# SE322 - Software Architecture
# Project - Phase 1

## 2021-22 Spring

SUMAY
18244710015-Aysena YEŞİLÖZ
17244710021-Sümeyye Ayşe ÖZDEMİR

22.03.2022

## 1.1 Project name and description

- Is the system solely a software, hardware or contains both?
  This code contains both of them.
- What is the system modelled?
  This system is drone system.
- What is the purpose of the system?
  Drones are unmanned aerial vehicles that can be controlled, commanded, and video and image recordings can be made with a remote control.

# 1.2 List of Components

## 1.2.1 Execution Platform Components

Device, processor, bus, and memory.

Radio: transmits information over a long distance.
Wifi: It is a device that wirelessly connects devices working with the drone.
Camera: The drone camera is used to record images in many areas.
UART: UART is a device that allows applications to communicate in asynchronous serial mode.
GS_Bus: used to transmit signal data on the station.
GS_Proc: GS Proc is used to recieve signals from grand station and to transmit signals.
GS_Mem: It is the memory area used in the ground station to transmit signals.
GPS_Receiver: It receives signals from satellites and interprets these signals to indicate location
MC_Mem: is the space in ram that the task computer allocates for its work.
MC_Proc: a device for interpreting and processing computer commands for tasks.
MC_Bus: it is the system that provide the data that the task system will carry out about the transitions and controls.
Wifi_Bus: It is used for data reception and transmission of the wireless internet used in the drone to other devices.
Serial_Bus: completes the data transmission with the help of bits.
RF_Bus: Allows the transmission of radio frequencies data.

### 1.2.2 Application system components

system, process, thread, thread group, subprogram, and data.

ASE_MsgHeader: This is where messages are collected in headers.
CASE_RF_Msg: It is the place where data is collected by radio signal.
CASE_WIFI_Msg: It is the part where the messages obtained through wifi calling are brought together.
CASE_UART_Msg: It is the place where asynchronous data in UART structure is collected.
FlightController: The system that controls the flight movements of the drone system.
GS_SW: This is a communication software. It enables the system to be used with radio signals while at the ground station.
GroundStation: The system forms the basis of communication by receiving and sending back radio frequency signals.
MissionComputer: This system recognizes all missions sent to the drone and returns its response according to the task.
Coordinate: The drones are positioned according to the coordinates and are taken as data.
Map:It is a data structure that stores a set of coordinates that a geographic area.
MapArray: This map array would carry numerous maps, but in this version, it carries an array of five maps.
FlightPattern: It controls the flight movements of the drone. For example, it regulates the take-off and landing movements.
Command: These are the instructions that govern all of the drone's capabilities.
RF_Msg: It is the message that must be received by the drone via radio waves in order for it to be used.
Mission: It refers to all of the tasks that the drone must complete in accordance with the specified instructions.
MissionWindow: The drone's route is defined using the Mission window.
WifiDriver: It's the software that allows the drone to connect to the internet and function.
RadioDriver: This software that allows the drone to pick up radio signals.
FlightPlanner: The route that the drone should take is planned.
NoFlyZoneDatabase: The drone is nearing forbidden or private zones, according to a database.
WaypointManager: To create waypoints and routes on a computer, Waypoint Manager software is needed.

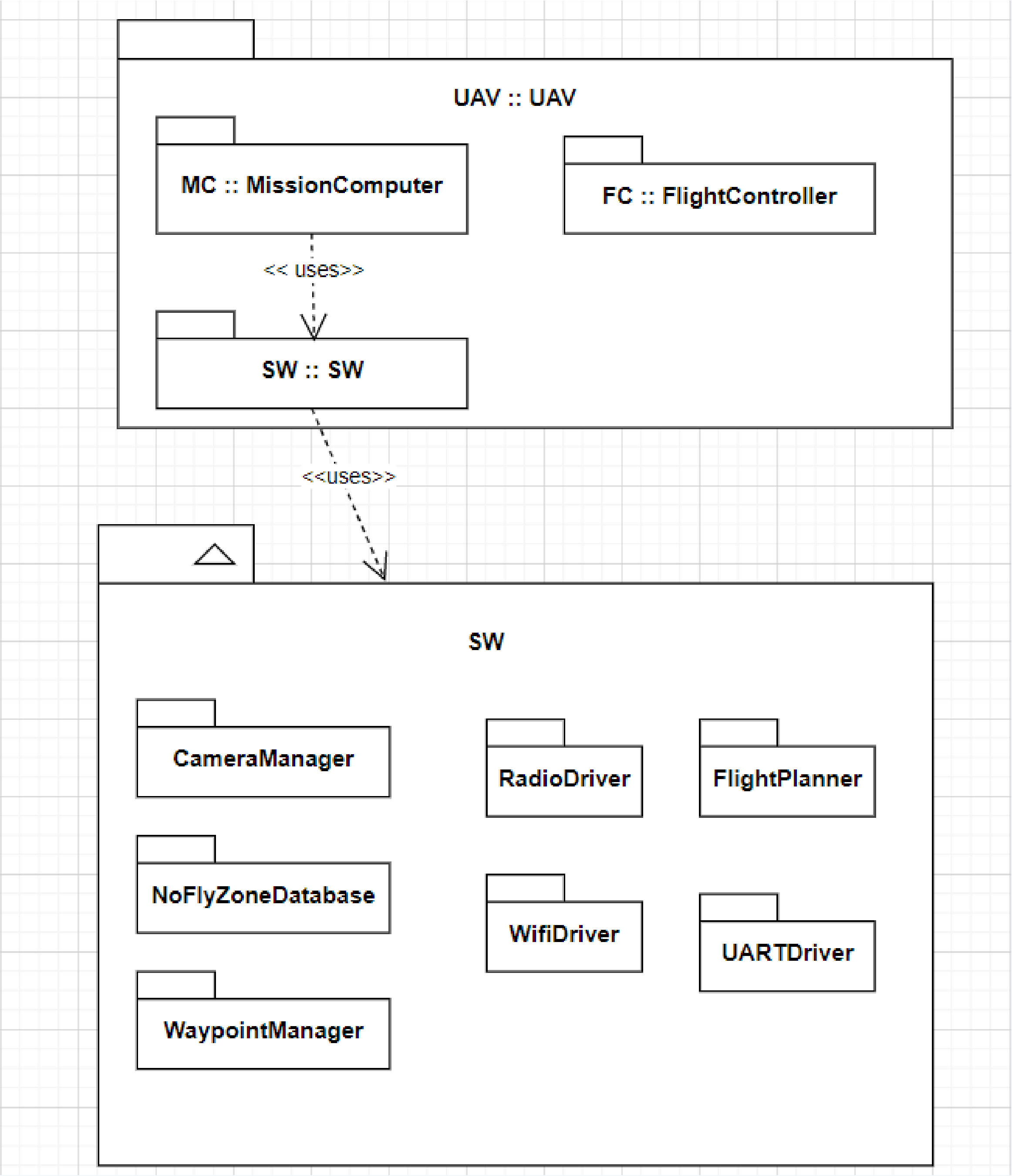
CameraManager: The camera on the drone is controlled and managed by this system.
UARTDriver:
SW: It refers to the drone's software.
SWS: All of the software for a drone is housed in a system.
UAS: These are all systems necessary to operate the drone system, for example camera, gps, radio, wifi.UAS is an unmanned aerial system.
UAV: UAV is an unmanned aerial vehicle. It is aircraft that flies about collecting vital data and images.

### 1.3 List of data exchanged by components

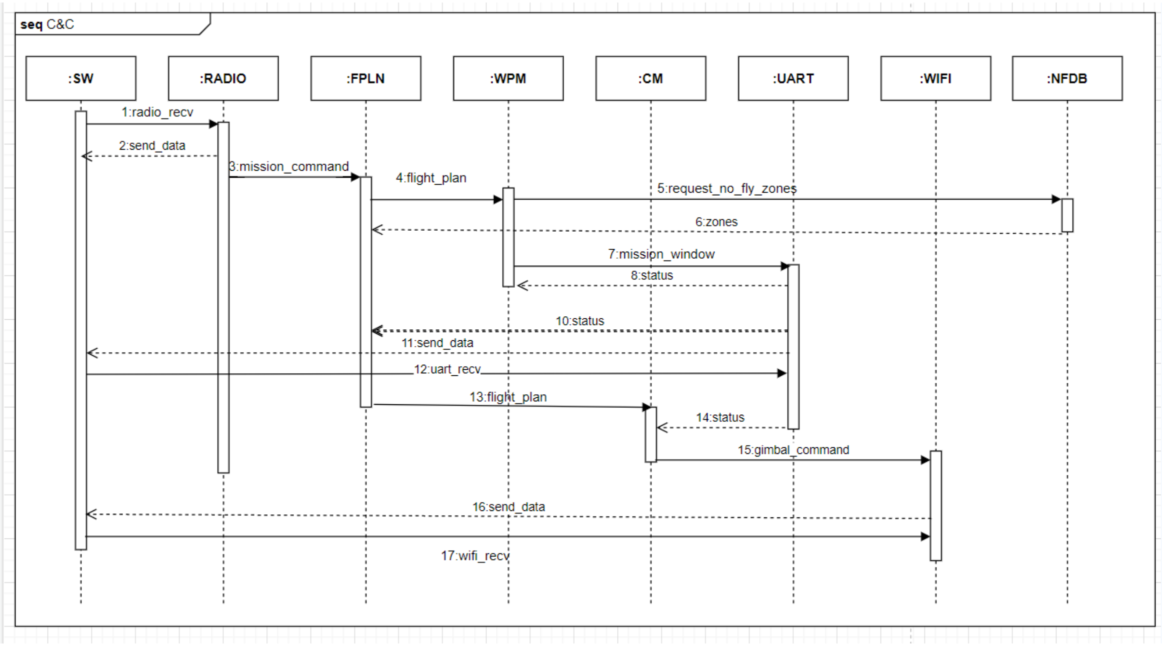
- The CASE_Msg Header data is shared in the data implementation definition of CASE_RF_MSG. The structure of this data has four subcomponents. in src and dst integer types, trusted and HMAC are Boolean types.
- CASE_RF_Msg has two subcomponents, header and payload. The CASE_RF_Msg data is the Radio in the MC and the Radio Driver in the SW is a thread, the subcomponent of the RF_Msg is the payload, and the data is being refined from the Command.
- CASE_WIFI_Msg has two subcomponents, crc (Boolean type) and message (String type). CASE_WIFI_Msg is associated with the Wifi device in MC and the WifiDriver thread in SW.
- CASE_UART_Msg has wiki sub-components, namely crc (Boolean type) and message (String type). CASE_UART_Msg is associated with the UART device in MC and the UARTDriver thread in SW.
- Coordinates have three subcomponents of integer type, latitude, longitude, and altitude.This is the structure to hold the lat/long/alt values of a coordinate. In addition, Coordinate data is shared with CameraManager, Flight Planner, Radio Driver, UARTDriver, and WaypointManager threads.
- Map data is shared in the FlightPlanner and NoFlyZoneDatabase threads and with the MapArray data. The map is a structure that contains a list of coordinates surrounding a region, and we fix the size of the map to 4 waypoints.
- MapArray shares its data with the FlightPlanner and NoFlyZoneDatabase threads. The MapArray is a structure that hold multiple Maps and we fix the size of the map array to 5 maps.
- The Flight Model is a numbering (ZigZag, StraightLine and Perimeter) that defines how the UAV will fly out of the detection zone to conduct surveillance.
- The Command structure contains data that the Ground Station passes to the UAV. The Command data is associated with RF_Msg, and its subcomponents are map from MAP and pattern from FlightPattern.
- The RF_Msg data is shared with the FlightPatten and RadioDriver threads.
- The Mission data is shared with Camera Manager, Flight Pattern and Waypoint Manager threads.The Mission is a list of waypoints in a 10 array.
- The MissionWindow data is shared with the UARTDriver and WaypointManager threads.It is a list of waypoints that the Waypoint Manager assembles from the Mission.

## 1.4 Views

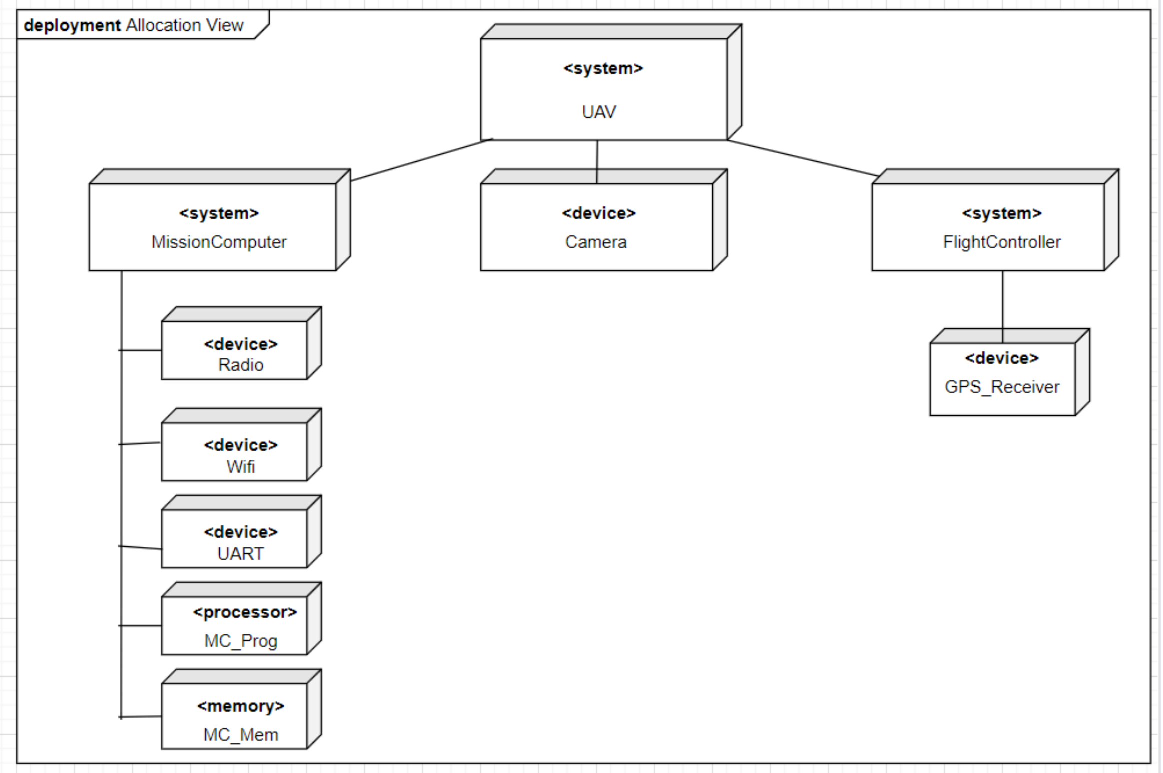
### 1.4.1 Module View



## 1.4.2 C&C Module

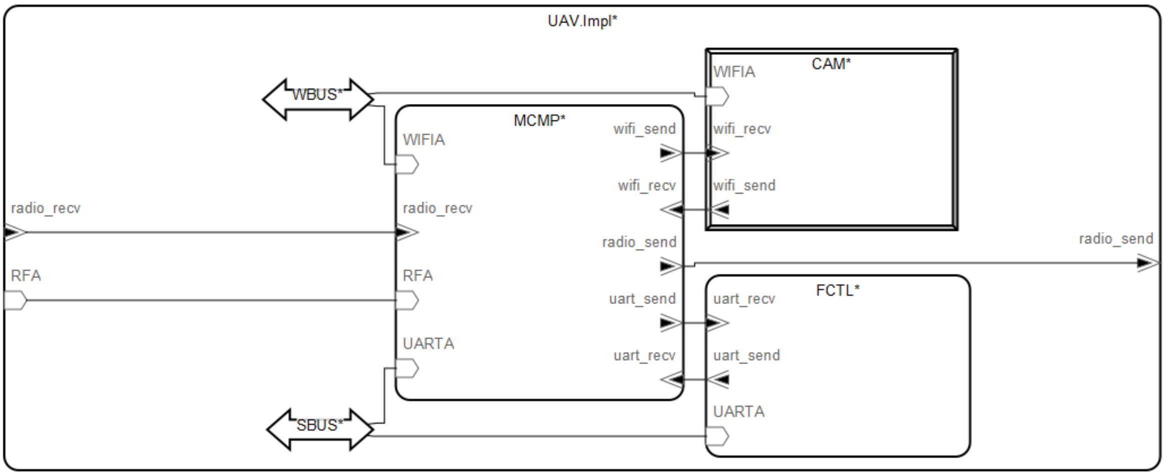


## 1.4.3 Allocation Module

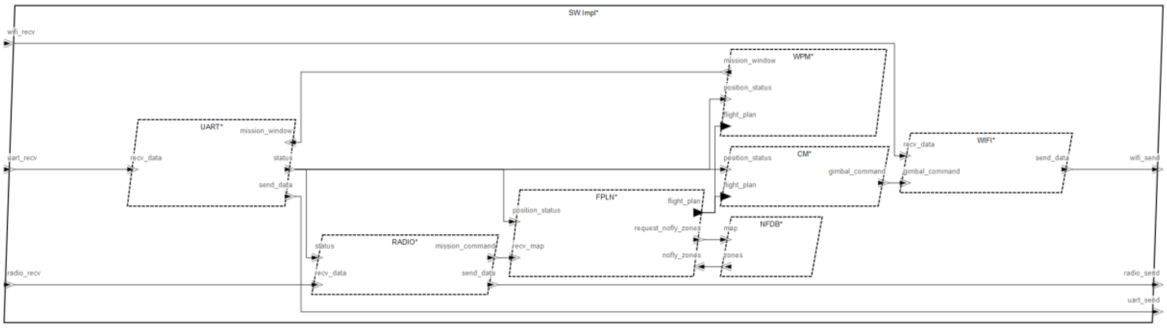


## 1.5 Architectural Diagrams by OSATE

(i) system UAV (refer to UAV.aadl)



(ii) system SW (refer to SW.aadl)



# 2 Github Repo

**https://github.com/SumeyyeAyse/SE322---SUMAY.git**

# 3 Discussion

## 3.1 Usability of the tool

To discuss the tool, our initial thought was neither positive nor negative. It was easy to understand the codes, understand what where and what functions, but at some points we had difficulties, like diagrams. For diagrams, videos, web resources like PDFs and weekly uploads to Moodle made it a little easier to understand and learn.

## 3.2 Learnability of AADL language

AADL is a language that not everyone knows. It deals with the collaboration of hardware and software. Although there are resources available to us, unfortunately they are not enough. The structures of the language are easy to understand and apply. As Software Engineer students, we think that if we can learn the subject in detail, it will be really useful for us in some areas of business life.

## 3.3 Complexity of the given system

-is the given system architecture complex: is it hard to understand the whole system?

Yes, the system architecture of the data was difficult and complex. The whole system was not difficult to understand. However, it was difficult to understand because abstract and concrete data were used together.

- is the complexity originated from the system (such as the system is already has a lot of elements, complexity comes from the domain) or the way it is modelled (the designer made it complex)?

  The complexity the system was generated from the way the elements were interacting with each other as it was the first time for us using this type of language however the designer tried to make it as simple as possible by adding little details to help us understand such as comments

- is there a convention followed for interfaces (naming, defining features of the software elements or connections)?

Yes, there is. But for this language, which we started learning for the first time, it was difficult to understand the interaction of systems and data with each other.