MRT Electronics and Communications - Task 1

Shridhar Patil

October 2024

0.1 Introduction

This introductory task went through topics such as:

- Linux and terminal commands.
- Meaning of nodes and topics, their creation and their interconnection.
- Structure of projects and packages in the directory.
- Usage of github and git commands through terminal.
- Usage and creation of reports in LaTeX.

0.1.1 Requirements:

- ROS2 installation (install/setup.bash file sourced).
- colcon for building packages.
- git installation.

0.1.2 Result:

Two nodes Publisher1 and Publisher2 are created which publish their message to two separate topics *Listen_1* and *Listen_2*. A listener node is subscribed to both *Listen_1* and *Listen_2* and displays the messages gathered from both.

0.2 Procedure:

0.2.1 Directory creation:

A workspace directory is to be created with name **mrt_ws**. This is done with **mkdir mrt_ws** command in terminal. Inside this directory, we create a directory called **src** which houses all the package files. We create a package with help of ROS2 inside **src** as:

ros2 pkg create –build-type ament_python sauron. (1)

This will create a directory called **sauron** and some premade files will be present inside it.

opening the terminal in **mrt_ws** directory and running **colcon build** will build the packages and create a **install** file with textbfsetup.bash file which should be sourced in the terminal to run commands in the package.

source mrt_ws/install/setup.bash (2)

0.2.2 Adding Dependencies:

A **packages.xml** file is present in **sauron** file. Some external packages are required to be used by **sauron** package. These dependencies are added by writing:

<exec_depend>rclpy</exec_depend>
<exec_depend>std_msgs</exec_depend>
In the **packages.xml** file.

0.2.3 Node Creation:

All node types and regarding data is present in the **mrt_ws/src/sauron/sauron** file. Here 3 files are created:

- publisher2
- publisher1
- listener

The Code for each of the nodes is as given in the repository. Both publishers create different topics to pass messages. Given node listener is subscribed to both these topics and hence it gets messages from both topics and displays them together.

0.2.4 Modification of setup.py:

The **setup.py** file is located in **mrt_ws/src/sauron** and it contains information on how the package is structured and built. This **setup.py** should be modified to define what code to run when publisher1, publisher2 and listener commands are executed. (This is present under the *console_scripts* heading.)

The **setup.py** file also requires to *import glob from glob* and *import os.*

0.2.5 Launcher file creation:

Inside the textbfmrt_ws/src/sauron/launch file, create a file called **sauron.launch.py** file. This file when run can create all nodes and topics simultaniously and setup the entire connection in one terminal command.

The **sauron.launch.py** file references the package name and command name (as defined in **setup.py** for the code in nodes located in **sauron/sauron** directory).

0.2.6 Building and execution:

The entire package can be built by using **colcon build** command inside directory **mrt_ws**. A new terminal is opened and **setup.bash** file is sourced with:

$$\textbf{source mrt\_ws/install/setup.bash} \tag{3}$$

This can be added to the *.bashrc* file to automatically execute on each new terminal. The command:

$$\textbf{ros2 launch sauron.launch.py} \tag{4}$$

Is to be run in **mrt_ws/src/sauron/launch** location through terminal. This will create all nodes as defined in launcher script.

0.2.7 Verification:

Running **rqt_graph** will show a graph of all nodes and topics present currently (A refresh may be required). **ros2 run sauron listener** will execute the **listener.py** file in the **sauron/sauron** directory (since this command is named in the **setup.py** file of package).

Text as given by both topics will be displayed together since listener node is subscribed to both nodes.