Requirements Definition

Tello Drone

1. Introduction and Context

The purpose of this project is to connect to a drone and be able to control it from the computer. To communicate between the drone and the computer a socket will be open to allow data transfer. Once the Communication Center (aka Computer) and the drone have a connection the Communication Center will direct the drone on specific missions. To start off there will be three pre-programmed missions. From the keyboard you type number “1”,”2”, or “3” to command the drone for the specific mission. If the connection is lost at any time the Tello drone has a 15 second window before is auto lands. This is built into the drone and the the Communication Center. The main function will have the objects needed to control the drone. There will be a CommunicationCenter that relays messages to the drone. The individual Missions are inherited from a virtual Mission class so that each mission has to have its own mission.

2. Users and Their Goals

Types of users

- **Communication Center** (See Figure)
    - Goals: Be able to control the drone and command drone to fly specific missions.
    - Optional Goal: Be able to free fly the drone from the computer.
- **Drone** (See Figure)
    - Goals: Accept command given from the Communication Center and pass info back to the Communication Center depending if there was an “error” or if the command was “ok”
- **MissionCollection** (See Figure)
    - Goals: Hold a collection of all the different missions
- **Mission** (See Figure)
    - Goals: This is a virtual class that will be used for each mission

- **Mission1** (See Figure)
    - Goals: This is a specific mission inherited from the Mission
- **Mission2** (See Figure)
    - Goals: This is a specific mission inherited from the Mission
- **Mission3**(See Figure)
    - Goals: This is a specific mission inherited from the Mission
- **Socket Connection** (See Figure)
    - Goals: Pass information from the computer to the drone and allow the drone to pass information back though WIFI.

3. Functional Requirements

1. Connect to Drone
2. The Communication center both need to send and receive messages.

4. Non-functional Requirements

a. Application-level logic will be tested .

b. Unit tests should cover 100% of logic classes

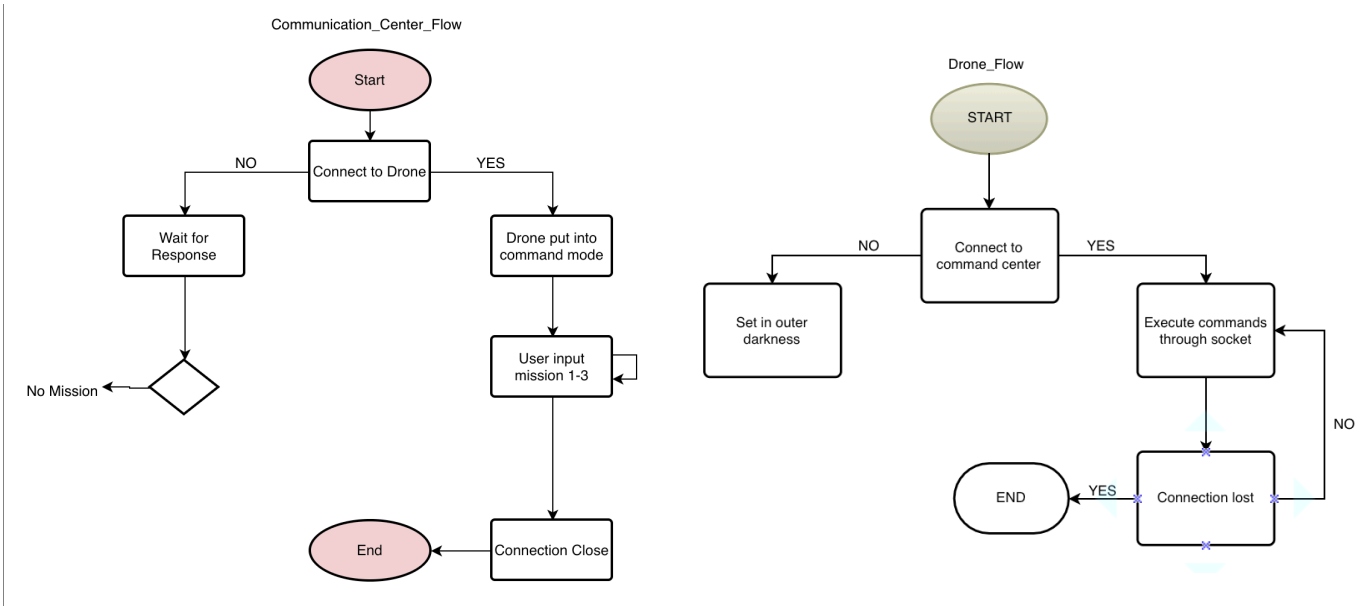
c. UX tests will be tested by me, so that the software design is working as intended (not including all possible fringe cases)

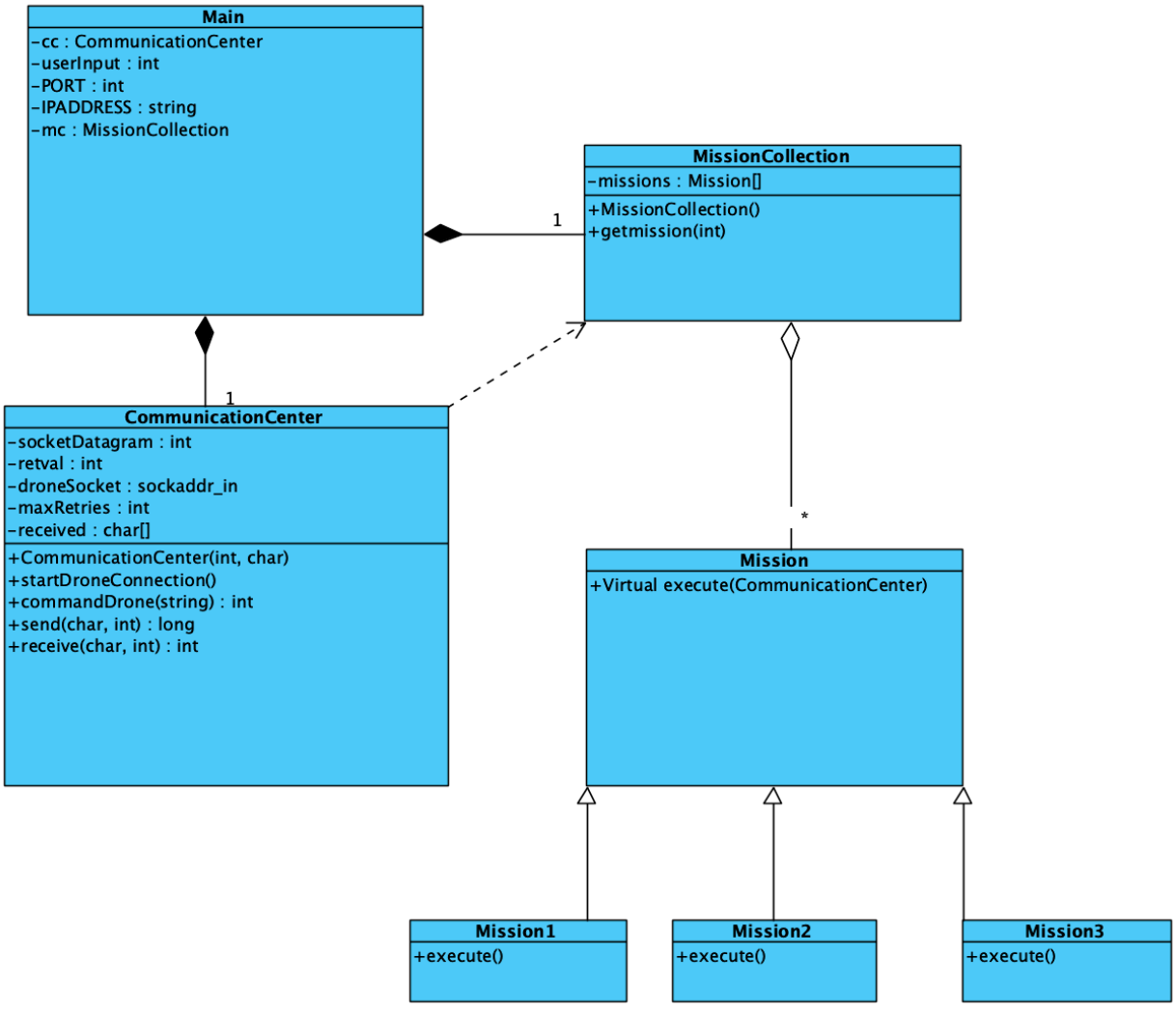
5. Insight into construction the code.

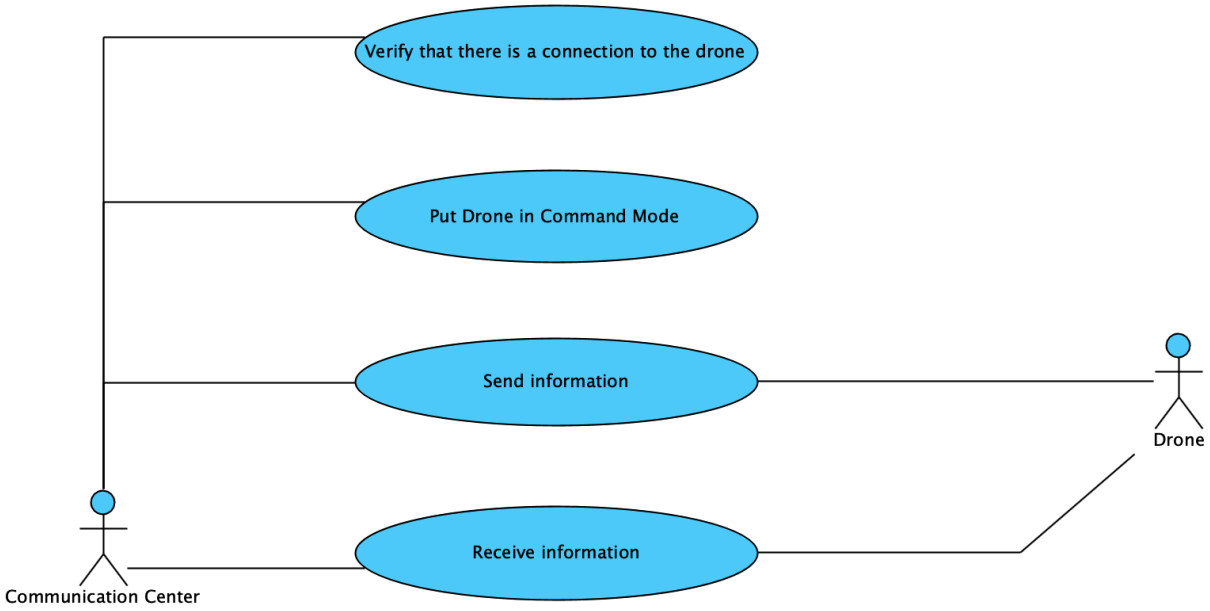
When the code was getting started it we difficult to decide the the best approach of development. At first I wanted to make the UML diagram form around the Drone but quickly determined that it was not such a great idea. I came to the conclusion that the UML needed to be centered around come sort of command center so I made a “Communication Center” and allowed this class to control the drone.

One of the biggest challenges that I had was the integration of the unit tests. It took quite a while to decide what unit test platform I wanted to use and even longer to figure out how it was installed and properly integrated into Cmake. There is still that I need to learn about “Catch 2” but now that everything is integrated it is much easier to start running forward.

6. Diagrams


Communication_Center_Flow
Start
Connect to Drone
NO
YES
Wait for Response
Drone put into command mode
No Mission
User input mission 1-3
End
Connection Close
Drone_Flow
START
Connect to command center
NO
YES
Set in outer darkness
Execute commands through socket
NO
END
YES
Connection lost


Main
-cc : CommunicationCenter
-userInput : int
-PORT : int
-IPADDRESS : string
-mc : MissionCollection
MissionCollection
-missions : Mission[]
+MissionCollection()
+getmission(int)
1
1
CommunicationCenter
-socketDatagram : int
-retval : int
-droneSocket : sockaddr_in
-maxRetries : int
-received : char[]
+CommunicationCenter(int, char)
+startDroneConnection()
+commandDrone(string) : int
+send(char, int) : long
+receive(char, int) : int
*
Mission
+Virtual execute(CommunicationCenter)
Mission1
+execute()
Mission2
+execute()
Mission3
+execute()


Verify that there is a connection to the drone
Put Drone in Command Mode
Send information
Receive information
Drone
Communication Center