MRT Assignment 2
ROS 2

Anushka

December 2023

Problem Statement

Create a ROS 2 workspace named `mrt_ws` and the package named `your_name_assn_2` to incorporate three Daughter Rovers, each contributing distinct information to the ROS ecosystem.

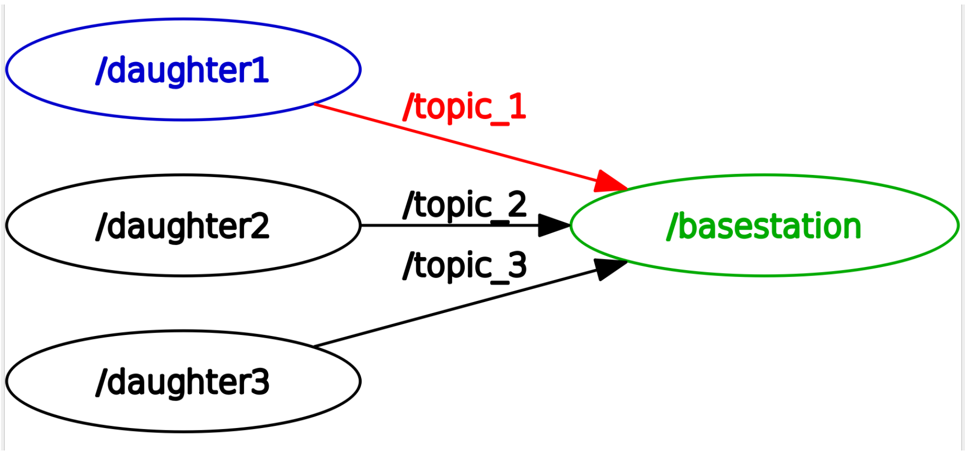

Figure 1: rqt graph

Daughter Rover 1 (`d_rover1`):

- Develop a ROS 2 node named `d_rover1` within the package.
- The `d_rover1` node must include a publisher that communicates with the ROS topic `topic1`.
- Utilize messages of type `std_msgs/Float32` to publish a random float value between 0 and 100 as altitude.

Daughter Rover 2 (`d_rover2`):

- Introduce a new ROS 2 node named `d_rover2` within the same package.
- The `d_rover2` node should publish three random float values between 0 and 200 as its current location to the ROS topic `topic2`.
- Employ messages of type `geometry_msgs/Point` to transmit location data.

Daughter Rover 3 (`d_rover3`):

- Integrate a third ROS 2 node named `d_rover3` into the package.
- The `d_rover3` node is responsible for generating a random integer between 1 and 10.

- If the generated integer is greater than or equal to 5, publish the string ”Task accomplished” to the ROS topic `topic3`.
- Otherwise, publish the string ”Mission Failed” to `topic3`.

Base Station Node (`basestation`):

- Create a ROS 2 node named `basestation` within the same package.
- Implement three subscribers in the `basestation` node:
  - Subscribe to `topic1` using a message type of `std_msgs/Float32`. Display the received altitude values.
  - Subscribe to `topic2` using a message type of `geometry_msgs/Point`. Display the received location data.
  - Subscribe to `topic3` using a message type of `std_msgs/String`. Display the received mission status string.

Launch file

Instead of initiating individual nodes one by one through separate terminal windows, employing a launch file streamlines the process. This file facilitates the efficient management and simultaneous initiation of all nodes. Therefore, generate a launch file for the simultaneous launch of these nodes.

rtq_graph

After making the above nodes, run them together and create an rqt graph to visually analyse the connections between nodes. Summarize your learning in a short report.

Bonus Task: Daughter Rover 4 (`d_rover4`):

- Implement a new ROS 2 node named `d_rover4` within the package.
- The `d_rover4` node should include a publisher that communicates with the ROS topic `topic4`.
- Define a custom message with the following fields:
  - `int32 rover_id`
  - `float32 battery_level`
  - `geometry_msgs/Pose current_location`
  - `string health_status`
- Publish messages of the custom type to `topic4`.
- Take random values for rover_id, battery_level, current_location and health_status.
- Add a subscriber to `topic4` in the basestation node created above.

Extras

To maintain repository of solutions to the MRT assignments, git is a handy tool. Create a remote git repository and push your codes and report into the remote repository.