basicAToB.py'.

When a drone is instantiated it opens up a port where it listens for a connection. This is how a real drone running MAVLink and ArduPilot (which the majority of drones use to power their on-board computer) would function. The connection can either be made through a wireless link (in the case of the 3DR Solo, it connects through 3DR's proprietary '3DR Wi-Fi Link' technology) or through the physical port on the drone.

Using DroneKit-SITL made for an accurate simulation because it is possible to connect a Raspberry Pi running a DroneKit-Python program to the port on a Solo drone to give flight instructions. The only added code that was required to simulate this was the step of instantiating the simulated drone, which only requires a few lines of code:

```
sitl = SITL()
sitl.download('copter', 'stable', verbose=True)
sitl_args = ['-I0', '--model', 'quad', '--home=-35.363261,149.165230,584,353']
sitl.launch(sitl_args, await_ready=True, restart=True)
```

The `sitl` variable can then be connected to with the code:

```
ip = 'tcp:127.0.0.1:5760'
vehicle = connect(ip, wait_ready=True)
```

In this example, it shows the default address of the simulated drone that was created from the `SITL()` command.

The remaining code is identical to how it would need to be written for functioning on a physical drone, except the connection address would need to match that of the drone being connected to.
Once the `vehicle` variable is connected to the simulated drone it is simple to access simulated 'live state' information about the drone, such as heading, airspeed, waypoints and voltage draw.

This worked very well and the ease of development using DroneKit-SITL and DroneKit-Python was surprising. However, the VTP requires multiple drones to be simulated in the same environment so that it could provide an understanding of how numerous drones could function and interact with each other in the same space. When attempting to implement multiple DroneKit-SITL drone simulations into a single program issues arose. A second DroneKit-SITL drone could be instantiated but was impossible to connect to because of conflicts between available ports. Using the debugging tools in PyCharm, it was possible to identify that the code that assigned the ports was part of the ArduPilot-SITL package, a C++ package that DroneKit-SITL is built on top of.
Multiple attempts were made to change the port that the DroneKit-SITL drone was listening on. Initially, alterations were made in the DroneKit-SITL code but it became apparent that the ArduPilot-SITL interface did not allow access to this. Another attempt was made to edit the ArduPilot-SITL code and then recompile the binaries that DroneKit-SITL relied upon but it was difficult to navigate and identify the changes that were required or the effects that some changes would have on the package.
The issue of DroneKit-SITL not being suitable for the VTP had been planned for in the initial work plan. As enough time had been spent trying to solve this problem and there was no clear solution the decision was made to move into the alternate plan of designing the VTP as a stand-alone program (covered in the next Section, § 4.1.2).
The reason that this occurred was because there was not a full understanding of DroneKit-SITL or how it functioned at the planning stage. Fortunately, this lack of knowledge was accounted for in the work plan, but this could have been avoided if there had been deeper research carried out before the initial planning of the project. Although this would have required more time so overall this approach is still considered as a suitable choice.

4.1.2 Java VTP Implementation

The revised implementation of the VTP was undertaken with the aim of building the platform as a stand-alone program so that it ensured that it provided the tools necessary for testing DTPs. This decision had benefits and weaknesses.
The main drawback was that, unlike the DroneKit-SITL solution where fully functioning DTP could be tested, this required abstractions of the DTPs (referred to as test DTPs, or tDTPs) to

be developed for it. The tDTPs would emulate the initial route planning and avoidance methods that could be carried out by DTPs designed to run on physical drones. Another disadvantage was that it requires more time to develop and did not utilise the existing DroneKit-SITL package, which in comparison had a much greater level of detail. However, the simplification can also be seen as a benefit as it made the whole simulation easier to understand, and therefore easier to gain insight from. Having built every component of the VTP meant that specific tests (such as identifying nearby drones) could be designed into the simulation, rather than developed on top of it, which would have likely been far more processer intensive.

The new VTP was developed in Java. Although the DroneKit suite is developed in Python the redesigned VTP was designed to be a completely separate entity, so any programming language could be used for its development. The main motivation for choosing Java was due to the developer's knowledge of the language and its cross-platform functionality. C# was considered but the restrictions on the development environment being only within Windows made it an infeasible choice. Java is a relatively fast and safe language. The fact that it is a managed language, and offers tools like garbage collection, makes it easy to develop with. Java, being a compiled language, makes it far more efficient that an interpreted language. JetBrains [24] creates a fantastic suite of IDEs that cover development for many different languages. However, IntelliJ (for Java development) is the only IDE they offer for a managed and compiled, object-oriented programming language, which enforced the choice of using Java.

The main components that were developed for the VTP were the Drone classes, the Test classes and the World class. The Test abstract class was developed to provide a shared codebase that the other tests could utilise, it also acted as an interface for how the main method should interact with them. Each test that is instantiated in the main method creates a single World instance. The World class can read in data from a comma separated values (.csv) file which is used to provide an easy way to set up the drones that will be loaded into a test. A typical line of the input .csv file looks like this:

```
BASICDRONE,0,0,10,0
```

In this example, the first value is the drone type to be instantiated, the next two are the starting coordinates and the final two are the destination coordinates (both start and destination coordinates are assumed to be at ground level). Using this method makes it easy to run large but repeatable tests and to be specific when setting out the initial state of any simulation.

After the World class has created an instance of all the drones it begins to simulate the drones' movement through the world. It repeatedly carries out the `tick()` method until all drones have completed their journeys. On each tick, every drone carries out a task that can be achieved in one unit of time. Most often this task is moving from one cell to the next, but it can also include detecting other drones and processing or carrying out responses to SAA scenarios. As the simulation is ongoing the World class can monitor the state of the drones to identify how they are behaving.

After all drones have completed (or fail to complete) their journeys the World class can pass this information up to the Test class. The Test class is responsible for monitoring the output from the World class to determine the result of the test.

Flow of data

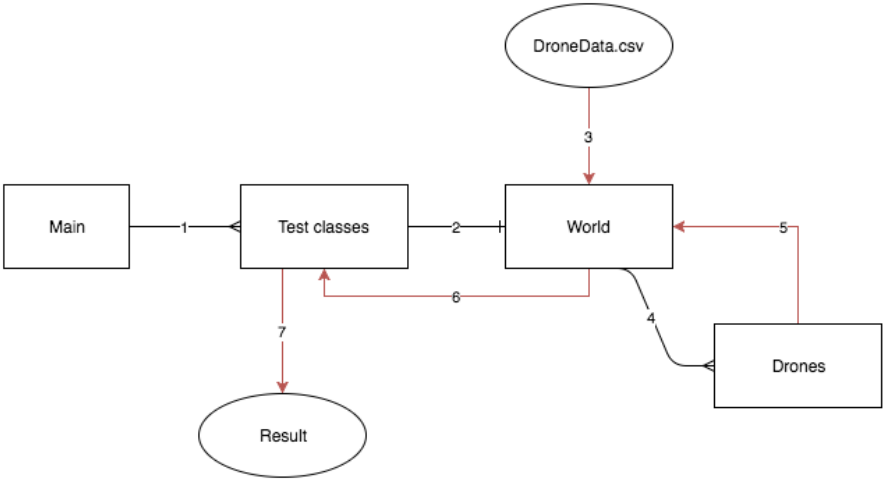
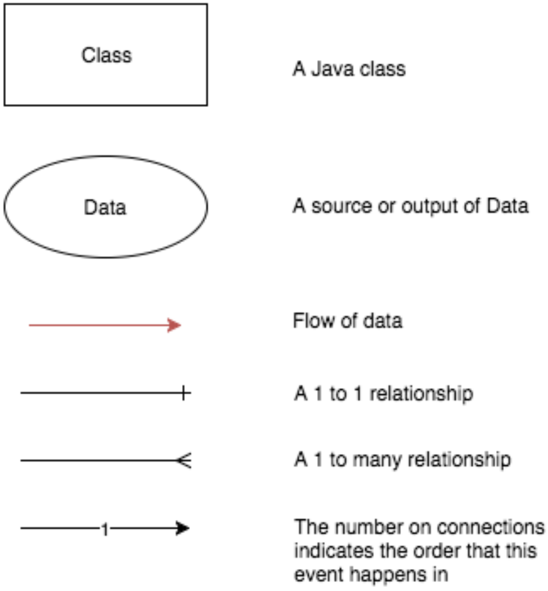


Figure 12 - Runtime flow of data

Legend


Class
A Java class
Data
A source or output of Data
Flow of data
A 1 to 1 relationship
A 1 to many relationship
1
The number on connections indicates the order that this event happens in

The Drone classes were implemented in a similar way to the Tests. There is an abstract Drone class which all others extend. The Drone class handles shared functionality, such as the implementation of moving between cells in the World, and provides an interface for the World class to follow when interacting with drones.

The purpose of the VTP is to test different tDTPs. All drones share the same core functionality, but different tDTPs will have different methods for selecting the initial path they plan from their origin to destination, and they will have different tactics for avoiding other drones when they encounter SAA scenarios. Implementing the abstract Drone class provided an easy way to quickly iterate on different tDTPs, that ensured functionality would not be broken.

4.1.3 Successes and Failures

Overall the final implementation of the VTP can be considered a success as it allows the testing and validation of tDTPs. However, there are some areas where it did not meet up to its initially planned potential.
A core component of any software project should be testing. The VTP does not incorporate any testing of the code functionality. A testing framework, such as JUnit, provides an easy way to perform repeatable tests on software. The reason that this testing was never implemented was due to time constraints. Testing provides a great way to ensure that code is kept to a high standard of quality and that breaking-changes are not made by contributors. However, as this program was developed by one person, it was possible to develop it without a robust testing framework, although it would have undoubtedly helped with general quality control.

GitHub was used to keep a backup of code versions. Git, and other version control software, provide the best experience for backing up codebases. GitHub provides a way to easily see changes when committing code, which helps with going though ‘code reviews’ before storing the changes. However, Git could have been used more effectively throughout this project. Branches were never utilised during this project, which would have been a big help when implementing big features into the program that required multiple checkpoints.

4.2 Implementing the Drone Transportation Protocol

The Drone Transportation Protocol consists of the Communication Protocol and the Drone Control component. It is the functional code that operates a physical drone to provide autonomous control. The DTP was initially planned to be the realisation of the tDTPs that were designed and tested in the VTP, however, the final deliverable of the DTP was greatly scaled back. Initially, the VTP was planned to support code written in DroneKit-Python that would also function on a physical drone, but when the change in the VTP happened it left a gap in the project for how DTP code would be tested. As a result of this, the scope of the DTP was changed to just provide enough Drone Control to demonstrate the Communication Protocol.

4.2.1 Communication Protocol

From the research carried out in Section § 3.3.2 it was decided that Bluetooth Classic was the best choice of wireless technology for the Communication Protocol. As well as fitting the criteria for battery life and range, a big advantage was that there are existing Python libraries that allow development of Bluetooth. This was an important consideration because the Control component of the DTP is built using DroneKit-Python, a Python based package. Therefore, existing Python libraries that provide a means to easily develop programs that utilise Bluetooth would allow the Communication Protocol to integrate tightly with Drone Control and execute in the same program.

The Bluetooth Python library that was used is PyBlueZ. This is the most popular Python Bluetooth library, it has an active developer community [25] and detailed documentation [26]. PyBlueZ is built on BlueZ, which is the Bluetooth stack utilised by many Linux distributions.
During this project, the DTP was tested on Raspberry Pis running Raspbian OS, which is a version of Debian optimised for the Raspberry Pi. This environment was chosen because of the fact that a Raspberry Pi can be connected directly to a Solo drone to give instructions to the on-board flight controller. Installing PyBlueZ onto the Raspberry Pis required ensuring that several base modules were installed before PyBlueZ would function correctly. The steps to installing PyBlueZ in Raspbian are to type the following commands into the terminal:

```
sudo apt-get install python-pip python-dev ipython
sudo apt-get install bluetooth libbluetooth-dev
sudo pip install pybluez
```

The initial development of the Communication Protocol was designed to connect two Raspberry Pis over Bluetooth and to share basic information. The purpose of this was to show that it was possible to get the basic functionality of Bluetooth working in this development environment. It was easy to identify the Bluetooth on each Pi by using the `hciconfig` command in the terminal, and using this information it was possible to get a Python script to connect a BluetoothSocket object to the available Bluetooth port. However, a problem arose when trying to get the devices to discover each other over Bluetooth. Time was spent looking through the PyBlueZ and BlueZ documentation and though online discussion boards without finding an obvious solution. A final attempt was made by using a USB Bluetooth dongle instead of the built-in Bluetooth module on one of the Pis and this fixed the issue. An important lesson was learnt from this: small and seemingly insignificant changes can have large effects on the outcome and Bluetooth connections can sometimes be unreliable. This would need to be taken into consideration when developing the DTP and when testing the tDTPs. Integrating Bluetooth's unreliability into the VTP would ensure that the DTP could still be considered safe enough to operate a drone (for example, by introduction a level of uncertainty to the simulated connection time between drones in the VTP).

The testing was carried out on two separate Raspberry Pis that both had Bluetooth dongles connected to them. Each Raspberry Pi would run a simulation of a drone using DroneKit-SITL and then connect to each other over Bluetooth. Using DroneKit-SITL to simulate a separate drone on two separate devices meant that there were none of the issue that arose when trying to simulate multiple drones in the same environment (as mentioned in Section § 4.1.1). Although this type of testing is not scalable, as it would require a different Pi for every drone being simulated, it did work at a scale of two drones and it provided a useful way to test the actual Bluetooth technology, not an emulation of it.

The final delivery of the Communication Protocol was two different Pis that were able to connect and share data about the simulated drones running on each one. The program instantiated a simulated drone and then connecting to another simulated drone so they could share information about their states with each other. The reason that the simulated drones did nothing more than get initialised is covered in the next section § 4.2.2.

Although the programs did not use the data shared with each other to determine the SAA scenario they had encountered, by sharing information such as GPS location, heading and air speed, this calculation would be possible in future development.

4.2.2 DroneKit-Python

As covered previously, after the redesign of the VTP the aim for the DTP was reduced to just providing a means to demonstrate the Communication Protocol. To demonstrate the Communication Protocol using the set-up of two separate Raspberry Pis required getting DroneKit-Python and DroneKit-SITL installed in the Raspbian environment. Setting up DroneKit-Python was an easy process that just required following the instruction on the DroneKit-Python documentation page [27], which is effectively just putting the following two lines of commands in the terminal:

```
sudo apt-get install python-pip python-dev

pip install dronekit
```

Setting up DroneKit-SITL was not as simple because DroneKit-SITL does not yet support ARM based architecture. There is a work around for this by installing and building ArduPilot's SITL (which is what DroneKit-SITL builds on top of) and then directing the DroneKit-SITL to the binaries created from this. To achieve this, it was necessary to follow the installation steps offered by ArduPilot. This installation process is much longer than the DroneKit install, but still simple to follow. The guide can be found on the ArduPilot website [28]. Following these steps made it possible to run a DroneKit-SITL simulation on the Raspberry Pi, however it was not a complete solution. A program was developed that provided some basic drone flight simulation to test the Communication Protocol. This program worked in an OS X 10.11.4 environment, but when the same program was run on a Raspberry Pi (with DroneKit and ArduPilot-SITL installed) the simulated drone was unable to take off. It would not progress beyond the initialisation process and would not respond to any input. After looking for a solution on the developer discussion boards for DroneKit and ArduPilot the most likely conclusion was that the problem was caused by DroneKit-SITL not having full support for the ARM architecture. This roadblock resulted in the final DTP deliverable only simulating communication between simulated drones that were stationary and did not carry out any flight action.

4.2.3 Successes and Failures

Overall, the final deliverable of the DTP was much smaller than what had been initially planned. The reason for this include the change in the design of the VTP. This had a twofold effect, the DTPs implemented to the VTP no longer contributed code to the Drone Control component, and the redesigned VTP took much longer to complete and therefore resulted in less project time being spent on the DTP overall. However, despite this the DTP did provide a good first step towards the building of a Communication Protocol.

Another reason for this deliverable falling short of its desired outcome could be due to the over-ambitiousness of the project. The project attempted to address three different areas that, although linked by a common goal of autonomous drone control, would have each provided ample scope for a single project.

A successful outcome of the DTP did arise from using GitHub. GitHub did draw attention to the importance of organising code properly, in a way that stuck to known conventions.

Having organised code is useful for big teams of developers, and it can be easy to forget the benefit it has even for projects with a single developer.

5 Results and Evaluation

This section of the report will examine the outcome of the project by looking at how fully the deliverables realise the original goal that they were developed to achieve. It will cover the testing that was carried out and evaluates the achievements and learnings that can be taken from this project.

5.1 Deliverables and Achievements

This project had the high level aim of being able to give the reader an informed recommendation for a safe, efficient and autonomous Drone Transportation Protocol. To achieve this the project was split into three objectives. Below are the details of each objective and evaluations of the deliverables achieved.

5.1.1 Objective 1: Virtual Testing Platform

Objective 1, the goal of creating a Virtual Testing Platform, was set because this was seen as the safest way to get a detailed insight into the performance of different DTPs.
The aims of this objective can be seen in Section § 1.2.1. They are as follows:

i. To develop a simplified model that can gather performance metrics of DTPs.
    a. Secondary aim: Be able to run and test code that is executable on a physical drone within the VTP.
ii. Develop the VTP so that it can test the communication protocol against failure.

The result of this objective was scaled back compared to the original plan. Originally it was planned to utilise DroneKit-SITL to provide a high fidelity environment that could simulate complex drone flights, but limitation in the capabilities of DroneKit-SITL meant that a VTP needed to be built from scratch instead.
Aim *i.* was achieved, but the VTP that was developed has limited applications. It is able to simulate and test different DTPs in a manner that provides modularity. This makes it easy to implement new Test and Drone classes (as long as they extend the Abstract classes) and integrate them into the platform, but this deliverable is lacking in terms of developing a wide range of Tests and Drones to gather performance metrics from. Only a few drone types were developed for the VTP, with the most complex being the Random Height Drone, that would avoid other drones by selecting a random height to move to and then continuing its journey. The result of this being that although the aim was achieved there is very little drone performance data to show for it, which was the reason that the VTP was proposed initially.
The VTP can be demonstrated by running the Main.main() method in the DroneSimulationPlatform project. The VTP informs the user of the state of the simulation by outputting text to console. An obvious improvement that could be applied to the VTP would be the ability to visually display the simulations. This would make it easier to see insights that are harder to observe from just the numerical data being generated. More

improvements that could be made and how they could be approached are covered in the Future Work Section § 6.1.

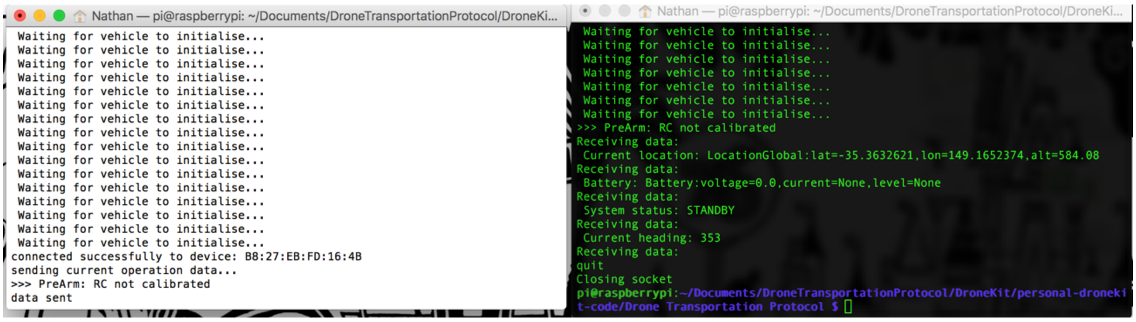
Aims *i.a.* and *ii.* were both effected by the change from using DroneKit-SITL as the base of the VTP to implementing it in Java.
Aim *i.a.* was completely unachieved. It became apparent that this was no longer an achievable goal when the VTP was redesigned. The reason for this being that the code that controls the physical drone was going to utilise DroneKit-Python, but to design a simulation platform that parsed functioning code would have been impractical, as it would have added a huge amount of unnecessary complexity to the VTP which would have obfuscated the core results that it was used to generate. On reflections, this objective should have been re-examined after the change in design of the VTP to account for the known changes that had occurred in the project. This could have help to re-focus the direction of the project.
Aim *ii.* was also effected by the change to the VTP. The aim was to test the Communication Protocol inside the VTP. However, the move away from DroneKit-SITL would have made testing Python code in the VTP very difficult. It was still important to be able to simulate how the Communication Protocol functioned within the VTP so that it could still be used as a tool to validate the functionality of different DTP, but it was no longer possible to test the actual implementation of the Communication Protocol. The Communication Protocol is an important feature of the overall project, so the responsibility of testing the Communication Protocol was transferred into Objective 2, where it was a more logical fit with the other aims (see the next Section 5.1.2).

5.1.2 Objective 2: Communication Protocol

Objective 2 concerns the development of the Communication Protocol. This objective was required to provide a means for the drones to collaboratively Sense and Avoid (SAA) each other.
The primary aims for this objective are:

1. Identify a viable wireless technology for the Communication Protocol.
2. Implement a Communication Protocol that can detect other drones and determine appropriate action required in SAA scenarios.

Aim *i.* was achieved in this report. Sections § 3.3.2 shows the comparison of different technologies and a justification of the final choice of Bluetooth Classic. The choice of Bluetooth Classic was justified by the fact that it provided a sufficient range while also having a reasonably low power consumption and a useful Python library to develop for it. The outcome of this choice can be seen in how Bluetooth Classic was utilised in the development of Aim *ii*.
The Communication protocol that was developed for Aim *ii.* was capable of making a connection between two devices and sharing information regarding their current state, but it was not developed as far as being able to determine the type of SAA scenario that had been encountered. The Communication Protocol can be seen sharing information in the screenshot below:


Nathan — pi@raspberrypi: ~/Documents/DroneTransportationProtocol/DroneKi...
Nathan — pi@raspberrypi: ~/Documents/DroneTransportationProtocol/DroneKi...

```
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
connected successfully to device: B8:27:EB:FD:16:4B
sending current operation data...
>>> PreArm: RC not calibrated
data sent
```

```
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
 Waiting for vehicle to initialise...
>>> PreArm: RC not calibrated
Receiving data:
 Current location: LocationGlobal:lat=-35.3632621,lon=149.1652374,alt=584.08
Receiving data:
 Battery: Battery:voltage=0.0,current=None,level=None
Receiving data:
 System status: STANDBY
Receiving data:
 Current heading: 353
Receiving data:
quit
Closing socket
pi@raspberrypi:~/Documents/DroneTransportationProtocol/DroneKit/personal-droneki
t-code/Drone Transportation Protocol $ 
```

Figure 13 - Screenshot of two DroneKit-SITL simulation connecting and sharing data over Bluetooth

In the above screenshot each terminal window is connected to a different Raspberry Pi that is emulating a drone using DroneKit-SITL. Each Raspberry Pi has a physical Bluetooth dongle which allows them to connect and share the information regarding the drone being simulated on them to the other Raspberry Pi's simulated drone.

Each Raspberry Pi is simulating a drone that could be travelling at any GPS coordinates, but they are always able to connect via Bluetooth because the physical devices are near each other (as can be seen in the photo below). Obviously, this would not be the case if physical drones are operating in a city-wide area. So when the devices connect they send their simulated location data to each other to determine if the simulated drones are near enough to one another to actually communicate.

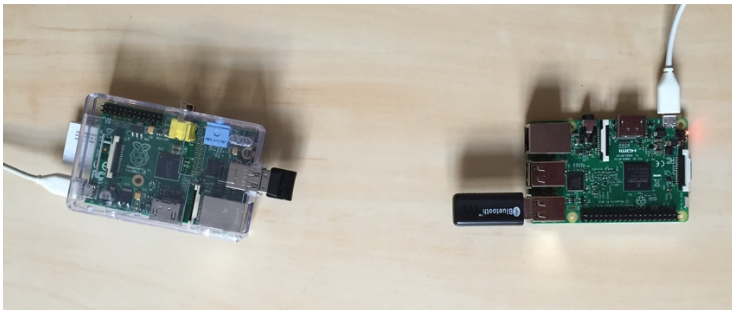


Figure 14 - Image of Raspberry Pi setup for testing Communication Protocol

Although a basic Communication Protocol was developed for this aim, the complete functionality of determining the type of SAA scenario that had been encountered was not achieved. The information required to determine this (the simulated drone's GPS, heading and current waypoint) is sent but the processing of this information was not implemented. Another limitation in the Communication Protocol is that it is currently developed to only allow for a single connection at a time. This is obviously a major drawback for use in a real world situation where many drones could be required to communicate collectively. However, it still has a value in that it demonstrates the concept behind using Bluetooth to communicate, and Bluetooth Classic allows for connections on multiple ports through a single Bluetooth dongle, so adding the functionality of drones communicating with multiple other drones would be an achievable extension of this project.

The reason that full functionality was unable to be developed was due to time constraints that resulted from the re-design and implementation of the VTP. A learning that can be taken from this would be to have a greater awareness of the knock-on effects of earlier delays in the project. This could have been tackled by re-examining the objectives and identifying how they have been effected by the changes to the project work plan and then re-defining the objectives in relation to the time limitations of the project. Having a clearly defined goal and expected outcome for a project can be hugely beneficial for maintaining project direction, particularly when the project deviates from the initial plan.

5.1.3 Objective 3: Drone Control

Objective 3 covered the development of the Drone Control aspect of the DTP. It also covered the planning and identification of the operational limitations that should be applied to drones that are being used to carry out autonomous transportation.
The aims of Objective 3 are as follows:

i. Identify the safe operating limits of the drones.
ii. Implement non-cooperative route planning that utilises Game Theory to minimise encountered SAA scenario.
iii. Implement a functioning DTP that uses the initially planned route and feedback from the Communication Protocol to direct the the physical movements of a drone.
iv. (Secondary aim) facilitate multi-destination drone journeys in the Drone Control component.

Aim *i.* was achieved by the work carried out in Section § 2.2.3, which looked at the physical capabilities of 3DR's Solo drone and identified and justified what limitations should be put in place for the autonomous operation of multiple models of this drone in the same air space.
Aims *ii., iii.* and *iv.* were either not or only partially achieved.
On reflection, setting this objective was misguided, as either the aims would have been more suited for being part of Objective 1, to be considered during the design process of a DTP (like Aims *i., ii.* and *iv.*) or they were infeasible aims, like Aim *iii*.
The development of a functioning DTP (Aim *iii.*) would have always been very difficult to deliver in a meaningful way. In the Scope Section § 1.2.2, the report even proclaims that a demonstrate-able physical drone is not a deliverable of the project, so setting an objective with the aim of developing the code that could be executed on a physical drone but without the safety or testing framework to demonstrate it was unachievable.

The final deliverable for this objective is a program developed using DroneKit-Python that allows a simulated drone to take off, travel a short distance and then land again (within a DroneKit-SITL simulation). There is also a program that just initialises a simulated drone to carry out the Communication Protocol. The second program provides a means for the Communication Protocol to demonstrate that it is capable of connecting two drone via Bluetooth and sharing information about their own position to each other. It was planned for this to be the final outcome of this objective after the redesign and implementation of the VTP was required, so, although the final deliverable for the Drone Control component was much smaller than initially planned, it still provides a useful purpose to the overall project.
Lessons have been learnt from this objective though. If the project was being planned from the beginning again it would be wise for this objective to be removed, or for some of the aims to be reassigned to Objective 1, which covers the design process of what is required for a suitable DTP. The Drone Control component being a deliverable in this project should have not been an objective at all, as it diluted the focus of the project, making in

unnecessarily broad. This objective also did not have any clear outcome, which led to valuable time resources being used in trying to identify how this objective should progress and what it added to the overall project. Instead, these needed to be identified as the project was ongoing, resulting in a lack of understanding of what the final ambitions of the project should consist of. Also, removing this objective and making the project narrower overall would have provided a greater sense of concentration and clarity on objectives 1 and 2, and the project as a whole. This would have had the benefit of being able to delve deeper into the core questions of the project, which would have resulted in narrower, but potentially more interesting findings.

5.2 Learnings

The development of this project has provided a huge learning experience. Below are the main areas that would be reconsidered if this project was being started again now.

5.2.1 Time Requirements

The biggest learning is that every part of project development (particularly software development) takes longer than expected, and that it can be very difficult to account for the extra time that will be required at the start of the project. This is most probably due to ‘unknown unknowns’ [29]. Known unknowns can be accounted for in the planning stages, but when an issue arises that there was no knowledge of during the planning stages it is very difficult to accurately predict the effect that this will have on the larger project.
Another learning is that small changes to the work plan can end up having large knock on effect to the rest of the project. This is similar to unknown unknowns in the sense that it is difficult to predict when or how this might happen.
These two problems are very difficult to overcome. Multiple rounds of planning could be used to incrementally learn more about the problem area of the project, but this still does not totally remove the possibility of unknown unknowns appearing during development.
Another option would be to switch to the waterfall planning method, but that has its drawbacks in terms of reduced flexibility and reactive-ness of the project overall. A final solution would be to plan extra time into the project for unknown issues, but this would most likely result in not improving the reliability of time predictions, but with most task-times being overestimated rather than underestimated.
None of the solution provide a clearly superior way of handling the time requirements during the planning stage of the project. However, it is always important to be more aware of these issues so that they are considered at the earliest possible occasion and so stakeholder of the project can be informed of them as they arise. Task-time prediction is a skill that develops with experience, but it is important to constantly be aware of the challenges that it presents.

5.2.2 Project Breadth

This project suffered from a broad subject area and a general lack of specificity in the objectives. The ambitiousness of this project meant that the objectives were often too

broad to provide a cohesive goal for the project. Although all the objectives' aims work towards the testing and development of a DTP, this project could have benefitted from being more focused. For example, a project with the goal of only making a robust yet versatile VTP would have been complex enough for a complete report, and a much deeper exploration of what a VTP could achieve would have likely been the end result. The same could be said for only exploring the requirements and functionality of a Communication Protocol. When a project is too broad it leads to there being several different directions, which can result in a broad but shallow insight into the problem.
Another learning would be to be as specific as possible in the objectives and aims of the project, as this helps with the guidance of how to achieve the goals.

This project could have benefitted from planning the initial objectives so that there is a clear idea of what the deliverable and test criteria were at the start of the project. In this project a lot of the objectives were quite open ended so the expected outcome of the project was being re-evaluated as the project progressed. Although reflection on the progress being made and the effects this will have on the outcome can be a good model to follow, not having the specifics defined in the initial plan can lead to a process where there is a lack of focus and direction for the project, especially when the work being completed start to deviate from the initial work plan.

5.2.3 Wider Learnings

The Agile development methodology is very robust and suitable for projects that require the adaptability and constant development process that it provides. However, to get the greatest use out of it requires the structure of the scrums to be adhered to, even for a 'team' of one person. By using the overarching structure of Agile but not having the daily progress reviews, or Scrums, means that the reactiveness of Agile can get lost, which can result in a Waterfall-like process but with less planning. This is not an ideal outcome as core components of project planning can get lost. Although, throughout this project the Agile methodology was utilised effectively for most of the process, it is important to identify where it is not being used to its full potential and how it could be integrated more efficiently.

Selecting the correct programming language is important and making the wrong choice within a development project can be difficult to rectify. Although programming languages are just a tool for realising software projects, choosing the 'right' one for the task can have a big effect on the overall outcome and quality. Factors such as developer community, available libraries and efficiency all come into play [30].
The two languages that were utilised in this project were Java, for the VTP, and Python, for the Communication Protocol and Drone Control component. Both of these were suitable choices, as they allowed the developer to utilise existing libraries (the DroneKit suite was developed in Python) or they provided the right balance of efficiency and expressiveness (utilising Java for the VTP was suitable due to the optimisation that the compiler brings to the bytecode).

6 Future Work

The Future Work section of this report will look at ideas that could be explored if this project were to continue development. This will range from ideas that were not achieved due to time constraints all the way up to larger ideas that could only be realised in the longer term.

6.1 Next Steps

This project had a very broad scope and due to unforeseen complications during the implementation not all of the initial ideas were achieved. The 'Next Steps' subsection covers work that was planned but not accomplished (see the Requirements in Section § 3).

Simulation Visualisation

Currently, the Virtual Testing Platform keeps track of drone statistic as it simulates flights and at the end of a test will calculate the metrics that the test was design to deduce. During the simulation the only way to understand what is happening is to read the text and numbers that are being printed to console. This is a dull and unintuitive way to display the simulation.

A task that could be carried out in the next stages of the project's development would be to implement a means to visualise the simulation as it is executing. The World class already stores a 3 dimensional array of the position of all the drones so this is the element that would need to be visualised. Achieving this would make the simulation much more interesting to observe and it could potentially reveal new insights that are easier to identify when observing a simulation visually rather than from the numerical data that the simulations currently generate.

This could be achieved by utilising tools such as JOGL [31] or Java 3D [32]. These are libraries that provide 3D graphical programming for Java. Other simpler options would be to use Java graph plotting libraries, such as JavaPlot [33]. Graph plotting libraries tend not to use 3D graphics, so can be simpler to implement and less processor intensive, while still managing to visualise the simulation in a meaningful way.

Collaborative Communication Protocol

The Communication Protocol is only capable of sending and receiving a very basic set of data. It should be developed further to allow the communication to be much more general. This would allow for more complex interactions. For example, the data sent via Bluetooth should be in the form of a Head and Body object, where the Head informs the receiving drone what the data is that they have just received in the Body. Then, if the receiving drone is capable of handling the data it could react to it.

This could provide the basis for collaborative SAA resolutions, where the drones communicate further to determine the most efficient action to take.

This would also provide the means for Emergency Service drones to give instructions to surrounding drones. This could be achieved by all drones storing the verified emergency service public key. Then, when an Emergency Service drone encounters another drone it could send a message with the Head stating that it is an Emergency Service message and the Body containing instructions for the drone that are encrypted using the Emergency Service private key. The non-Emergency Service drone could then decrypt the Body to see if it is legitimate, and if so follow the instructions.

Utilising Different SAA Scenarios

In the design of the Drone Transportation Protocol, it was planned that the Communication Protocol would identify which Sense and Avoid scenario it had encountered. This information would then be passed to the Drone Control component, which would choose the appropriate action. As implementing this into the VTP took much longer than planned, this feature never got implemented into the Communication Protocol. It was also never utilised in any of the test DTPs that were run in the VTP. These would all react the same way regardless of which SAA scenario had been encountered. This was intended to be a core feature, and the benefit of implementing this would lead to a much greater understanding of how DTPs could be improved.

Collaborative Initial Route Planning

A feature that could be implemented in the future would be a centralised database of all drone journeys that have been started but not yet completed. This database could then be utilised by drones in the route planning process. It would be interesting to see how this effected the number of SAA scenarios that were then encountered, or even if it was possible to eliminate SAA encounters entirely by using this method of route planning.

6.2 Longer-Term Future Work

6.2.1 Mid-Term work

The Mid-Term future work include ideas that were not part of the original plan or requirements, but that were identified as interested possibilities to explore as the project was underway. These should be considered to be developed after the initial core components are all complete. The aim of the Mid-Term future work would be to make the VTP and the DTPs more generalised, so that they could be applied in more situations and be accessible to more people.

Simulate More Diverse Areas

The VTP currently has a very abstracted simulation of city areas. It currently uses the diameter of a city to generate a 3 dimensional grid of cells that represent the space. This could be drastically improved by making the simulated city much more detailed. Each cell

should have a value that exposes what type of space it represents. For example, cells could be identified as No Fly Zones, Urban (with or without very tall buildings), Suburban or Rural. Using this information, it would be possible to implement test DTPs that respond the the air zone type that they are in.
Also, drones in the current simulation do not treat high-speed (>200 feet) and low speed (<200 feet) traffic areas any differently. This should also be implemented into the test DTPs to see how it can be used to improve results.

Making the VTP more accessible

The VTP currently requires a reasonable understanding of the code to implement new tDTPs to be tested in it. It would be great to make the testing of new tDTPs very accessible. This would require making much more documentation of the VTP or giving the VTP a user interface where it would be possible to upload code to be tested. This could require hosting the VTP on a service such as Azure or Amazon EC2 to make it accessible to a wide audience.

6.2.2 Long-Term Future Work

Longer term future work covers less specific but interesting directions that the project could go in. It is inadvisable to go into detail when mapping out longer term plans because it is very difficult to accurately predict where the project will be in the longer run, in terms of how it ends up getting developed or the state of external affairs that could have an effect on it.

Extending testing into Physical Drones

The major long term goal for this project would be to graduate it from a framework for virtually testing drones to testing physical drones. This would require re-writing a DTP that had been tested in the VTP in DroneKit-Python so that it could be executed on a drone. Then it would be good to test this in a controlled environment, such as a large sports hall. If these physical tests could run a DTP successfully it would be a major step towards being able to verify that autonomous drone transportation could be utilised in open environments.

7 Conclusion

In conclusion to this project, I believe that I was only partially able to complete the objectives and goals of this undertaking.
To reiterate, the original aims of the project was to be able to make an informed recommendation for a safe, efficient and autonomous Drone Transportation Protocol. To achieve this, I planned to implement a Virtual Testing Platform which would be used to gather performance metrics regarding different DTPs and inform the decisions made throughout the project. This information would then be used to implement a Drone Transportation Protocol, which would consist of a Communications Protocol and a Drone Control component.
Although I believe that I was only partially able to complete the project objectives, I still see this project as a success. I learnt a huge amount about how an autonomous protocol should be designed and the considerations required for ensuring safety while achieving this. I think that this report provides a solution that can be built upon by others who are interested in pursuing this subject.

The outcome of the project was that I have achieved the objective of being able to recommend a DTP that is autonomous, safe (as it is able to avoid collisions) and efficient (it is possible to carry out a journey across Cardiff). However, in terms of the efficiency, this area could be greatly improved by exploring different methods of route planning and more complex Sense and Avoid actions.
Although the Virtual Testing Platform that was developed was simpler than originally planned it did provide a suitable means for testing different DTPs and gaining an insight of how they would operate in the physical world. The information generated from the VTP provided guidance for the development of the project and was a core component for building an understanding of how drones should be designed. The change from the originally planned VTP to the revised, Java based, VTP did bring unexpected development work for the project. However, I am pleased with the final result of the VTP and the vision and reflection it brought to the development process.
The development of a single executable Drone Transportation Protocol was the biggest victim of the time constraints caused by the altered VTP implementation. The Communication Protocol provides only simple detection and reaction capabilities between drones, and the Drone Control component was reduced to only provide a means to demonstrate the Communication Protocol. However, the implemented DTP did allow me to explore how this problem could be approached and provided a basic step towards autotomizing drone operation.
In hindsight, the goal to implement a full DTP which would have been capable of running on a physical drone was an unwise objective, as actually testing and demonstrating it was always infeasible.

I believe that drones are becoming important tools in modern society and will continue to have wider and more novel uses in the coming years. As drones become more ubiquitous they will play a pivotal role in the proliferation of technological services and they will be a key component in the seismic shift caused by the automation of work. I am pleased to have contributed to the conversation on how the autonomisation of drones could be approached and I am proud for taking on the challenge of implementing a Drone Transportation Protocol.

8 Reflection

In this section of the report I will critically evaluate my own performance throughout the project and identify the skills that I have developed and the areas in which I should focus my learning in the future. I will examine my project objectives to see how well they fit the SMART objectives criteria. Finally, I will compare my progress against the Skills Framework for the Information Age (SFIA) Foundation framework reference, to identify my current level of proficiency and identify where I can improve.

8.1 Improvements

8.1.1 Time Management

Before starting the main body of this project I set out a work plan for how my time would be utilised throughout this project. This can be seen in the Initial Report [Appendix A]. I believe that this was an accurate and achievable time plan in itself, but did not account very well for disruptions to the project progress or changes in designs.

I planned my work load by splitting the project into its three objective components and identifying the major tasks in a single timeline. This allowed me to easily see that all the tasks for each objective were planned for but also that there was not an overload of concurrent tasks that were required to be achieved. I planned the project like this because I know from previous experience that context switching between tasks can create a large amount of time overhead, and that I personally do not work well when multitasking. By planning the project like this I hoped to utilise my time in the most efficient way.

I was able to adhere to my Work Plan until week 8 (the week before the Easter break), at which point I started to fall behind in both the Communication Protocol and Drone Control objectives. This happened because of the additional work that was required to implement the updated VTP.

On reflection, I should have made a more detailed plan of how I would tackle the project if the VTP had to be re-designed. Although I did acknowledge that the original VTP was an area of the project that was likely to require re-examination after the development had started, I did not follow this through with a plan of how to achieve this.
Another method that I could have used to improve my time management would have been to carry out a mid-project review. I was aware of the changes that were required to the VTP by week 5, so I should have taken action midway through the project to look at how I was progressing and how the work plan needed to be updated to reflect the changes that had occurred. By not carrying out a mid-project review, I was left with a plan that was no longer entirely suitable for the work I was doing, and resulted in me falling behind on my work plan

by week 8. In future projects I will plan mid-project reviews into the initial plan, where I can assess the progress made against the initial plan and update it if necessary.

8.1.2 Project Planning

In hindsight, I think that the breadth of this project was very large and as a result of this I was only able to get a reasonably shallow insight into the different objectives of the project. If I were to start this project again, I would severely scale back the coverage of this project and instead only focus on one of the three objectives that I had set myself. I think that any one of these objectives would have provided enough material for a complete report, and by focusing on a smaller area, I think the insight would have been much deeper and therefore more interesting. The only drawback that I could see from having taken this approach would be if I had hit a very difficult blocking point in a project that covered a smaller research area, then it may have been difficult to continue down the narrower research route.

I have also learnt that when planning for a project, it is very important to try to accurately predict the time required for each task, although, as Hofstadter's law states, it is somewhat inevitable to avoid under estimating the time required (*"It always takes longer than you expect, even when you take into account Hofstadter's Law"*).
I will try to improve my ability to predict the time required for project tasks by making an initial estimate, and then after each task is complete compare the planned time to the real time. If they are different I will try to identify the aspects of the task that had taken longer than expected and account for them the next time that I plan a project or task. This method should allow me to constantly be gaining insight from the feedback of the work that I carry out and should help me to improve in the future.

8.1.3 Project Management

For this project I used an Agile Development Methodology. I chose this methodology mainly because it was the one I had most familiarity with from using it during my year in industry. I also felt like it would be suitable for a project where I was aware that the VTP design might have to change mid-way through the project. I also utilised the Scrum Task Board in the form of the Project Trello Board (see Section § 2.4). This board had a list of all the tasks required for my project and it provided a clear way to visualise the tasks that I was currently working on, which tasks where upcoming and which had been completed. This was particularly helpful with keeping me focused on the task at hand as it provided a useful way to 'dump' task ideas onto the board and then organise and select a new task at a later date. I also feel that this iterative approach of constantly writing, testing and re-writing code suited my working style.

I think that I did choose a suitable methodology for this project, as it allowed me the flexibility to adapt the design and direction of the project as it progressed. This was particularly useful when it became apparent that the VTP needed to be completely redesigned and implemented from scratch.
However, I think that I could have benefitted more from the Agile methodology if I had been stricter on myself and stuck more rigorously to the Agile conventions. For example, during

the project I assumed that lots of the activities that are carried out during ‘Stand Ups’/ daily scrums (such as reviewing progress and planning next tasks) were for the benefit of the team, and as such I was far laxer about sticking to the daily scrums because I was working alone. In hindsight, these recaps and reviews of progress are a vital way to ensure that focus is being put into the areas where it is required, and these are a very important aspect of the whole Agile methodology. I think that if I had been stricter carrying out scrums on my own, I would have benefitted from having a greater level of focus about what I was aiming to achieve that day, and what issues had got in the way of progressing the previous day. I think this would have helped with the development of the project overall.

8.2 Project Objectives

An objective can be considered to be of high quality if it meets the SMART criteria. SMART is an acronym for: Specific, Measurable, Achievable, Relevant, Time-Bound. [34]
I will be reviewing my three main objectives that were set out in Section § 1.2.1 to see how they achieve (or not) being a SMART objective.

8.2.1 Specific

Overall I think the objectives lacked specificity. I think that there are a few reasons for this. As discussed in the previous section (§ 8.1.2), I think that the whole project was too broad and therefore would have benefitted from being more precise. With this being an issue at the project level, I think that it was difficult to create objectives that were also specific. Another cause for my objectives not meeting this criterion is that it was very difficult to know what the project would entail at the planning stage. For example, a primary aim of Objective 2 was “Implement a Communication Protocol that determines appropriate action”. The term “appropriate action” is rather vague but it was difficult to be more specific before knowing things like the capabilities of the wireless technology that would end up being used for the Communication Protocol or the dexterity of the drones.

8.2.2 Measurable

I think that all of my objectives could have been more measurable. The objectives were somewhat measurable, as each one has multiple aims which are categorised into Must, Should or Could have. By splitting the objective into prioritised aims it is easier to see how deeply an objective was completed. Although they could be made easier to measure defining exactly at what state a deliverable would be considered complete, and which aims were considered as stretch goals. An example of a measurable aim would be: “A DTP should be developed which can enable 1000 drones to fulfil their journeys in the space of the Cardiff urban area within 30 minutes without any collisions.”

8.2.3 Achievable

I think that Objective 1 (Virtual Testing Platform) was achievable. However, to achieve the objective I was required to plan and attempt a solution before discovering that it was not suitable, and then plan and execute a different solution, which required deviating from the work plan.

I was only partially able to achieve Objective 2 (Communication Protocol). Out of two primary aims I only achieved one completely (identifying a viable wireless communication technology to use). I did make progress towards the overall objective of implementing the Communication Protocol but failed to develop it far enough to identify the different types of SAA scenarios that it had encountered. This was due to the time limitation that was caused by having to redesign the VTP.

On reflection, I think that Objective 3 (Drone Control) had some poorly considered aims. Namely, utilising game theory to implement the route planning methods and implementing a DTP that could operate a physical drone. I think that the possibility of utilising Game Theory was ambitious, because it relied upon me learning about and applying a subject area that I am completely unfamiliar with. I think that the aim to implement the DTP was infeasible because demonstrating physical drone operations was never a planned outcome for the project, so it did not make sense to set a deliverable that would not have the safety or testing frameworks in place to be demonstrable.

8.2.4 Relevant

I think that all three of the objectives can be considered Relevant. I think that the project has a broad but clear direction and all the objectives that were set were directly associated to this goal. Each objective covered a distinct but critical area that was required towards the development of a DTP. The VTP was vital, as it provided a means to test DTPs in a safe manner, and both the Drone Control and Communication Protocols were key components in the DTP.

8.2.5 Time-Bound

In this initial project plan I made a work plan that covered all the major milestones for each different objective on a week-by-week basis. It also covered key checkpoints and check-in meetings that were planned into the project.

I believe that this is detailed enough to allow the objectives to be considered as Time-Bound.

8.3 Progress against SFIA

I think that a reliable way to track personal development and progress is to use an internationally recognised and external set of criteria for what skills a computer scientist should possess.

I will use the Skills Framework for the Information Age (SFIA) [Appendix E] guideline to rate my skill level at the moment. I am using this resource because it is specifically design to apply to ‘Professionals working in Information Technology’.

I also rated myself using this frame work at the end of my year in industry [Appendix F], so I think that it is useful to return to this self-analysis and identify my progression.

The three IT Professional Skills that I have been focusing on throughout this project are Programming/Software Development (PROG), Data Analysis (DTAN) and Solution Architecture (ARCH).

Programming/Software Development (PROG)

Programming and Software Development [Appendix E – Page 30] have been a major part of completing this project.
Previously I had assessed myself as being level 4 in this area. I believe that I have progressed to level 5 because I have displayed that I am capable in driving a project to completion, taking technical responsibility for the development process and I have displayed that I can monitor the performance of the project and keep a detailed report of the progress.

Data Analysis (DTAN)

The VTP provided me with the opportunity to process and generate data in a controlled and organized manner.
Previously I had assessed myself as level 3 in Data Analysis [Appendix E – Page 29]. I believe that I am now level 4 because I have displayed the ability to assess the data requirements of a system. I have applied modelling and analysis in a manner that assures quality and integrity to the data being handled. I think that I have shown this with the planning and execution of the VTP that was implemented in Java.

Solutions Architecture (ARCH)

Solutions Architecture [Appendix E – Page 21] deals with the design and communication of high level architecture to enable the guidance of the development stages.
I had previously assessed myself as being level 5, and think that I am still a level 5 in this skill area. The reason for thinking that I have not developed to level 6 is because I do not believe that I have displayed *“consistency with specified requirements agreed with both external, and internal customers”*. However, I do think that I can still achieve the requirements for level 5: *“Produces detailed component specifications and translates these into detailed designs for implementation using selected products”.* I believe that I have displayed that skill in this report. To progress to level 6, I will aim to develop my system architecture design skills by examining how projects are designed in a professional environment and incorporate these practices into my own designs. I will also aim to get feedback from the project stakeholders to ensure that my design covers all key points that are required for the solution.

8.4 Summary of Learnings

From this project I have had the opportunity to developed my programming and project management skills. I have learnt how to organise and tackle a large scale problem and undertaking this project has given me an insight into the breadth of challenges offered by

developing drone software and an appreciation for the importance of software engineering practices.
Overall I have learnt and developed many skills throughout the completion of this project. I have had the opportunity to work with large open source projects and I have learned a lot about utilising online communities for gathering information and help when it comes to using the open source software. I have also uploaded my project code onto GitHub, which has made me realise the importance of organising software in a clear and decipherable manner. I have had the chance to complete a large project in Java which uses spatial data analysis.

These are useful experiences for my immediate development. However, much more interesting and applicable learnings that I have achieved from carrying out this project are “Double Loop Learning” [35] or “ Transferable Learning”.
I think the biggest learning that I can take away from this project is the importance of regular re-evaluation. This can be applied to any aspect of development, such as consistently re-evaluating the progress of a project and determining if the approach that was planned yesterday is still appropriate for the work being carried out today, or regularly re-evaluating myself: the skills I am learning, the knowledge that I am acquiring or my professional development. By being aware of this it makes it much easier to gain a deeper understanding of myself and my work, and how I should plan future projects to have the biggest impact on my development.

I hope to continue my personal development through undertaking challenging and unique projects within teams and individually, and constantly reflecting on myself and evaluating my work in the future as I start my Software Engineering career.

9 Support

9.1 Table of Abbreviations

Abbreviation	Full name	Meaning
CAA	Civil Aviation Authority	The British government body responsible for regulating aviation.
.csv	Comma separated values	A file type that consists of values separated by the ‘,’ character.
DTP	Drone Transportation Protocol	The code being developed for this project that combines the Communication Protocol and the Drone Control component.
FAA	Federal Aviation Administration	The American government body responsible for regulating aviation.
LoS	Line of Sight	When refereeing to operating a drone, to mean with the operator can see the drone. This is usually considered to be 400 feet.
RPAS	Remotely Piloted Aerial Systems	A drone, interchangeable with UAV.
SITL	Software in the Loop	Software that provides a way to simulate drones in a program.
tDTP	Test Drone Transportation Protocol	An implementation of a DTP that is design to run within the Virtual Testing Platform.
UAV	Unmanned Aerial Vehicle	A drone, interchangeable with RPAS.
VTP	Virtual Testing Platform	The software developed for this project that allows tDTPs to be simulated and validated.
3DR	3D Robotics	The company that produces the Solo drone and develops and maintains the DroneKit open source software suite.

Appendices

The appendix files provide information that is relevant to the report and can provide a deeper understanding of the subject matter discussed.
All appendices have been uploaded to PATS2 with the same file names given below.

Appendix A – Project Initial Plan

Appendix B – Amazon: Determining Safe Access with a Best-Equipped Best-Served Model for sUAS

Appendix C - Amazon: Revising the Airspace Model for the Safe Integration of sUAS

Appendix D – FCC: A Technology Comparison

Appendix E – SFIA 5 framework reference

Appendix F – Nathan Ahmad Year in Industry Report

9.2 References

[1] Civil Aviation Authority, “Statement on Heathrow drone incident,” 17 April 2016. [Online]. Available: https://www.caa.co.uk/News/Statement-on-Heathrow-drone-incident/. [Accessed 24 April 2016].

[2] A. P. Danica Kirka, “Drone collision with jet highlights growing aviation danger,” 18 April 2016. [Online]. Available: http://www.dailymail.co.uk/wires/ap/article-3546098/Drone-collision-jet-highlights-growing-aviation-danger.html. [Accessed 24 April 2016].

[3] Know Before You Fly, “Know Before You Fly,” 01 January 2015. [Online]. Available: http://knowbeforeyoufly.org/. [Accessed 24 April 2016].

[4] Civil Aviation Authority, “Small unmanned aircraft: Specific regulations about small drones,” 01 January 2015. [Online]. Available: https://www.caa.co.uk/Commercial-industry/Aircraft/Unmanned-aircraft/Small-unmanned-aircraft/. [Accessed 24 April 2016].

[5] Amazon, “Amazon Prime Air,” 29 November 2015. [Online]. Available: https://www.youtube.com/watch?v=MXo_d6tNWuY. [Accessed 24 April 2016].

[6] Facebook, “Connectivity Lab,” Facebook, [Online]. Available: https://info.internet.org/en/story/connectivity-lab/. [Accessed 24 April 2016].

[7] 3D Robotics, “DroneKit. Developer tools for drones,” 01 January 2015. [Online]. Available: http://dronekit.io/. [Accessed 24 April 2016].

[8] 3. Robotics, “Welcome to DroneKit-Python’s documentation!,” [Online]. Available: http://python.dronekit.io/. [Accessed 24 April 2016].

[9] 3. Robotics, “Setting up a Simulated Vehicle (SITL),” [Online]. Available: http://python.dronekit.io/develop/sitl_setup.html. [Accessed 24 April 2016].

[10] ArduPilot, “SITL Simulator (Software in the Loop),” [Online]. Available: http://ardupilot.org/dev/docs/sitl-simulator-software-in-the-loop.html. [Accessed 24 April 2016].

[11] Amazon, “Udi Nano Quad Electric Micro Quadcopter Orange A-U839-G,” UDI, 25 April 2016. [Online]. Available: https://www.amazon.co.uk/Electric-Micro-Quadcopter-Orange--U839-G/dp/B00MW3ZRAG/ref=sr_1_13?s=kids&ie=UTF8&qid=1461598832&sr=1-13.

[12] Currys, “3DR Solo SA11A Smart Drone - Black,” [Online]. Available: http://www.currys.co.uk/gbuk/smart-tech/smart-tech/smart-toys-and-gadgets/smart-toys/3dr-solo-sa11a-smart-drone-black-10135023-pdt.html. [Accessed 25 April 2016].

[13] B. N. Lincolnshire, “Drones and 'Robocrop' on show at Cereals 2015 farming event,” 11 June 2015. [Online]. Available: http://www.bbc.co.uk/news/uk-england-lincolnshire-33078857. [Accessed 26 April 2016].

[14] Civil Aviation Authority, “Small unmanned aircraft - Specific regulations about small drones,” [Online]. Available: https://www.caa.co.uk/Commercial-industry/Aircraft/Unmanned-aircraft/Small-unmanned-aircraft/. [Accessed 25 April 2016].

[15] World Drone Prix, “World Drone Prix,” [Online]. Available: http://www.worlddroneprix.com/. [Accessed 25 April 2016].

[16] Amazon, “Amazon Prime Air,” [Online]. Available: https://www.youtube.com/watch?v=MXo_d6tNWuY. [Accessed 25 April 2016].

[17] D. Thompson, “A world Without Work,” The Atlantic, [Online]. Available: http://www.theatlantic.com/magazine/archive/2015/07/world-without-work/395294/. [Accessed 26 April 2016].

[18] 3. Robotics, “Solo Specs: Just the facts,” [Online]. Available: https://3dr.com/solo-gopro-drone-specs/. [Accessed 26 Aoril 2016].

[19] BBC, “Drones build rope bridge that humans can cross - BBC News,” BBC, 27 September 2015. [Online]. Available: https://www.youtube.com/watch?v=DcuwCtDgVr0. [Accessed 26 April 2016].

[20] k. (https://github.com/karulis), “PyBluez,” [Online]. Available: https://github.com/karulis/pybluez. [Accessed 30 April 2016].

[21] R. M. a. G. Wahl, “wifi, a Python interface,” [Online]. Available: https://wifi.readthedocs.io/en/latest/index.html. [Accessed 30 April 2016].

[22] Amazon, “USB Bluetooth dongle,” [Online]. Available: https://www.amazon.co.uk/s/ref=sr_st_price-asc-rank?keywords=usb+bluetooth+dongle&rh=i%3Aaps%2Ck%3Ausb+bluetooth+dongle&qid=1462545867&sort=price-asc-rank. [Accessed 05 May 2016].

[23] GitHub, “PyBluez Contributions,” [Online]. Available: https://github.com/karulis/pybluez/graphs/contributors. [Accessed 30 April 2016].

[24] JetBrains, “The drive to develop Create Anything,” JetBrains, [Online]. Available: https://www.jetbrains.com/. [Accessed 03 May 2016].

[25] GitHub, “GitHub,” [Online]. Available: https://github.com/karulis/pybluez/graphs/code-frequency. [Accessed 05 May 2016].

[26] karulis, “pybluez README,” [Online]. Available: https://github.com/karulis/pybluez/blob/master/README.rst. [Accessed 05 May 2016].

[27] DroneKit, “Quick Start,” [Online]. Available: http://python.dronekit.io/guide/quick_start.html. [Accessed 05 May 2016].

[28] ArduPilot, “Setting up SITL on Linux,” [Online]. Available: http://ardupilot.org/dev/docs/setting-up-sitl-on-linux.html. [Accessed 05 May 2016].

[29] A. Brockmeier, “Things Known And Unknown,” [Online]. Available: http://www.projecttimes.com/articles/things-known-and-unknown.html. [Accessed 02 May 2016].

[30] C. Sandrock, “ Factors to consider when choosing a programming language,” 27 October 2010. [Online]. Available: http://negfeedback.blogspot.co.uk/2010/10/factors-to-consider-when-choosing.html. [Accessed 03 May 2016].

[31] JogAmp, “JOGL,” [Online]. Available: http://jogamp.org/. [Accessed 06 May 2016].

[32] J3D, “The Java 3D Community Site,” 09 June 2010. [Online]. Available: http://j3d.org/. [Accessed 06 May 2016].

[33] JavaPlot, “About JavaPlot,” [Online]. Available: http://javaplot.panayotis.com/. [Accessed 04 May 2016].

[34] R. L. Bogue, “Use S.M.A.R.T. goals to launch management by objectives plan,” Peter Drucker, [Online]. Available: http://www.techrepublic.com/article/use-smart-goals-to-launch-management-by-objectives-plan/. [Accessed 01 May 2016].

[35] M. K. Smith, “Chris Argyris: theories of action, double-loop learning and organizational learning,” [Online]. Available: http://infed.org/mobi/chris-argyris-theories-of-action-double-loop-learning-and-organizational-learning/. [Accessed 02 May 2016].

[36] C. A. Authority, 24th March 2016. [Online]. Available: https://www.caa.co.uk/Commercial-industry/Aircraft/Unmanned-aircraft/Unmanned-Aircraft/.

[37] N. Ahmad, “Initial Plan: Drone Transportation Protocol for High Traffic Areas,” Cardiff, 2016.

[38] P. Community, “XBee 2.2.3,” [Online]. Available: https://pypi.python.org/pypi/XBee. [Accessed 30 April 2016].