**Group Member NAMES: Ryan Sauer, Demetrius Johnson,**

**Jonathan Schall & Olivia Pellegrini**

SKY SOCKET PROJECT MANAGEMENT PLAN

# Table of Contents

[**Table of Contents**](#_bm6v7e2ausk7) **1**

[**1.0 Introduction**](#_sk9bwhy0ve3d) **2**

[1.1 Project scope](#_7vmmb8zfb3cl) 2

[1.2 Major software functions](#_cvtwaw6qdbzg) 2

[1.3 Performance/Behavior issues](#_aw8zmivdga77) 2

[1.4 Management and technical constraints](#_y0g8cmgxeerw) 2

[**2.0 Project Estimates**](#_p6hcn182c9cy) **2**

[2.1 Historical data used for estimates](#_rmjgcircurq5) 2

[2.2 Estimation techniques applied and results](#_txnk6b3lr00h) 2

[2.2.1 Estimation technique m](#_er3ed8svmzc) 2

[2.2.2 Estimate for technique m](#_nia73ij8hf1z) 2

[2.3 Reconciled Estimate](#_lfsj0cl5i4t8) 2

[2.4 Project Resources](#_vp96fv21euvm) 2

[**3.0 Risk Management**](#_sl0vm5ui508l) **3**

[3.1 Project Risks](#_gvrevkpwb8ly) 3

[3.2 Risk Table](#_h4oq1j24xpgx) 3

[3.3 Overview of Risk Mitigation, Monitoring, Management](#_s1xateyk5pd5) 3

[**4.0 Project Schedule**](#_dbqzd7iaqn6b) **3**

[4.1 Project task set](#_o4m8nb3bfr0e) 3

[4.2 Functional decomposition](#_4vprn46iz5st) 3

[4.3 Task network](#_6qjyqaf0kbqh) 3

[4.4 Timeline chart](#_590mmfbxq4i0) 3

[**5.0 Staff Organization**](#_kawypjft11wu) **3**

[5.1 Team structure](#_id3r8v4q1ulm) 4

[5.2 Management reporting and communication](#_12d4u1f5he1o) 4

[**6.0 Tracking and Control Mechanisms**](#_j432uebnalhh) **4**

[6.1 Quality assurance and control](#_d6gpd3snzl60) 4

[6.2 Change management and control](#_184n8ebx3qd) 4

[**7.0 Appendix**](#_aassul5657lc) **4**

# 1.0 Introduction

The main purpose of Sky Socket is to create a prototype for researching and optimizing the analysis and communication between a car and a drone for autonomous driving. The goals of this project are:

* Create a working prototype of a car and drone
* Implement an efficient analysis and communication method between the drone and car
* Optimize the communication and analysis methods on the car and drone.

## 1.1 Project scope

The following requirements were given for Sky Socket:

* An image recognition algorithm running on the drone for detecting a predefined object
* A manual/automatic control mechanism for the car
* A manual/automatic control mechanism for the drone
* A multiple communication protocols for use between the drone and car
* An ascii based log system to store data points about the drones battery life, bandwidth, and latency.
* A way to extract drone data logs from the raspberry pi
* A way to extract car data logs from the raspberry pi

## **1.2 Major software functions**

1. Drone-Car communication network interface
2. Vision data processing in OpenCV Python
3. Drone autonomous flight programming
4. Car autonomous driving programming
5. Car and Drone data metric collection and synthesis from sensory information

## **1.3 Performance/Behavior issues**

Communication optimization:

The client would like several differing communication protocols that the drone and car will use to transfer data. This is to help in solving an optimization problem for the communication time and effectiveness of the system. Thus we will have to create/implement several protocols that differ in bandwidth limitations, response time, and power draw.

Drone Movement:

Given a functional model is created, our client is also interested in having a dynamic environment for testing that would require the drone to autonomously follow the car. This is extended functionality to the scope that would require extra working hours that we may not have and has been agreed upon to not be required but potentially added with extra time.

## **1.4 Management and technical constraints**

*Time:*

We have 5-6 months to complete all documentation, software, hardware, and tests. We have many goals and some that are not required as we realize time may be too limiting. We also have 4 members with jobs, other courses, and responsibilities and thus conflicting schedules that will limit collaboration.

*Access:*

In order to test our prototype, we must do so at certain times in certain spaces with certain clearance as drones are highly regulated on the University of Michigan - Dearborn campus and by law enforcement. This will force us to do scheduled and limited testing.

# **2.0 Project Estimates**

## **2.1 Historical data used for estimates**

We obtained the following data from the Bureau of Labor Statistics who’s data is from May 2021:

<https://www.bls.gov/oes/current/oes_stru.htm>

| **Job Title** | **Avg Hourly Pay** |
| --- | --- |
| Network and Computer Systems Administrator | $43.87 |
| Software Developer (can act as project manager) | $58.17 |
| Network Architect | $58.01 |
| Computer Programmer | $46.46 |
| Average of above roles for this project | $51.63 |

Here is a table of our time analysis based on the documentation we have already completed, including the remaining documentation that will soon be completed. This will be factored into our estimates for the total project.

**Table of Documentation Working Time**

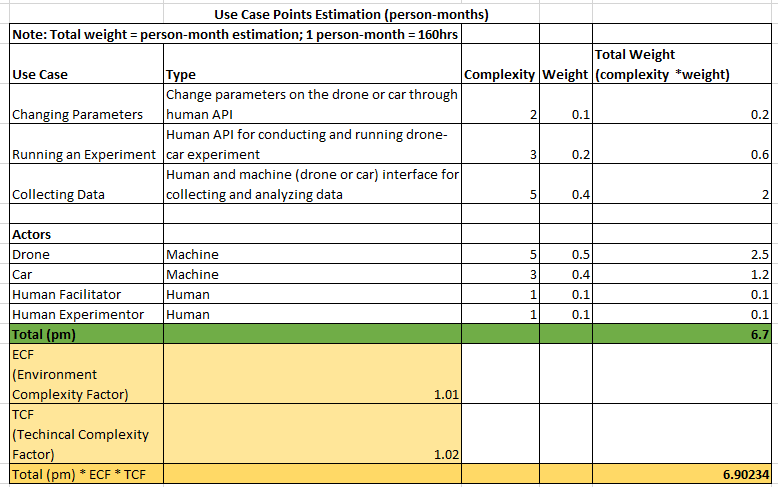
| Document Name | Number of Main Sections | Estimated Hrs/Section | Total Hours |
| --- | --- | --- | --- |
| RMMM | 4 | 2.2 | 8.8 |
| Project Plan | 7 | 2.2 | 15.4 |
| Use Case Document | 9 | 3 | 27 |
| Project Timeline Documents | N/A | 3 | 3 |
| Use Case Presentation | N/A | 3 | 3 |
| RMMM and Project Plan Presentation | N/A | 3 | 3 |
| Total Hours for All Documents | 20 |  | 60.2 |

## **2.2 Estimation techniques applied and results**

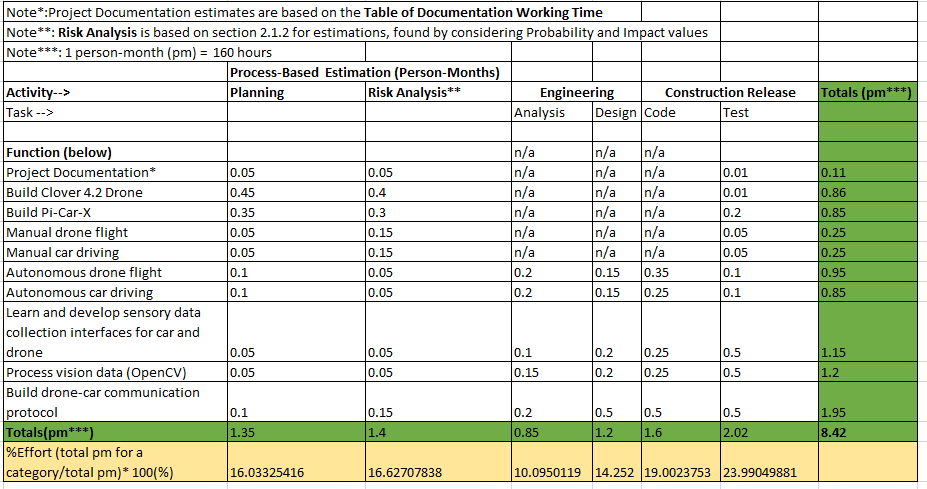
**Source: Roger S. Pressman\_ Bruce R. Maxin - Software Engineering\_ A Practitioner’s Approach-McGraw-Hill Education (2014)(chapters 33.6 and 33.7)**

Given our level of access and amount of time for developing estimations for the project, we will employ the Use Case Points and Process-Based estimation techniques. Then we employ the COCOMO FP technique using an online calculator resource.

## **2.2.1 Estimation technique** 1: Use Case Points Estimation

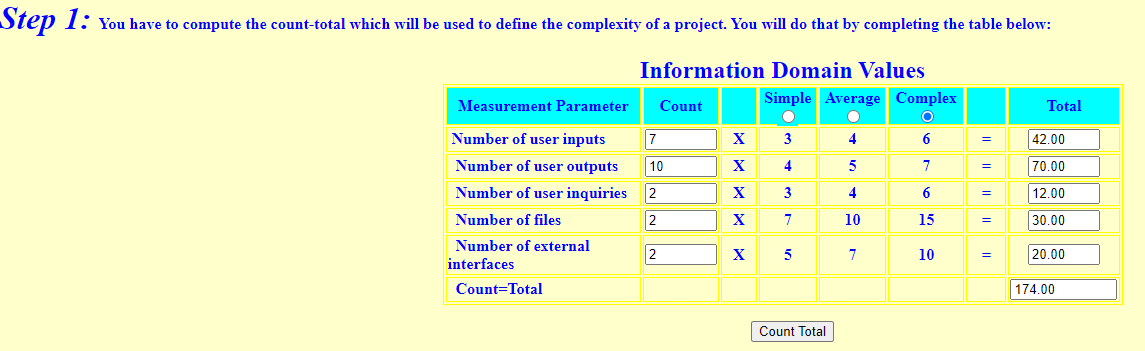
****

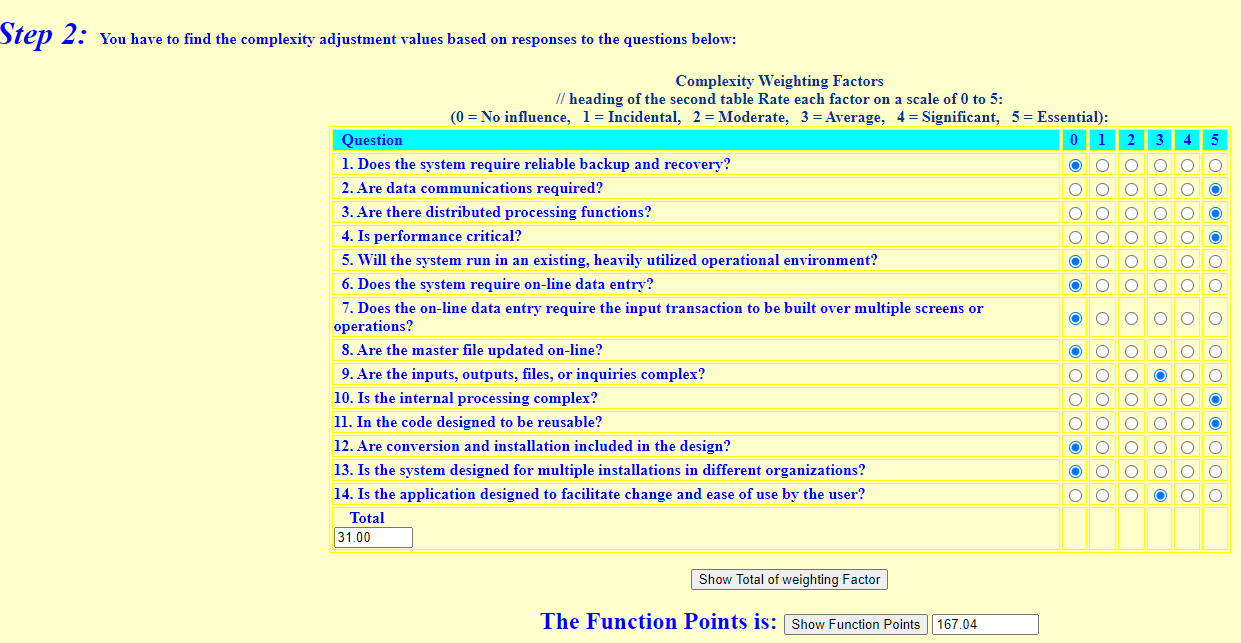
## **2.2.2 Estimat**ion **technique** 2: Process-Based Estimation

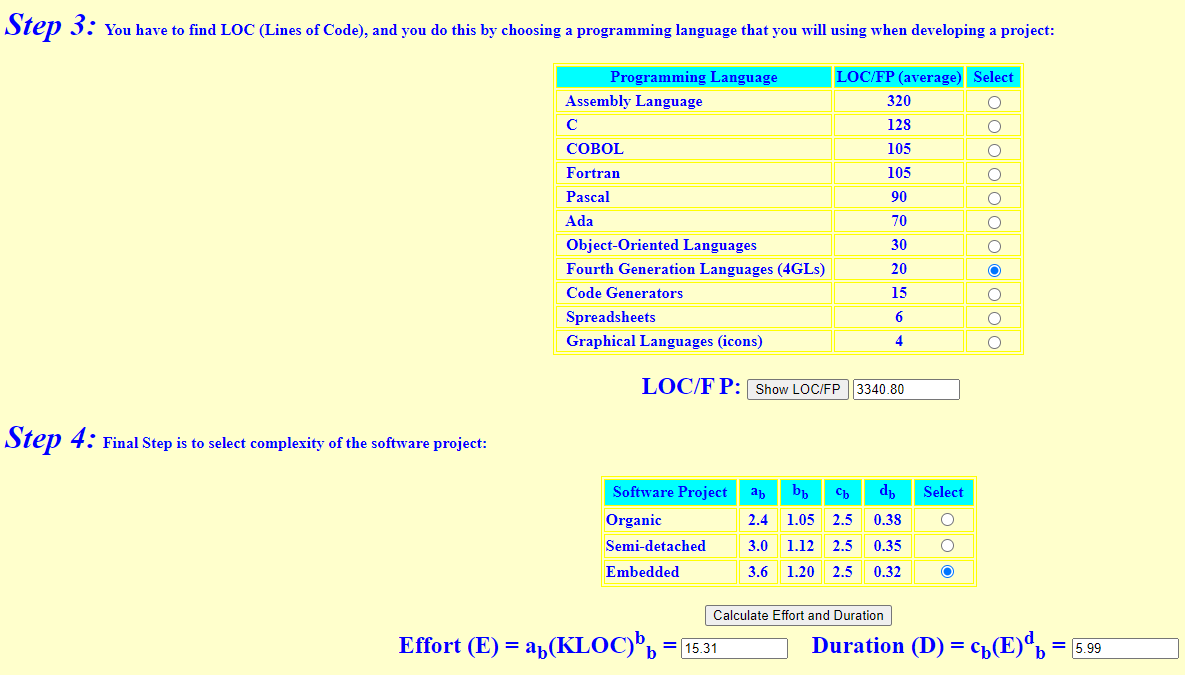
****

## 2.2.3 Estimation Technique 3: COnstructive COst MOdel (COCOMO) Function Points

Source of tool used: <http://groups.umd.umich.edu/cis/course.des/cis525/js/f00/gamel/cocomo.html>







**Result: 5.99 Person-months**

## **2.3 Reconciled Estimate (combined estimates** from both techniques**)**

* As a summary of the results of our estimation from the techniques:
  + Use Case Points person-month estimation: 6.9
  + Process-Based person-month estimation: 8.42
  + COCOMO FP estimation: 5.99
  + **Average: 7.10 person-months**
* Based on the hourly average hourly pay of the technical roles necessary for this project, then monetary cost for the project is estimated to be
  + 7.10pm \* 160hr/pm \* $51.63 = **$58,651 TOTAL PROJECT COST**
* Special Note, we estimate that on average among 4 team members, we can each work on average 8hrs per week, which equates to (0.2pm = 32hrs per month).
  + 4members \* 0.2pm/member \* 7months for senior design = 5.6pm that our team estimates we can allocate from our time.
  + Thus, based on this estimate, we would need to work above average at some points in the project, particularly during the summer (senior design 2).
* Senior design 1 and 2 combined leave about 7 months to complete the project. Our estimation of 7.10pm means we definitely face the possibility of shortcomings in the project including failing to finish it, including considering that we estimate we can only contribute 5.6pm as a team since we as students do not have a full 40 hours (160hrs per month = 1 pm) to commit each week - and person-months are based on full-time (40hr/week) workers.

## **2.4 Project Resources**

* **Raspberry Pi model 4B**
  + <https://www.raspberrypi.com/products/raspberry-pi-4-model-b/>
  + Used as the onboard computing device for both the car and the drone (each has their own Raspberry Pi computing device) for autonomous driving/flying by taking input from the sensors and processing the information, and also for communications (networking) with between devices (vehicles), such as wireless communications, i.e. WIFI.
* **Raspberry Pi Ai Car Kit (PiCar-X) for Intermediate**
  + <https://www.sunfounder.com/products/picar-x>
  + <https://docs.sunfounder.com/projects/picar-x/en/latest/introduction.html>
  + Used as the physical device for the car including motors, frame, driving mechanism, and sensors.
  + The Raspberry Pi OS imager should be used to image the sd card that will serve as the nonvolatile memory unit the Raspberry Pi computer of the car:
    - <https://www.raspberrypi.org/software/>
  + Here is the Repository that is cloned onto the Raspbian OS image of the Raspberry Pi; it will contain all of the installation files needed to program and control the car:
    - <https://github.com/sunfounder/robot-hat>
* **Clover Drone 4.2** 
  + <https://clover.coex.tech/en/>
  + Used as the physical device for the drone including motors, frame, propellers, sensors, Electronic Speed Controllers (ESC), GPS, etc.
  + Includes Pixracer R15 Mini Pixracer Autopilot Xracer FMU V4 V1.0 PX4 Flight Controller
    - <https://docs.px4.io/main/en/flight_controller/pixracer.html>
* **Python Programming Language**
  + <https://www.python.org/>
  + We will use the Python programming language for both the car and the drone.
* **OpenCV-Python Library**
  + <https://pypi.org/project/opencv-python/>
  + This is a Python vision analysis library that has been adapted from a library originally written for C++
  + We will use it to analyze vision data collected from cameras on both the car and the drone.
* **Raspbian OS builds with Linux Kernel**
  + <https://www.kernel.org/>
  + Both the car and the drone have their own onboard computing device (Raspberry Pi model 4B) with a custom modified version of the Raspbian Operating System image that uses the Linux Kernel.
  + Additionally, the Clover 4.2 Drone uses the ROS robotic framework used for advanced robotic distributed systems.
    - <https://wiki.ros.org/>
  + Here is the image used for the Clover 4.2 Drone Raspberry Pi computer:
    - <https://github.com/CopterExpress/clover/releases/tag/v0.23>
    - Image features:
      * Raspbian Buster
      * [ROS Noetic](http://wiki.ros.org/noetic)
      * Configured networking
      * OpenCV
      * [Mavros](http://wiki.ros.org/mavros)
      * Periphery drivers for ROS ([GPIO](https://clover.coex.tech/en/gpio.html), [LED strip](https://clover.coex.tech/en/leds.html), etc)
      * Aruco\_pose package for marker-assisted navigation
      * Clover package for autonomous drone control
* **Q Ground Control**
  + <https://docs.qgroundcontrol.com/master/en/>
  + This is an open source software used to communicate with and calibrate and configure a drone’s flight controller firmware. We will use this to calibrate the drone and manage the flight controller’s parameters and how the flight system of the drone uses and responds to sensor data.
  + Here is the firmware image used for our flight controller:
    - <https://github.com/CopterExpress/Firmware/releases/tag/v1.8.2-clover.13>
* **Clover Drone Simulation virtual machine (VM) image**
  + <https://github.com/CopterExpress/clover_vm>
  + This is the virtual machine image used to run the simulation software used to simulate programmed autonomous flights for the Clover 4.2 Drone.
  + Image contains:
    - Ubuntu 20.04 Focal.
    - ROS Noetic.
    - PX4 autopilot, QGroundControl.
    - Preinstalled [Clover](https://github.com/CopterExpress/clover) and Clover simulation packages.
    - Shortcuts for running Clover simulator.
    - VSCode.
    - Useful robotics-related software.
* **Drone Simulation Environment (Using Gazebo software)**
  + The simulation environment is based on the following components: [Gazebo](http://gazebosim.org/), a state-of-the-art robotics simulator;
    - <http://gazebosim.org/>
  + [PX4](https://px4.io/), specifically its SITL (software-in-the-loop) components;
    - <https://px4.io/>
  + [sitl\_gazebo](https://github.com/PX4/sitl_gazebo)  package containing Gazebo plugins for PX4;
    - <https://github.com/PX4/sitl_gazebo>
  + ROS packages and Gazebo plugins
  + **Note:** all of the above components are installed on the Clover Drone Simulation VM in order to do simulation programming without the Raspberry Pi on board computing device of the drone. This allows for programming and simulation without needing the physical drone present.
* **Etcher - Flashing Software**
  + <https://www.balena.io/etcher>
  + This is the software used to flash the micro-SD card with the respective OS (drone or car) used as the nonvolatile memory unit for the Raspberry Pi computers.

# **3.0 Risk Management**

Below are the risks that may be encountered during the project and how our team will manage these risks.

**3.1 Project Risks**

**Equipment Risk**

If some of the major equipment used to create the model fails/breaks during or shortly after development. The two main pieces of equipment necessary are the drone and the car. Either failing would cause a massive problem as there wouldn’t be the necessary tools required to build the model.

**Employee Risk**

If one or more of the team members doesn’t put in the required amount of effort to complete the project over the course of the year. The project relies on everyone doing their part and it is not easy for the team to pick up the slack of another team member.

There is also the risk of losing one of our team members due to unforeseen circumstances, making the project extremely difficult to complete.

**Quality Risk**

There is a chance that the model the team makes is not up to the client’s standards and is rejected at the end of the project. The quality of the model may be lacking in a few areas:

* Drone battery may not last long enough
* Too much latency in communicating between car and drone
* Lack of consistency in detecting objects

**Customer Risk**

There is a chance that the client loses interest in the project and/or the team.

**Policy Risk**

Before using the drone in a new area the team needs to make sure they are permitted to use it. If they are unsure, they should speak with the client before proceeding and operating the drone.

## **3.2** **Risk Table**

**Probability and Impact for Risk m**

**The following is the sorted version of the above table by probability and impact:**

| **Category** | **Risks** | **Probability** | **Impact** |
| --- | --- | --- | --- |
| Equipment Risk | Drone failing | 50% | 1 |
| Equipment Risk | Car failing | 30% | 2 |
| Policy Risk | Breaking campus policy | 15% | 1 |
| Employee Risk | Member(s) slacking | 20% | 3 |
| Customer Risk | Client loses interest | 5% | 1 |
| Employee Risk | Losing member | 5% | 2 |

Impact values:

1= catastrophic

2= critical

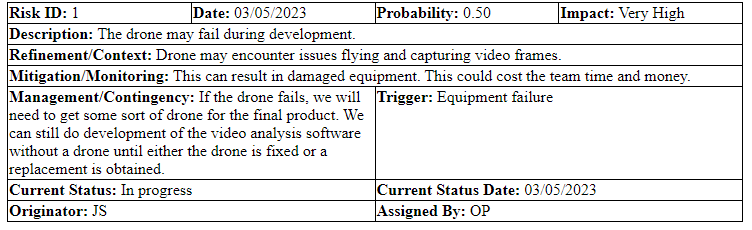
3= marginal

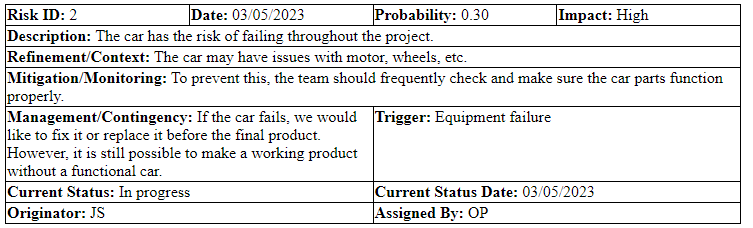
4= negligible

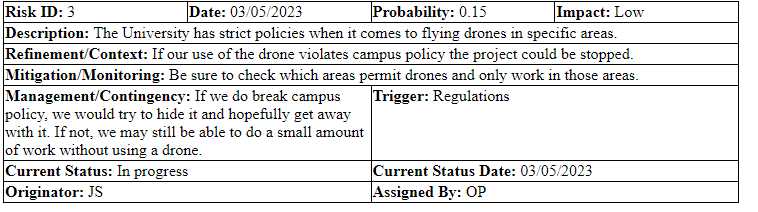
(Information in 3.1 and 3.2 taken from [RMMM](https://docs.google.com/document/d/1eSKZZmf55wZ8s7j0sWaZZmcPMFLtaNlcUYi8I8vuBi4/edit?usp=sharing) document)

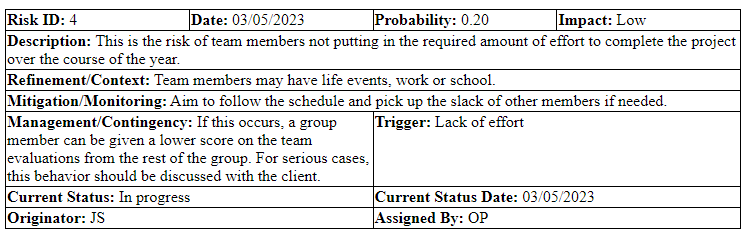
## **3.3 Overview of Risk Mitigation, Monitoring, Management**

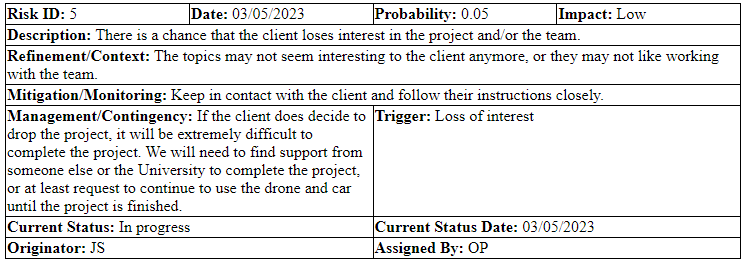
Below is the Risk Information Sheet (RIS) for each identified risk based of the risk presented in the RMMM document.

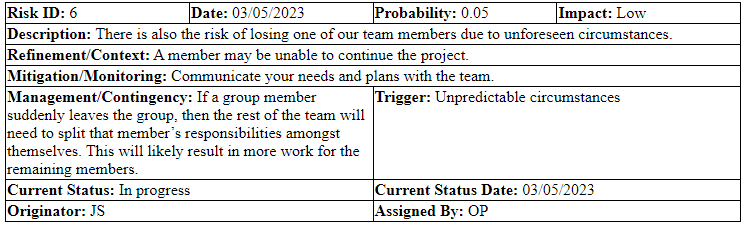










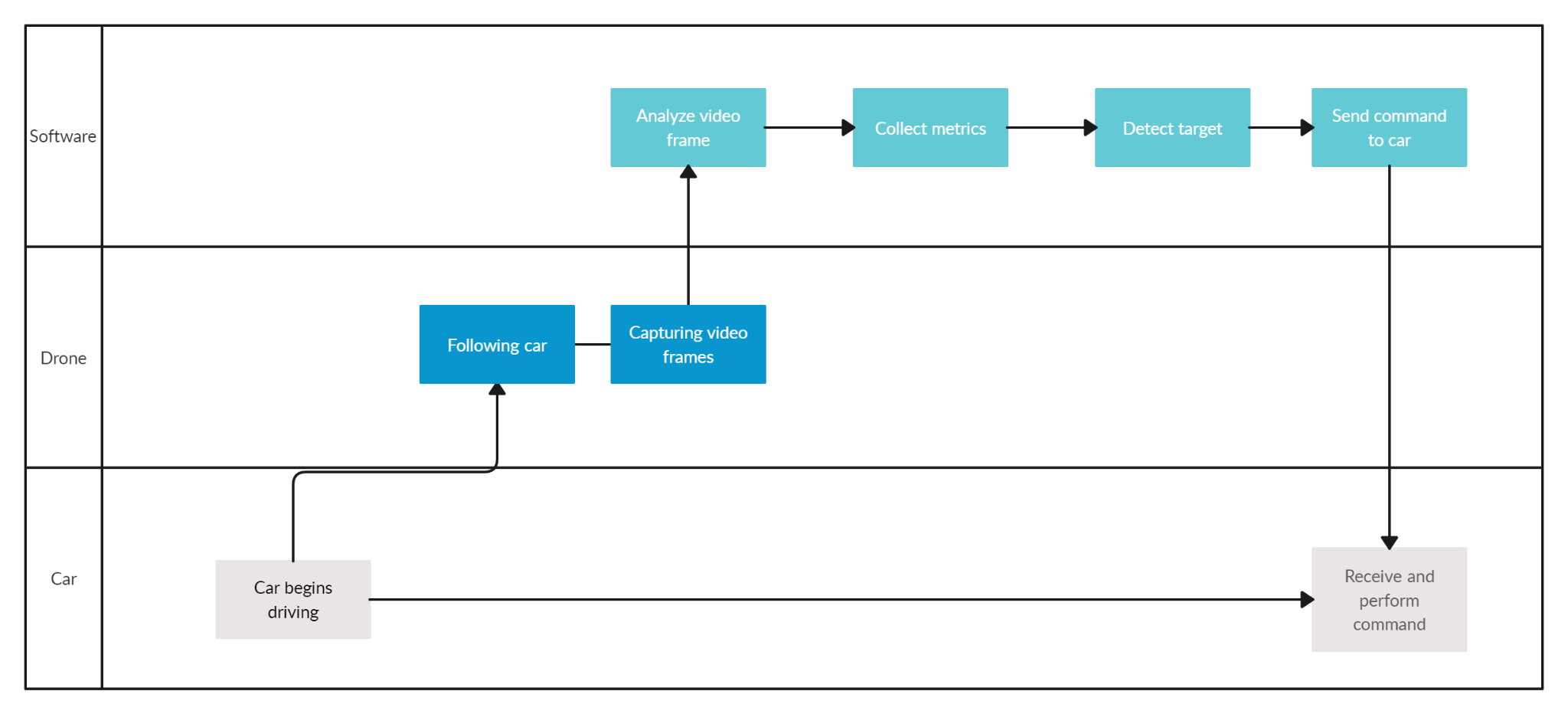


# **4.0 Project Schedule**

This section will address the project tasks and shows the output of the project scheduling tool.

## **4.1 Project task set**

This process model is showing the sequence of events within our project:



Process framework activities show how we will successfully complete all tasks:

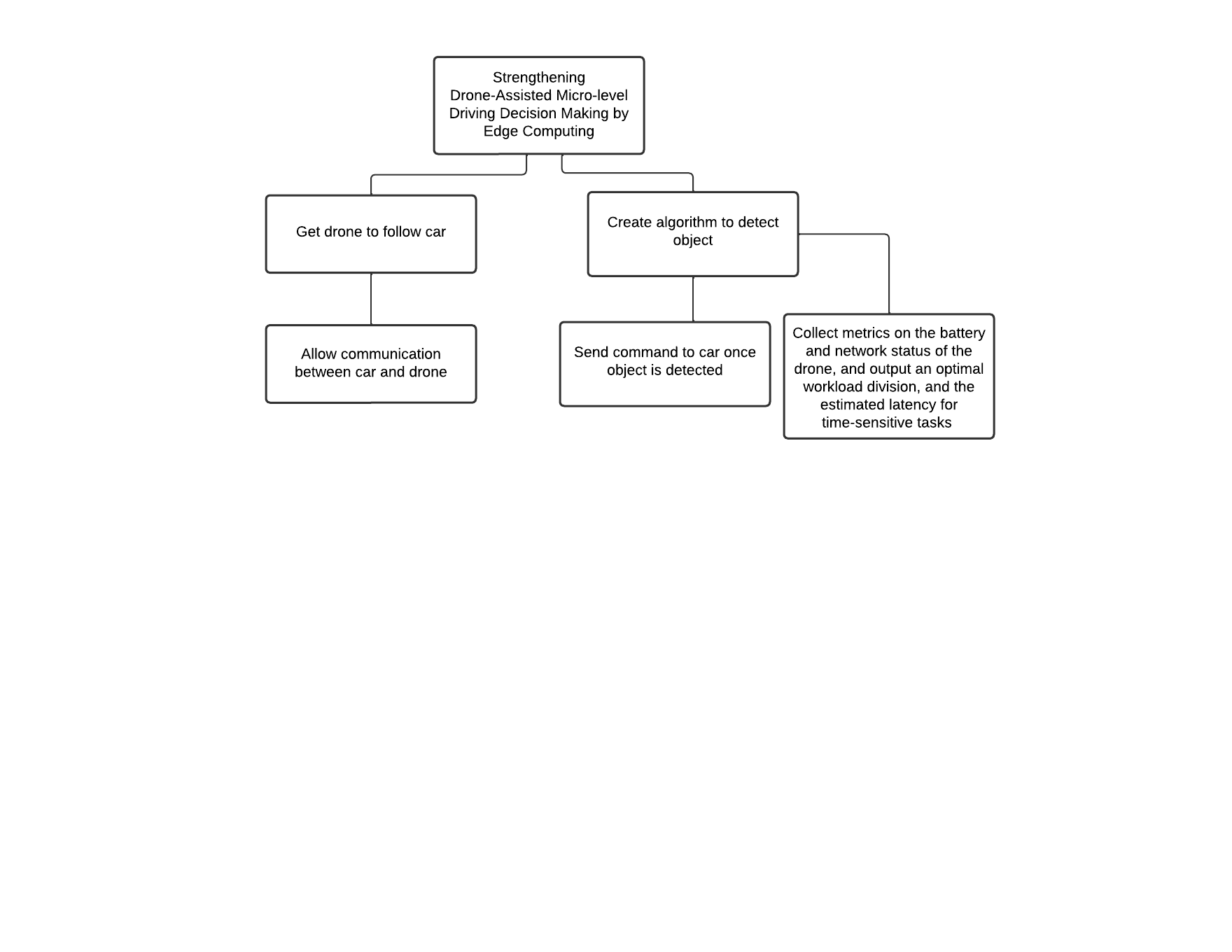
* **Communication**: Our team must meet with the client bi-weekly to guarantee we can meet his expectations of the project.
* **Planning**: We have created schedules to help keep track of our work and what needs to get done.
* **Modeling**: Our team has decided how the car, drone and software should function.
* **Construction**: To help this project function properly without bugs, we must thoroughly test our code and fix issues we come across.
* **Deployment**: During this phase, we will need to spend time with the client and get feedback on what can be improved.

The task set shows the deliverables and milestones created by our team.

| **Stage of Development** | **Stage Completion Date** | **Deliverable** | **Deliverable Completion Date** |
| --- | --- | --- | --- |
| Planning | 03/14/2023 | 1. Review preliminary software specification 2. Develop functional specification 3. Milestone | 1. 03/14/2023 2. 03/14/2023 3. 03/14/2023 |
| Requirements Definition | 03/14/2023 | 1. Develop prototype based on functional specification 2. Review functional specification 3. Develop Testing Environment and Metric Collection 4. Milestone | 1. 03/14/2023 2. 03/14/2023 3. 02/27/2023 4. 02/27/2023 |
| Programming for image processing | 08/13/2023 | 1. Learn OpenCV library 2. Create algorithm for image detection 3. Collect metrics 4. Finish all testing 5. Milestone | 1. 03/10/2023 2. 03/31/2023 3. 03/31/2023 4. 08/13/2023 5. 08/13/2023 |
| Car and drone communication | 03/28/2023 | 1. Successfully manually control car with Web Control Interface 2. Develop communications between drone and car 3. Milestone | 1. 02/28/2023 2. 03/28/2023 3. 03/28/2023 |
| Integration and Testing |  | 1. Test module integration 2. Modify code 3. Re-test modified code 4. Results evaluation 5. Milestone | 1. 07/14/2023 2. 07/20/2023 3. 07/28/2023 4. 08/14/2023 5. 08/14/2023 |
| Acceptance and deployment | 08/14/2023 | 1. Get feedback from Prof. Song 2. Evaluate testing information 3. Milestone | 1. 08/05/2023 2. 08/11/2023 3. 08/14/2023 |

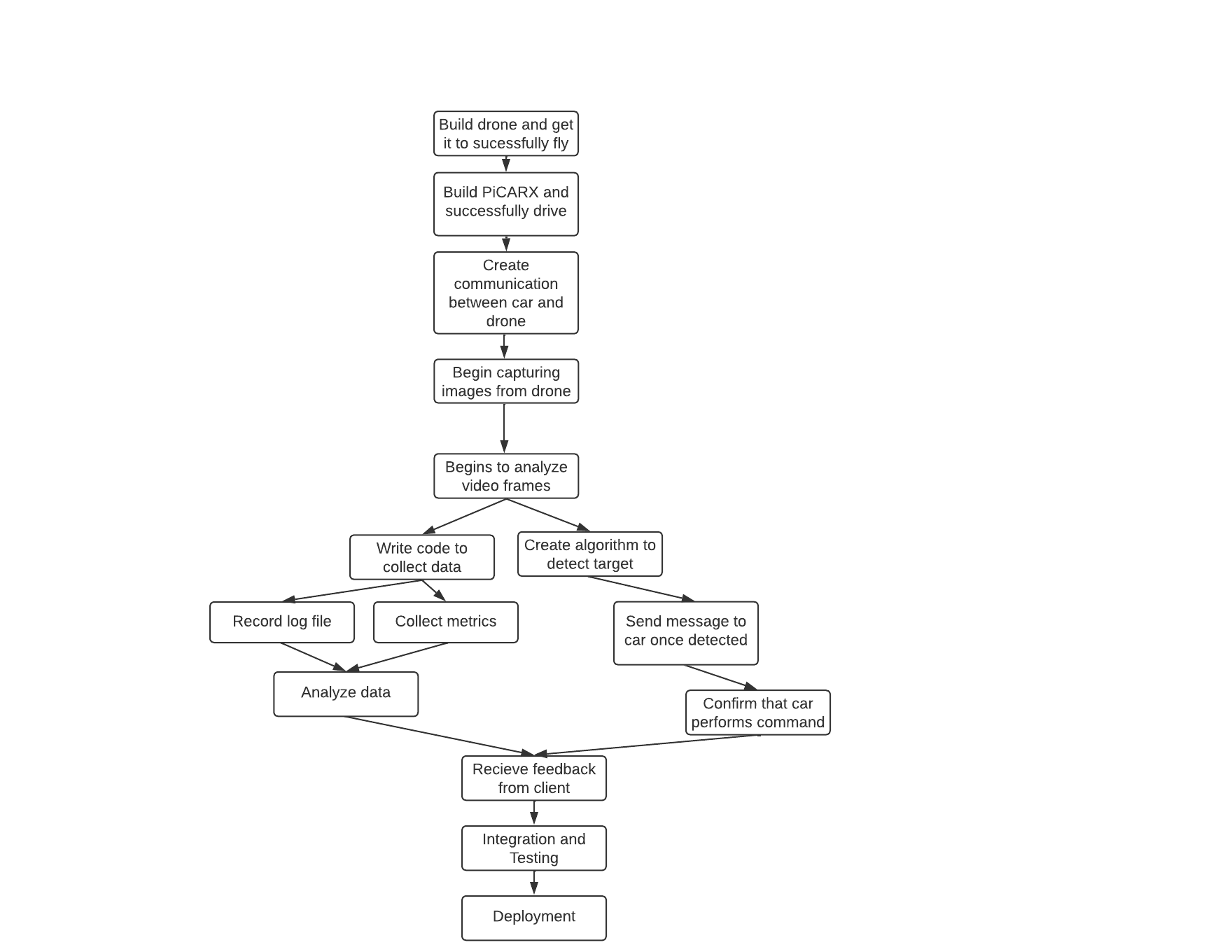
## **4.2 Functional decomposition**

Functional decomposition is used to help break down the function of their project into smaller tasks.



## **4.3 Task network**

This dependencies diagram displays the sequence in which tasks need to be achieved for this project to be completed on time.



## **4.4 Timeline chart**

Below is document that shows the timeline of the project and each task assigned to team members:

* [**https://docs.google.com/spreadsheets/d/1Np0KL0MUnTC7GoU5v01r7UdblBakfGlUsDKDI06vU\_Y/edit?usp=sharing**](https://docs.google.com/spreadsheets/d/1Np0KL0MUnTC7GoU5v01r7UdblBakfGlUsDKDI06vU_Y/edit?usp=sharing)

# **5.0 Staff Organization**

Due to the small team size and the technical skills needed to complete the different tasks in this project, we have delegated roles based on the technical needs to allow for specialization. We will also have some duties that aren’t particular to any individual and will have more collaboration.

## **5.1 Team structure**

**Network Architect and Administrator: Ryan Sauer**

* P2P communication development
* Network deployment and maintenance
* Data log development
* Data analysis

**Software Developer: Olivia Pellegrini & Jonathan Schall**

* Item detection algorithm development
* Software quality assurance / testing
* Sensor software development

**Vehicle Administrator and Developer: Demetrius Johnson**

* Hardware quality assurance
* Vehicle maintenance
* Vehicle control development

**Shared Responsibilities:**

* Documentation
* System testing
* Class reports
* Scheduling

*Note: Due to small team size and limited availability, team members may help other roles with duties in varying amounts.*

## **5.2 Management reporting and communication**

Our schedule system will be managed through Jira for regular documentation requirements. Our progress will then be transferred to a google drive folder for distribution to professors, our client, and ourselves.

We are currently using a combination of regular email, Zoom, and in person meetings with our client to keep them updated and informed on our progress and to gain feedback. For within our team, we are using Google Chat to communicate.

For version control, we will be using GitHub to monitor our software changes and collaborate.

# **6.0 Tracking and Control Mechanisms**

**For this section, we used** Software Engineering: A Practitioner’s Approach: Roger S. Pressman Bruce R. Maxin (2014)(chapter 21.1 and 21.2)

## **6.1 Quality assurance and control**

We will maintain quality assurance and control through:

1. Weekly meetings with the client.
2. Weekly meetings as a team.
3. Making sure that we check off items on the product backlog.
4. Accountability through team members completing their respective tasks.
5. Staying on schedule so that we have enough time to produce quality products throughout the project lifecycle.

## **6.2 Change management and control**

1. We will always consult with the client via a meeting before making any major project changes, such as downgrading or upgrading the final product, etc.
2. We have a project manager role that will serve to keep everyone on track with the respective tasks and make sure that members do not overstep their boundaries and make changes in areas of the project that are not assigned to them.
3. If changes are made, even if it is in a member's area of responsibility, they will still be documented and communicated to the rest of the team and to the client.

# **7.0 Appendix**

## 7.1 Traceability Matrix

| **ID** | **Use Case** | **Requirements** | **Priority** | **Depends on (ID)** |
| --- | --- | --- | --- | --- |
| **1** | **Run an Experiment** | * **Car driving** * **Drone video** * **Drone following car** | **HIGH** |  |
| **2** | **Retrieve and Analyze Data** | * **Develop API to collect data** * **Route the data to another API for analysis** | **HIGH** | **1,4** |
| **3** | **Change Car and Drone Parameters** | * **Develop API to change variable values** | **MEDIUM** | **1** |
| **4** | **Collect/track battery voltage of car and drone** | * **System voltage measurement interface** | **MEDIUM** | **1** |

## 7.2 References

* Bureau of Labor Statistics: <https://www.bls.gov/oes/current/oes_stru.htm>
* Software Engineering: A Practitioner’s Approach: Roger S. Pressman Bruce R. Maxin (2014)(chapter 33.6 and 33.7, 21.1, and 21.2)
* COCOMO Function Point Estimation Model
  + <http://groups.umd.umich.edu/cis/course.des/cis525/js/f00/gamel/cocomo.html>