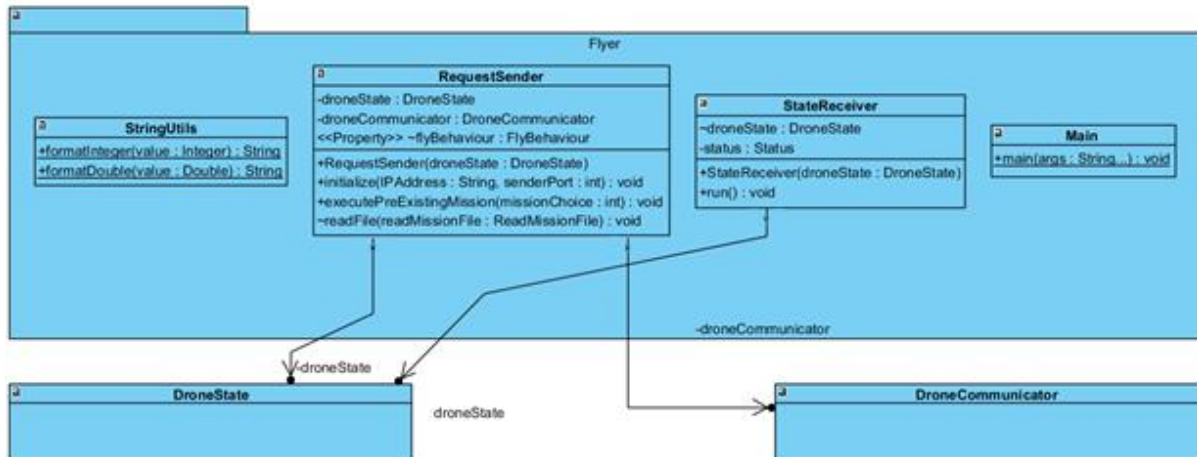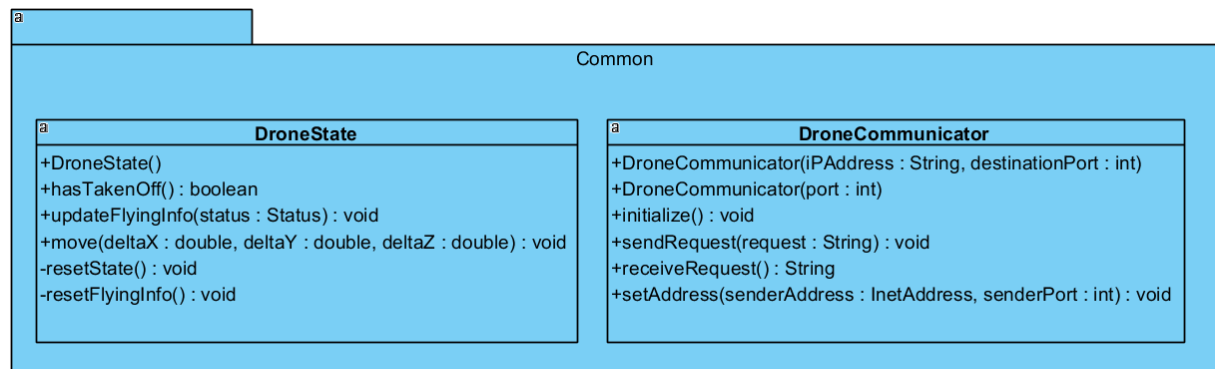# HW2 Report
Submitted on: 04 October 2019
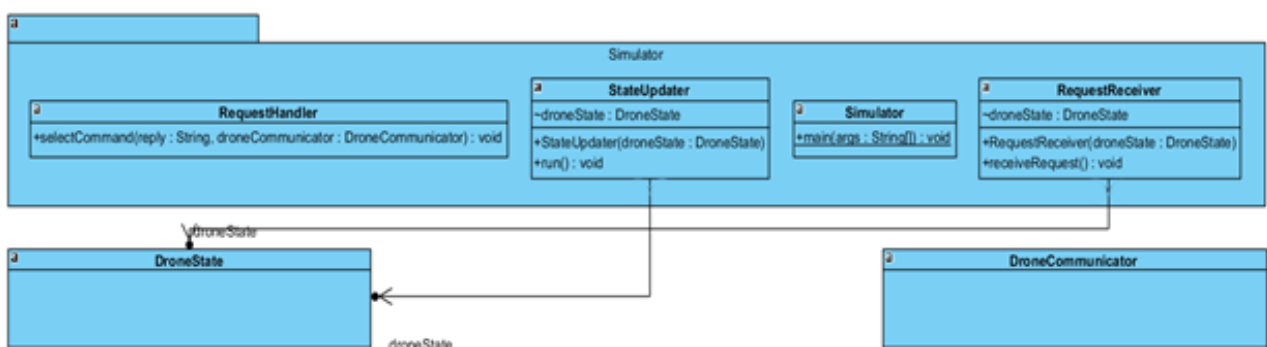
UML Diagrams
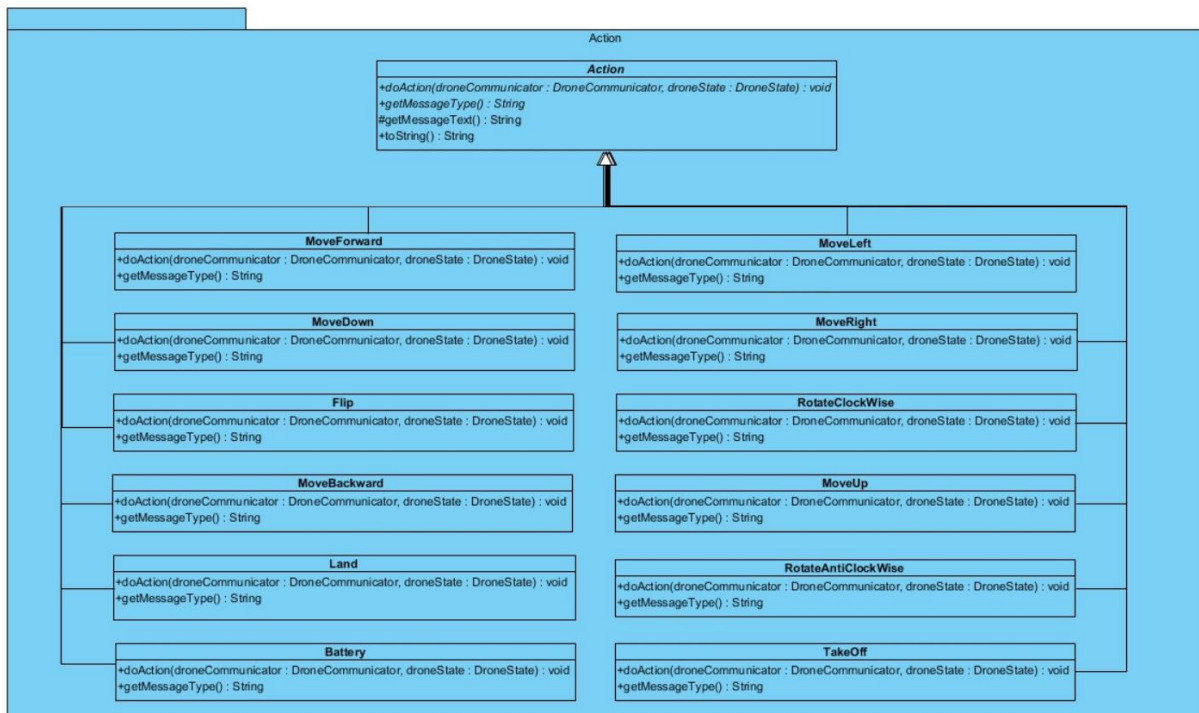
1. Class diagram of package Flyer:

2. Class diagram of package Common:

3. Class diagram of package Simulator:

4. Class diagram of package Action:

Action

Action
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String
#getMessageText() : String
+toString() : String

MoveForward
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

MoveLeft
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

MoveDown
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

MoveRight
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

Flip
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

RotateClockWise
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

MoveBackward
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

MoveUp
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

Land
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

RotateAntiClockWise
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

Battery
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

TakeOff
+doAction(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+getMessageType() : String

5. Class diagram of package ReadMission:

ReadMissionFile

ReadMissionFile
~status : Status
+readFile(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
+executeMission(droneCommunicator : DroneCommunicator, result : String[], droneState : DroneState) : Action[]

ReadXML
+readFile(droneCommunicator : DroneCommunicator, droneState : DroneState) : void

ReadCSV
+readFile(droneCommunicator : DroneCommunicator, droneState : DroneState) : void
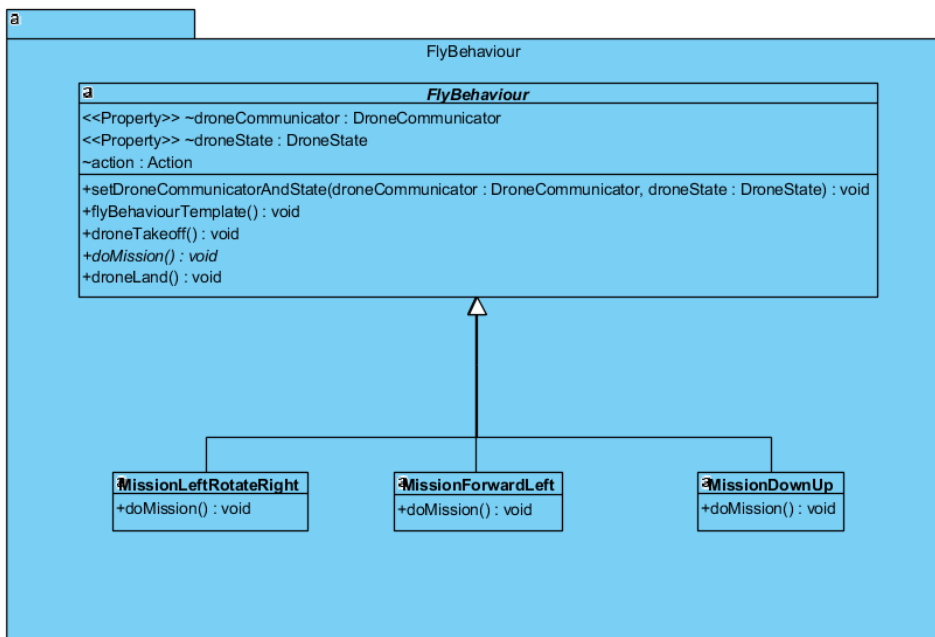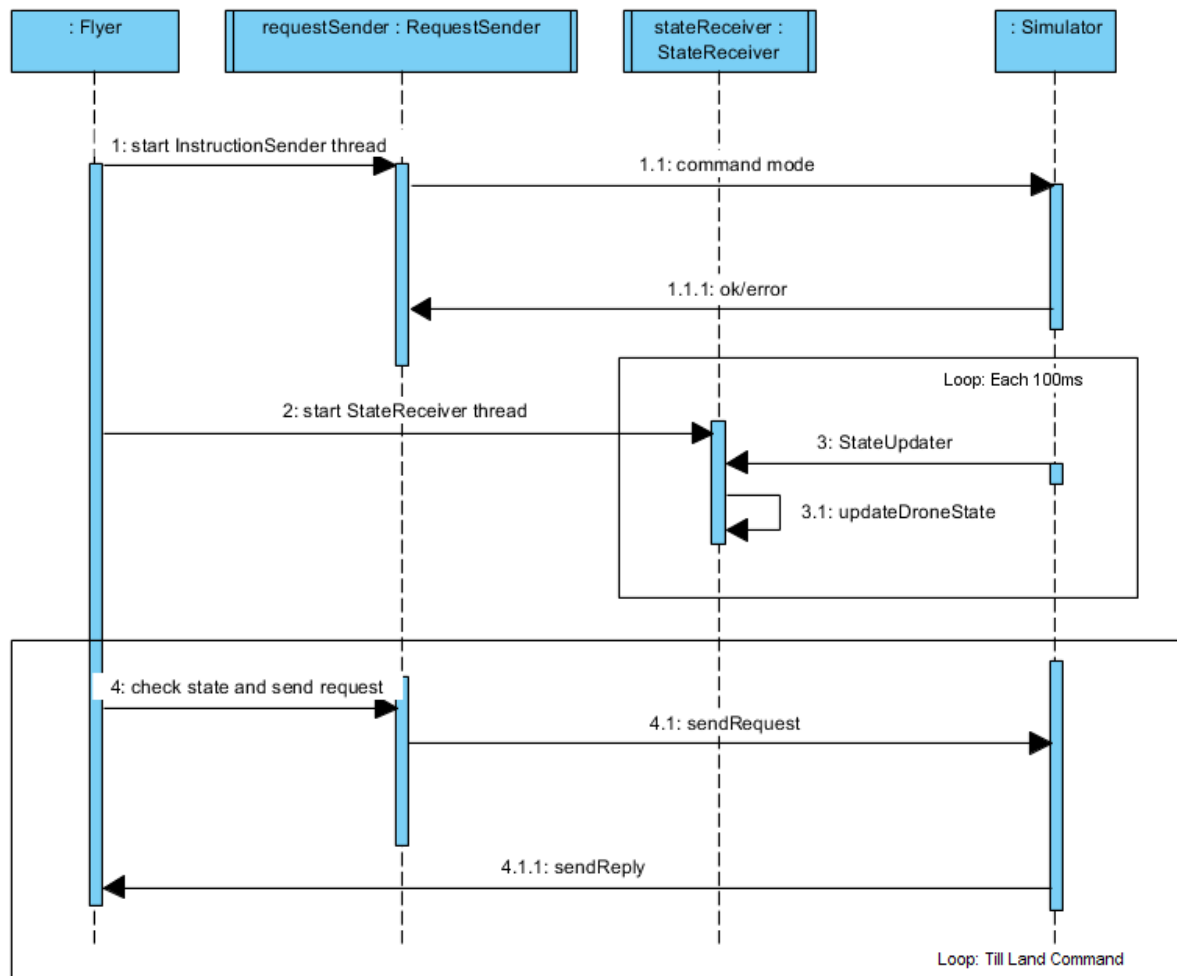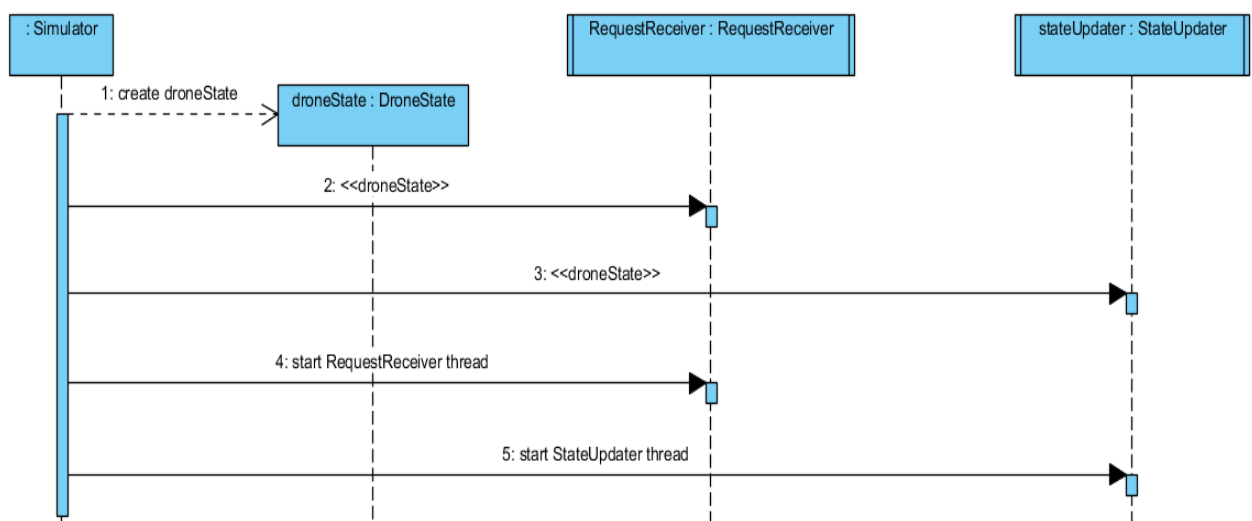
6. Class diagram of package Message:

7. Class diagram of package FlyBehaviour:

8. Interaction Diagram of Flyer:

9. Interaction diagram of Simulator:

Insights uncovered during the project

This project helped me a lot in understanding importance of modularity, abstraction and encapsulation. I learned many aspects including importance of testing and how to code in such a way that it is easily maintainable and testable.

I also learned how to pass objects from one class to another while maintaining the abstraction and proper design and flow of the project.

More importantly, I learned how to apply abstraction in the project. The difference between public and private and also the significance of declaring the data members in different places was something new. I never understood the importance of abstraction before this assignment since I had no practical exposure so that is something very important I learned from this project. I also learned how to establish communication between 2 classes using socket programming. This project also helped me learning more about datagram socket and packets.

Talking about the factory pattern, it was fun to study and implement. It can be applied to some places only but it gives better abstraction since we can create objects at runtime.

One more thing worth sharing is the UML diagram itself. I didn't know anything about how to develop class diagrams and so this project gave me motivation to learn that as well. Also, just like everything else, I didn't know how to do unit testing as well. Although that's the only thing which is incomplete in this submission but considering the fact that there was too much to learn during this assignment, I am very satisfied that I made such efforts and learned so many things. I will definitely complete the unit testing in upcoming submissions.

After creating the interaction diagrams, the flow was clearer to me and it was easy to understand what is going on in the project.

Overall, after starting this project and applying design pattern in this assignment, I realized the value of modularity and abstraction and learned many new things in the process as well.