EN4594 Autonomous Systems
Laboratory Sheet
Pre-Lab Session

Title: Writing a Simple Publisher and Subscriber (C++)

1. Background

Node is the ROS term for an executable process that communicates over the ROS network. First, we will create a publisher ("talker") node which will continually broadcast messages. Next, we will create a subscriber ("listener") node that will listen to the broadcasted messages by the publisher.

2. Steps

Here, we describe the steps involved in writing a simple publisher and subscriber in C++.

2.1. Create a Package

Packages should be created in the `src` directory, not the root of the workspace.

- Open a new terminal and navigate to the `src` directory of your ROS workspace.
  e.g. `cd ~/ros2_ws/src`
- Let's create a package named `cpp_pubsub` using the following command:

```
ros2 pkg create cpp_pubsub --build-type ament_cmake
```

The terminal will return a message verifying the creation of your package and all its necessary files and folders.

- Navigate to the newly created package folder (e.g. `ros2_ws/src/cpp_pubsub`) using a file explorer and copy the given files to the corresponding folders (`src`, `include/cpp_pubsub`). You may replace `CMakeLists.txt` and `package.xml` files.

2.2. Examining the Files

We can use Visual Studio (VS) Code to examine the files.

- Open a new terminal and navigate to your ROS workspace.
  e.g. `cd ~/ros2_ws`
- Type the following command to open VS Code inside the ROS workspace:
  `code .`
- You will observe the file structure given in Fig. 1, where certain files are just duplicates of others but contain comments to understand the code (e.g. `my_publisher_comments.cpp`).

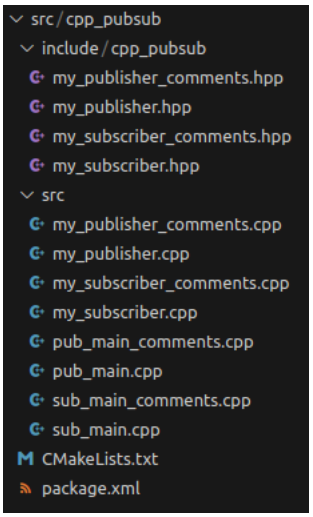

Fig. 1: File structure observed in VS Code

We have three files associated with a node (executable). For example, for the *talker* node we have

- my_publisher.hpp (header file)
- my_publisher.cpp (c++ file)
- pub_main.cpp (main file)

Go through the associated comments files to understand the code. For the *talker* node, we have

- my_publisher_comments.hpp (header file)
- my_publisher_comments.cpp (c++ file)
- pub_main_comments.cpp (main file)

Now open the `CMakeLists`.txt file. You can observe that,

- the dependencies are met through the `find_package(...)` lines.
- executables are created using `add_executable(...)` lines, where we have given the names *talker* and *listener* so that you can run these nodes using `ros2 run` command later.
- `install(TARGETS...)` tells you where to find the executables.

Next, open the `package.xml` file, which contains meta information about the package. The `<depend>...</depend>` tags list the dependencies on other packages, for `colcon` to search for.

2.3. Build and Run the Nodes

You likely already have the `rclcpp` and `std_msgs` packages installed as part of your ROS system. It's good practice to run `rosdep` in the root of your workspace (e.g. `ros2_ws`) to check for missing dependencies before building. Open a new terminal and type the following:

```
rosdep install -i --from-path src --rosdistro humble -y
```

Still in the root of your workspace, build your new package:
```
colcon build
```

To run the *talker* node, open a new terminal and type:
```
ros2 run cpp_pubsub talker
```

To run the *listener* node, open another terminal and type:
```
ros2 run cpp_pubsub listener
```

Enter `ctrl+c` to terminate the nodes.