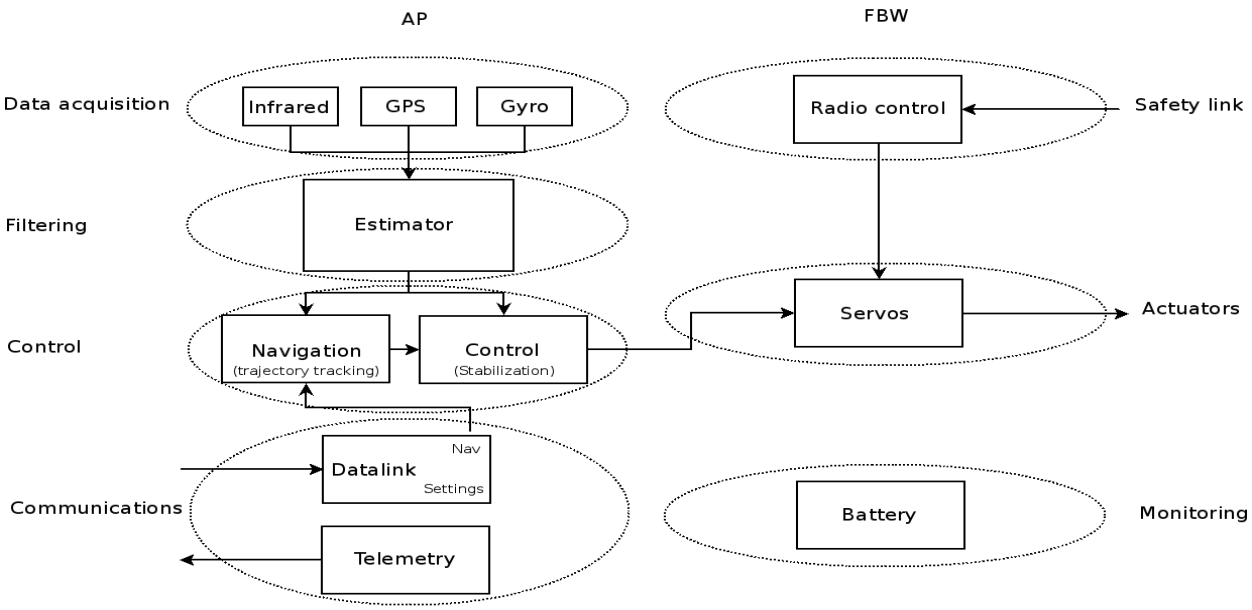

Figure1: Functional Diagram

The above figure shows the complete functional diagram of interaction between the various modules in the “Paparazzi” system. In this document, I will be explaining about the organization of the current code and what part in current code will require changes so that we can port this system to a “RTOS”.

The code of the complete system has been divided in two parts, first part is firmware which is deployed on the actual hardware (means the Air Vehicle) and the second part includes “Ground Control Station” and “server” which runs on the laptop and used for controlling the vehicle. After studying and looking the code I felt that the code which runs on the ground will require no changes and major changes need to done in the firmware.

Paparazzi way of modularity:

As the above functional diagram shows there are many hardware units in the hardware and each unit can be of different capabilities(for example there might two boards with different GPS unit) and can come from the different vendors. Therefore it was necessary to keep the code clean and modular so that it can be extended very easily. But unfortunately the code was written in “C” and “C” doesn’t provide automatic facility for modularity. “Paparazzi” achieves this modularity and the hardware abstraction by the means of configuration files and conventions of modules and subsystems. Whenever a flight is prepared for flying, all the units necessary in firmware are configured through .xml files and only those parts are compiled which was configured in the configuration files.

Besides this below are two conventions followed by “Paparazzi”:

- Subsystems : Subsystems provides several implementation for a hardware unit like(GPS,MCU or Sensors) or an algorithm like(for navigation , control loop).
- Modules: Modules are similar to subsystems but the functions for a module are added through xml files without explicitly change in the code. The functions in the modules are basically events and periodic functions.

Currently the system runs in a loop which is called the main loop of the system. In this main loop various functions are called periodically with some configured time and many events are handled for different hardware units.

As every task in the “Paparazzi” is being scheduled with a definite period of time and frequency and with it's own scheduling algorithms, therefore major changes needs to be done in this part so that we can make “Paparazzi” to use the scheduling algorithm provided by “RTOS”. The below diagrams shows a sample event loop in current code:

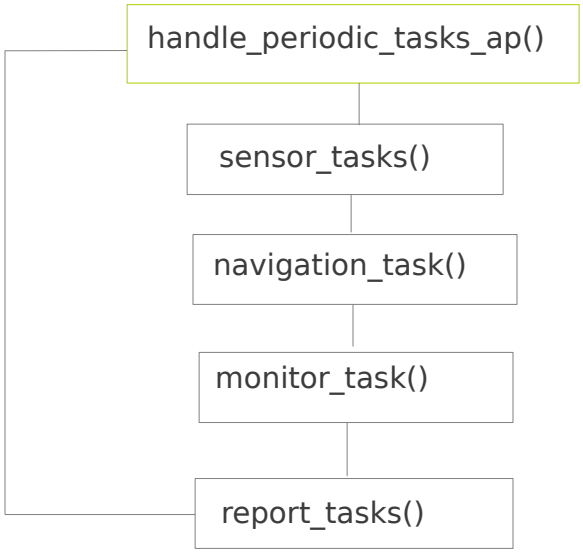

Figure 2: Sample MainLoop in Paparazzi

Besides the scheduling part, we also need to handle the events generated by the hardware units.