Task 4.2 – Survey & Rescue

Problem Statement

1. Aim of this task is to make a *scheduler* script that will command the drone to serve a particular event during the run.
2. The script will subscribe to the messages published by *beacon detector* script and will take decision when to serve which event during the run, according to the servicing rules section of Rulebook.

About the Scheduler script

1. The script should start a ROS node named *scheduler* and should subscribe and publish to the following topics
   For more information refer Figure 1 and Figure 2

   Subscribes to:
   i. /detection_info:
      Message type: SRInfo
   ii. /serviced_info:
      Message type: SRInfo

   Publishes to:
   i. /decision_info:
      Message type: SRInfo

   Note: Apart from these topics, you can publish/subscribe to any other custom or standard theme topics from other nodes if you wish but the above subscription and publications are must for the scheduler script.

Survey & Rescue Flowchart

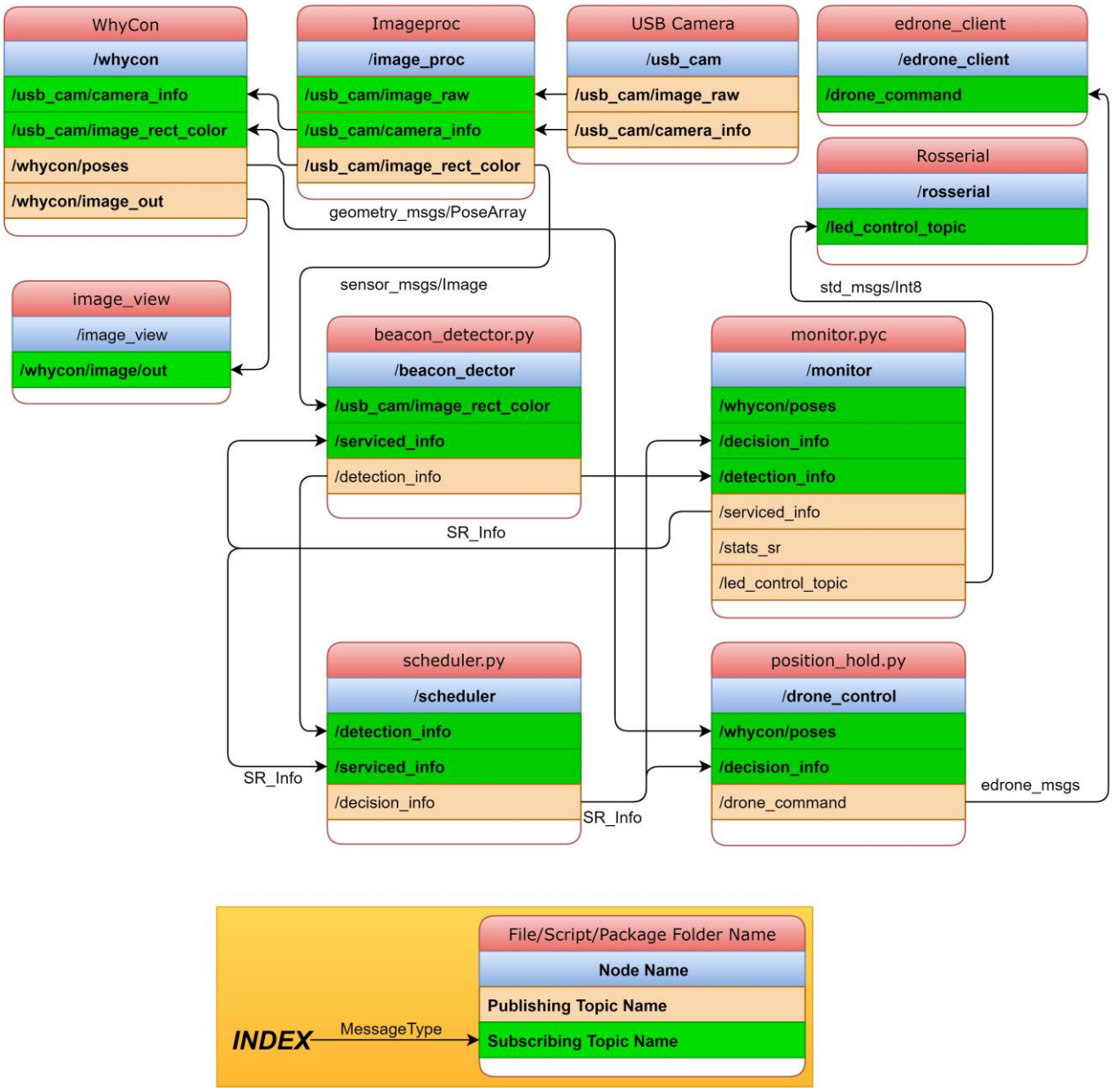
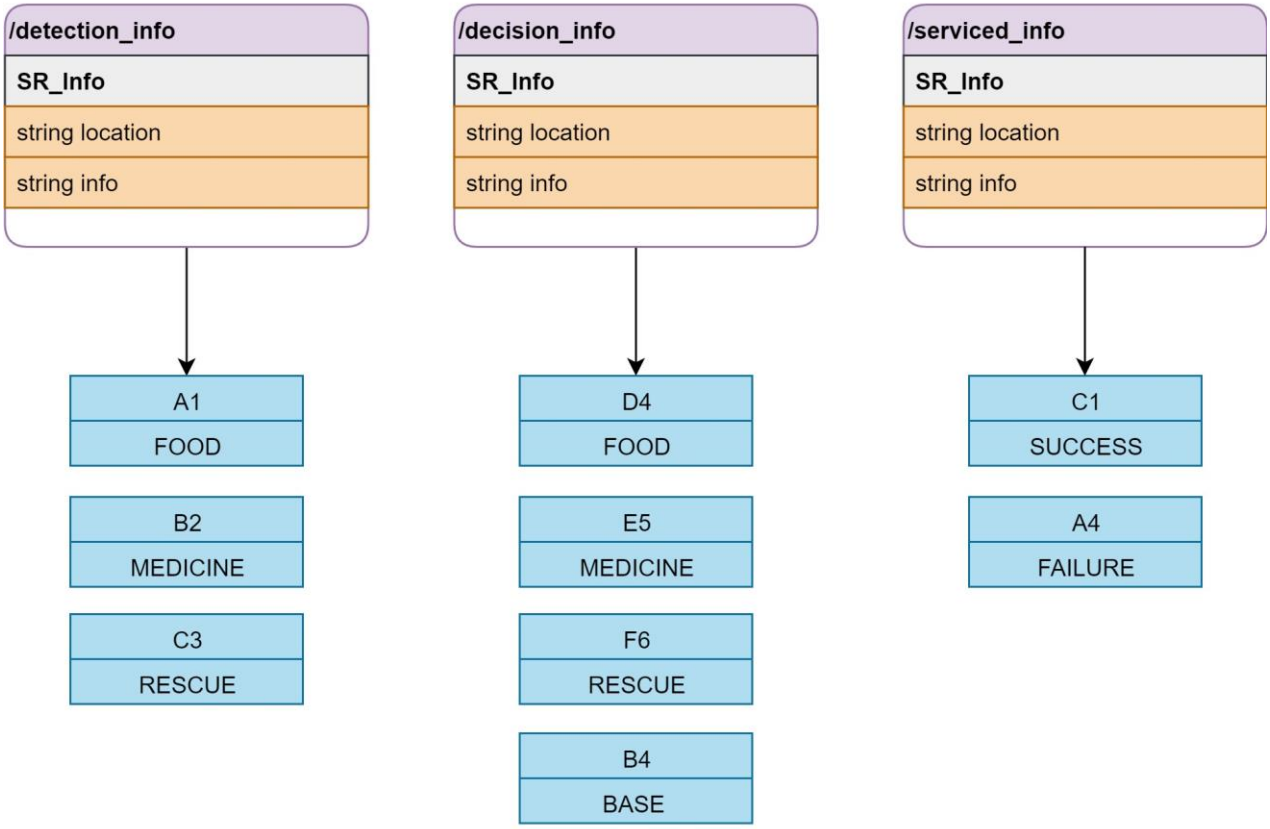


Figure 1: ROS nodes description

Survey & Rescue Flowchart



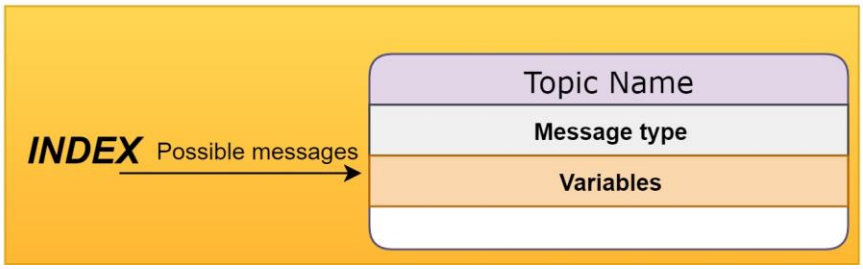


Figure 2. Topics description

2. The *scheduler* gets information about the type of Beacon and location of Beacon from the *beacon detector* node.
3. The *scheduler* publishes its decision on the topic */decision_info*. The decision consists of two parameters i.e. **Location** and **Info.** Examples of decisions taken by the *scheduler* can be

Location: "A1"	Location: "F4"	Location: "C3"
Info: "MEDICINE"	Info: "BASE"	Info: "RESCUE"

4. The messages that are published on the topic */decision_info* should be subscribed by the position hold script. Unlike in Task 3.2 where you had to visit all the cells. Now the position hold script should dynamically change its setpoint according to the location information obtained from the subscribed topic and should hover at that co-ordinate.
5. The drone should hover over a cell for the time depending on the type of event described in the Rulebook.
6. The hover time is monitored by the *monitor.pyc* node that runs in parallel with the *scheduler*, the command of a successful service or a failed service is sent on the topic */serviced_info* by the *monitor.pyc* and the *scheduler* can use the messages on this topic to schedule further tasks. That means the *scheduler* need not run a separate timer to check the hover time of the drone. Refer the github page of *monitor.pyc* for more information.

Tip: After the completion of *scheduler* script along with all previous tasks, you can start to practice the final theme implementation by creating your own *LED_Config.tsv* and *LED_Timing.tsv* files, but in the final Task 5, you will be given the configuration.

Note: This will be the last individual task that you will be performing, the next task will be Task 5 in which you will be given a final configuration for the complete theme implementation and you have to submit the complete run. Like Task 3.3, you are not required to submit task 4.2 but you still need to complete scheduler along with all the previous tasks before the release of Task 5 and you should be ready with all the scripts required to complete the theme.