Robotics Operating System (ROS)

Basic principles, use cases, and examples

Overview

- Introduction to ROS
  - What is ROS
  - Why ROS
  - ROS graph
- Developing with ROS
  - Workspace
  - Packages
  - rclcpp & rclpy
  - Utilities
- Example
- ROS community

Naïve Robot Software - Example 1

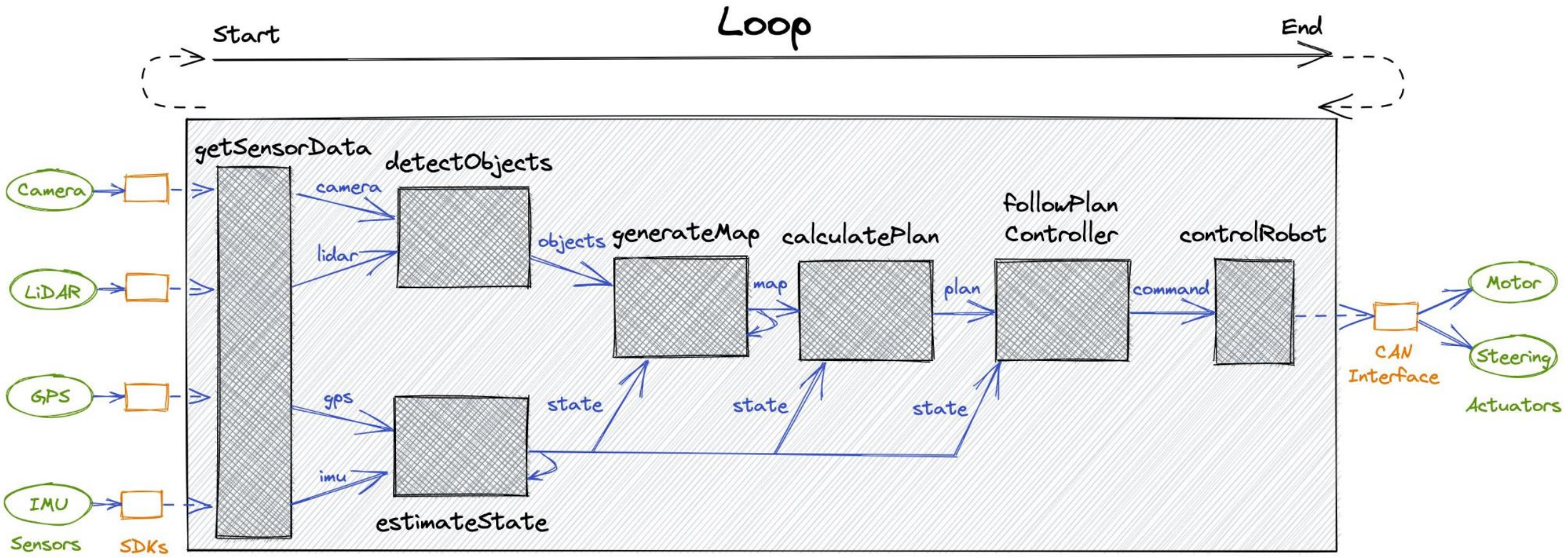
```
while (isRunning()) {
  camera, lidar, gps, imu = getSensorData()
  objects = detectObjects(camera, lidar)
  state = updateState(state, gps, imu)
  map = updateMap(map, objects, state)
  path = calculatePlan(map, state)
  command = followPathController(path, state)
  controlRobot(command)
}
```

Naïve Robot Software - The problem

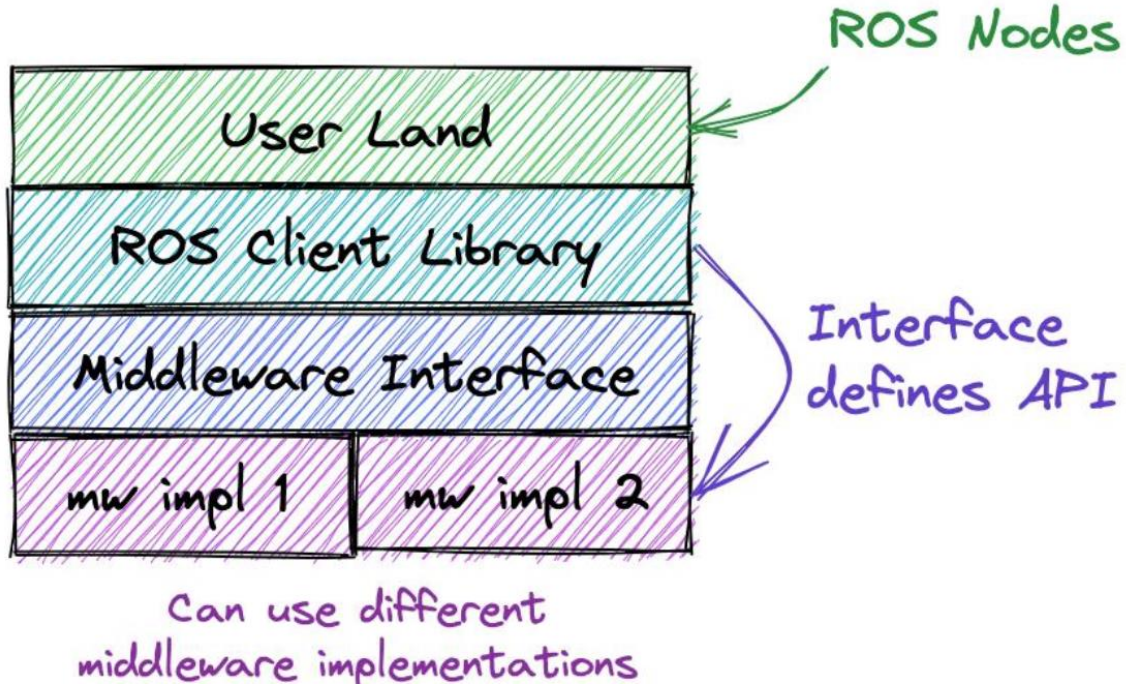
```
while (isRunning()) {
  camera, lidar, gps, imu = getSensorData()
  objects = detectObjects(camera, lidar)
  state = updateState(state, gps, imu)
  map = updateMap(map, objects, state)
  path = calculatePlan(map, state)
  command = followPathController(path, state)
  controlRobot(command)
}
```

- Single process (synchronous)
- Single program
- Poor modularity
- Develop custom tools for:
  - Visualization
  - Simulation
  - Managing configuration
  - Hardware interfaces

Naïve Robot Software - Example 2

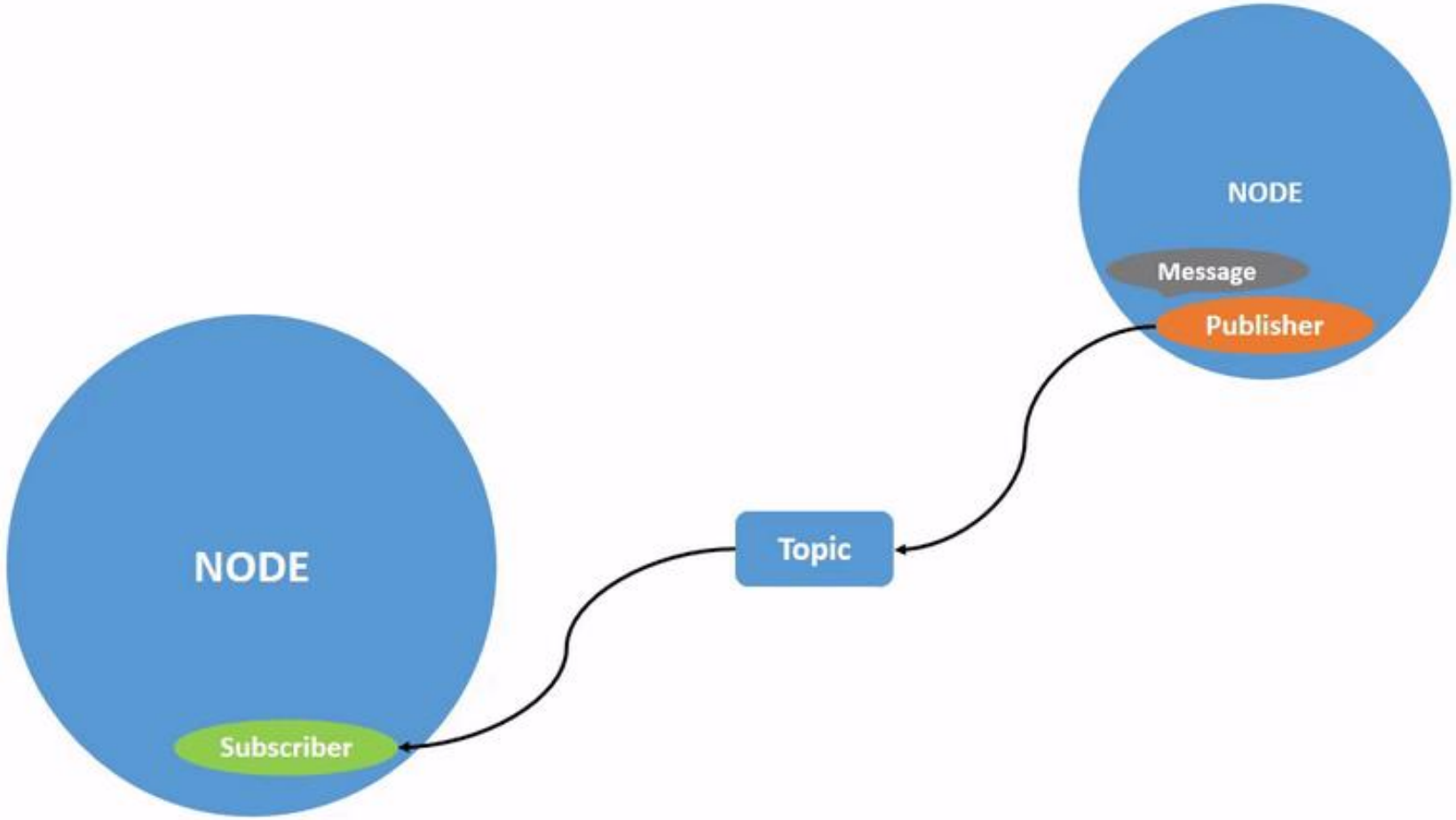


ROS - what is it?



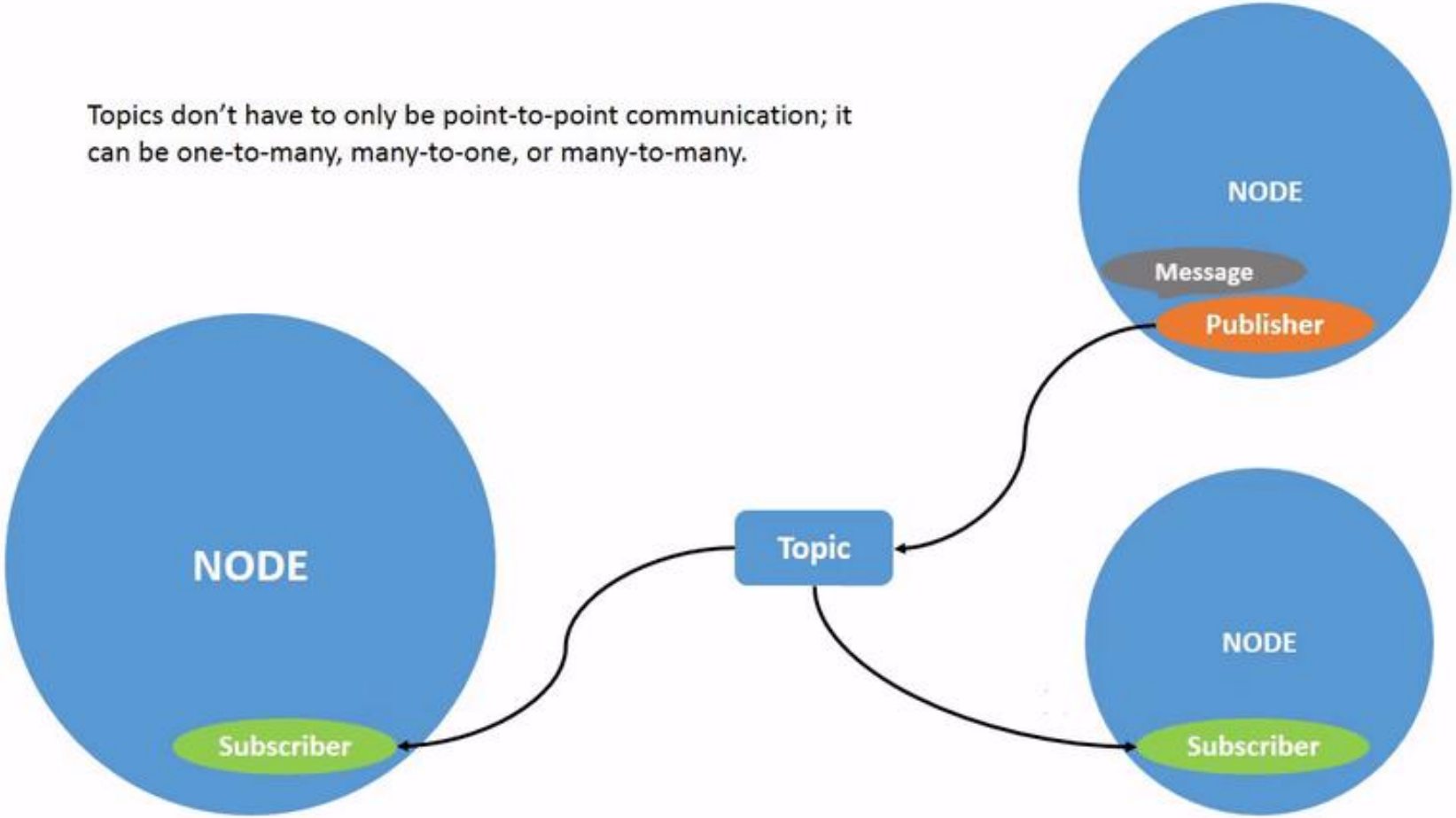
- ROS 2 is a **middleware**
  - Layer between Operating System and Applications
  - “Software glue”
- Spring Boot for robotic (kinda)

ROS graph - Nodes

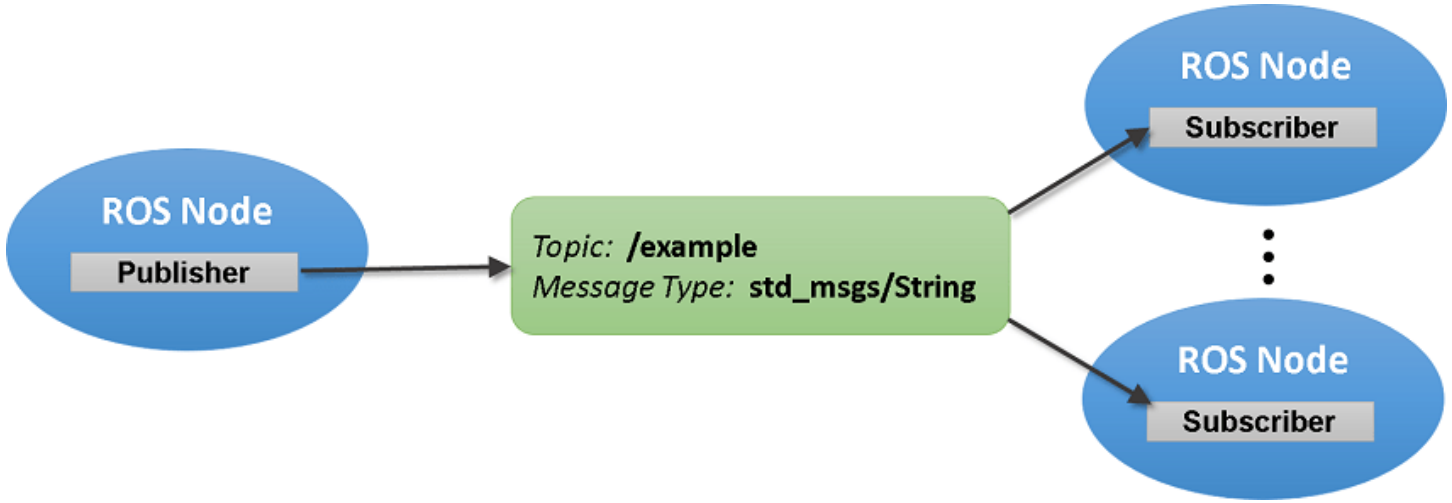


ROS graph - Nodes

Topics don't have to only be point-to-point communication; it can be one-to-many, many-to-one, or many-to-many.



ROS graph - Topic and messages



std_msgs Message Documentation

- msg/MultiArrayLayout
- msg/UInt16MultiArray
- msg/Int16
- msg/UInt32MultiArray
- msg/UInt8
- msg/MultiArrayDimension
- msg/Empty
- msg/UInt8MultiArray
- msg/String
- msg/Int32
- msg/UInt16
- msg/Int32MultiArray
- msg/UInt64MultiArray
- msg/Float64
- msg/Int8
- msg/Int16MultiArray
- msg/ColorRGBA
- msg/Float32MultiArray
- msg/Int64MultiArray
- msg/UInt32
- msg/Header
- msg/Byte
- msg/Float64MultiArray
- msg/Int64
- msg/Float32
- msg/ByteMultiArray
- msg/UInt64
- msg/Char
- msg/Bool
- msg/Int8MultiArray

autogenerated on Oct 09 2020 00:02:33

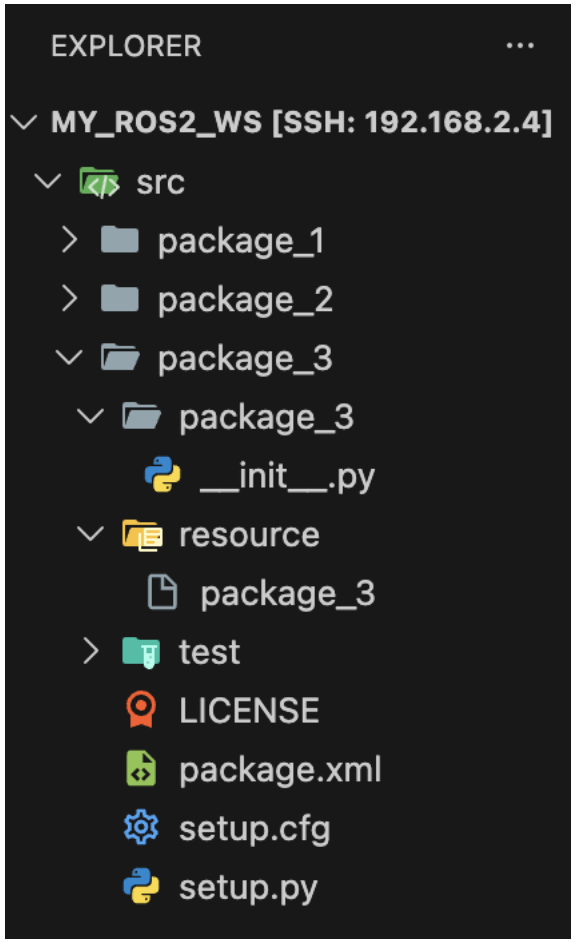
ROS graph - Discovery

- When a node is **started**,
  - Advertises its presence to other nodes on the **network** with **the same ROS domain**.
  - Nodes respond with information about themselves
  - Establish connections

- Nodes **periodically** advertise their presence so that connections can be made with **new-found** entities, even after the initial discovery period.

- Nodes advertise to other nodes when they go offline.

ROS 2 - Package(Python)



package.xml - containing meta information about the package

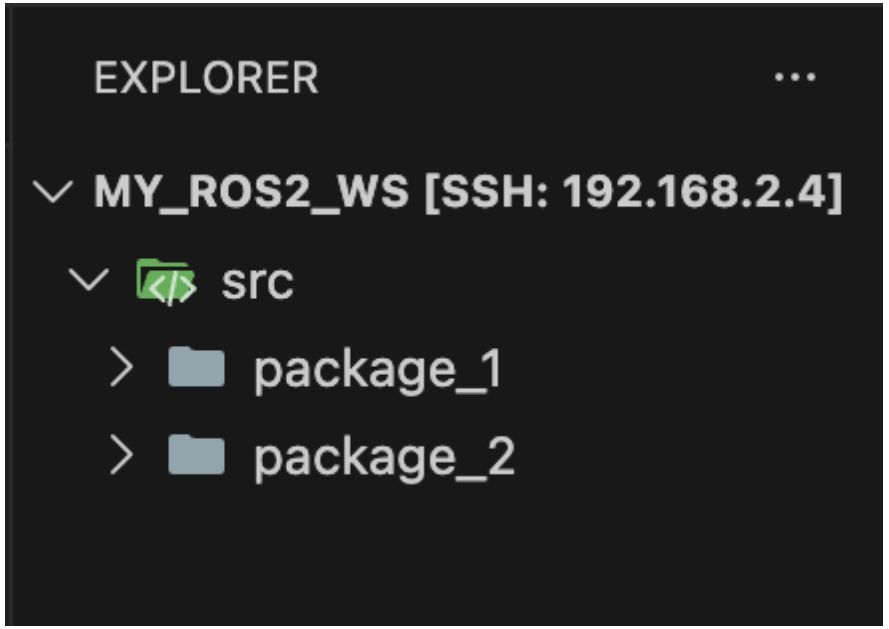
resource/<package_name> - marker file for the package

setup.cfg - is required when a package has executables, so ros2 run can find them

setup.py - containing instructions for how to install the package

<package_name> - a directory with the same name as your package, used by ROS 2 tools to find your package, contains __init__.py

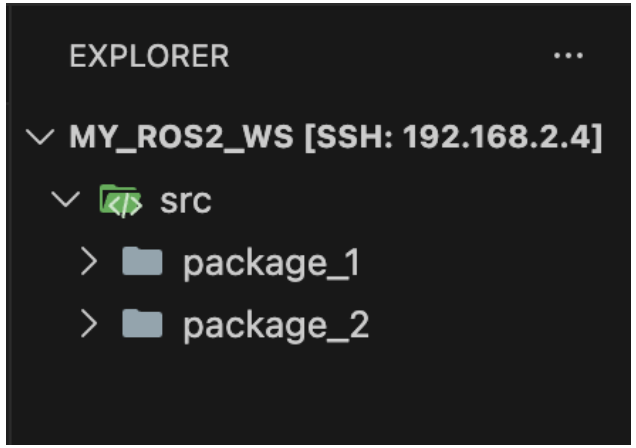
ROS 2 - Workspace



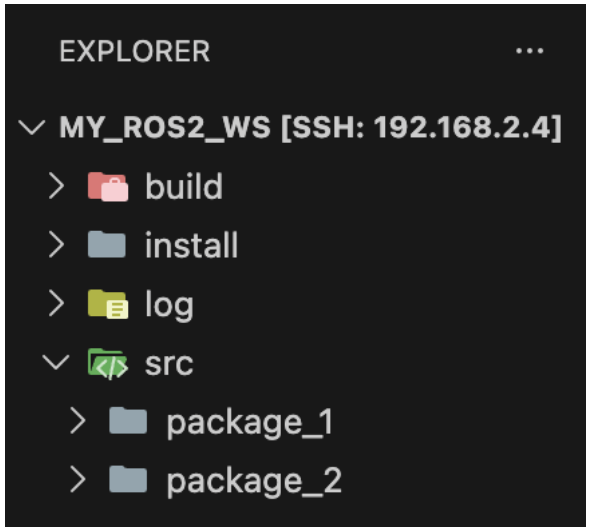
ROS 2 - Build with Colcon

- Colcon
  - command line tool for building and testing your ROS workspace
  - **colcon build** - build workspace
  - **colcon test** - run tests in workspace
- Useful flags
  - **--symlink-install** - uses ‘symlinks’ instead of copying files (Python only)
  - **--continue-on-error** – Continue other packages when package fails
  - **--packages-select** - Build only specific packages

ROS 2 - Workspace and Colcon



Workspace before Colcon build



Workspace after Colcon build

ROS 2 - Workspace and Colcon

Workspace before Colco

space after Colcon build

ROS 2 - Sourcing

- Before using ROS 2 in terminal, source your ROS 2 installation.

  source /opt/ros/humble/setup.bash

- source your ROS 2 workspace.

  source install/setup.bash

ROS 2 CLI

- **ros2**
  - **launch** – Allows running launch files
  - **node** – Display information about node
  - **param** – Allow manipulating parameters
  - **pkg** – Create package or get information about package
  - **run** – Allows running executable
  - **test** – Run launch tests
  - **topic** – Display debug information about topic
  - **wtf** – Check ROS setup and other potential issues

ROS 2 bag

- **ros2 bag** - command line tool for recording/playback data

- Commands
  - ros2 bag record <topic_name>
  - ros2 bag info <bag_file_name>
  - ros2 bag play <bag_file_name>

ROS 2 Launch

- Way to start multiple nodes at the same time.
- Launch files written in Python, XML, or YAML.
- Command
  - ros2 launch <package_name> <launch_file>

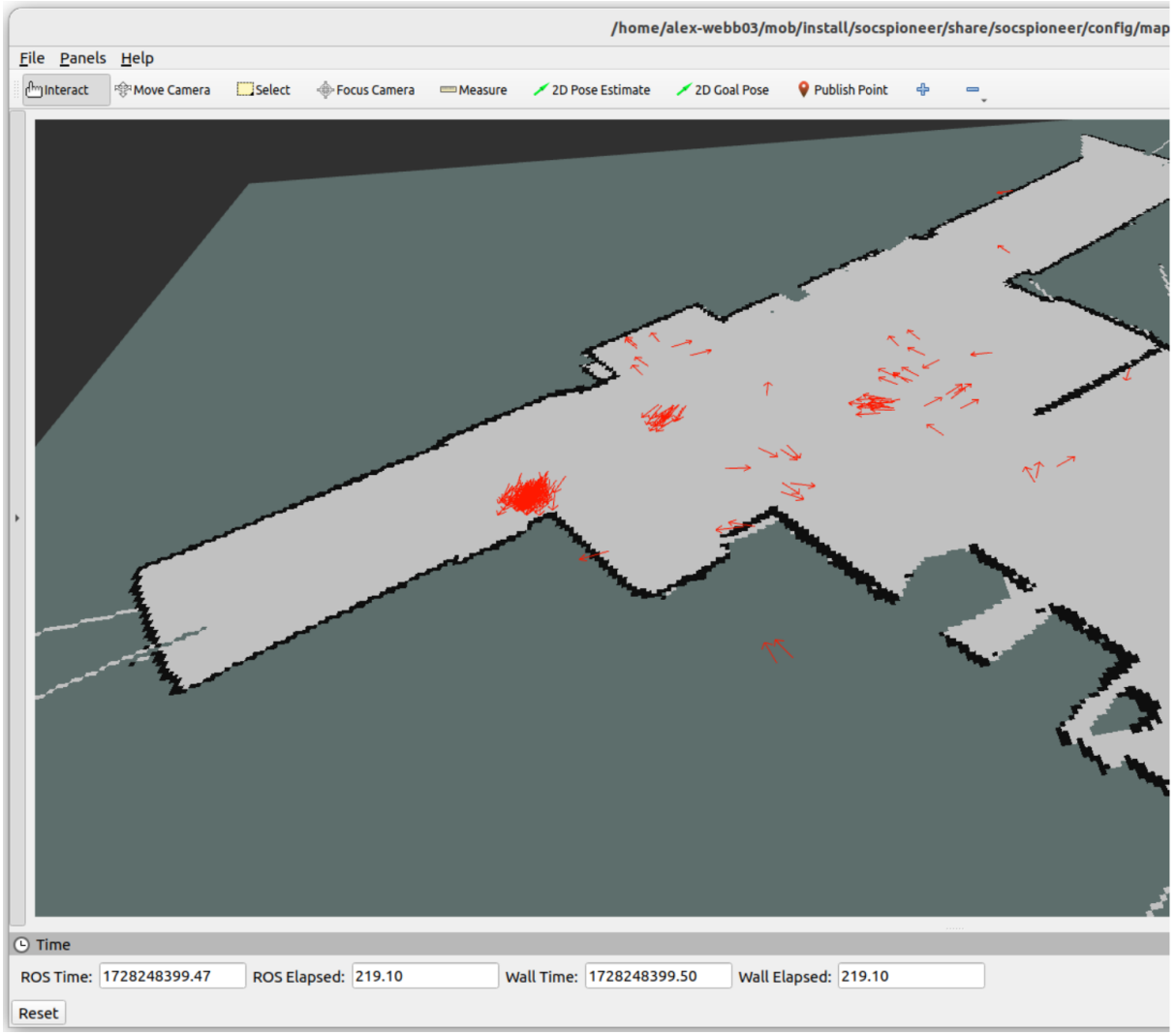
```python
from launch import LaunchDescription
from launch_ros.actions import Node

def generate_launch_description():
    return LaunchDescription([
        Node(
            package='turtlesim',
            namespace='turtlesim1',
            executable='turtlesim_node',
            name='sim'
        ),
        Node(
            package='turtlesim',
            namespace='turtlesim2',
            executable='turtlesim_node',
            name='sim'
        ),
        Node(
            package='turtlesim',
            executable='mimic',
            name='mimic',
            remappings=[
                ('/input/pose', '/turtlesim1/turtle1/pose'),
                ('/output/cmd_vel', '/turtlesim2/turtle1/cmd_vel'),
            ]
        )
    ])
```

Example – simple publisher and subscriber

ROS2 package - RViz

Visualization tool

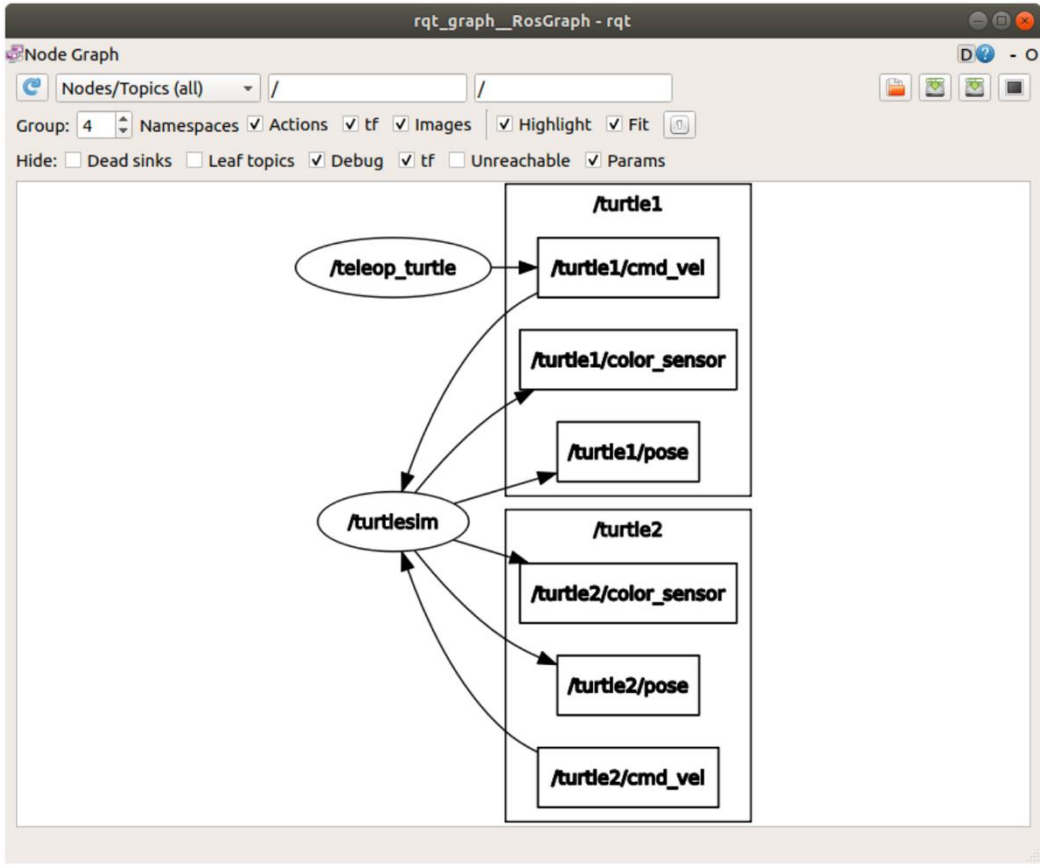
- Select topics to subscribe to
  - Visualizes message contents
  - Only specified message types
- Plugins - Add custom features
  - Display custom message types
- Config files - Save and share layouts



ROS2 package - rqt_graph

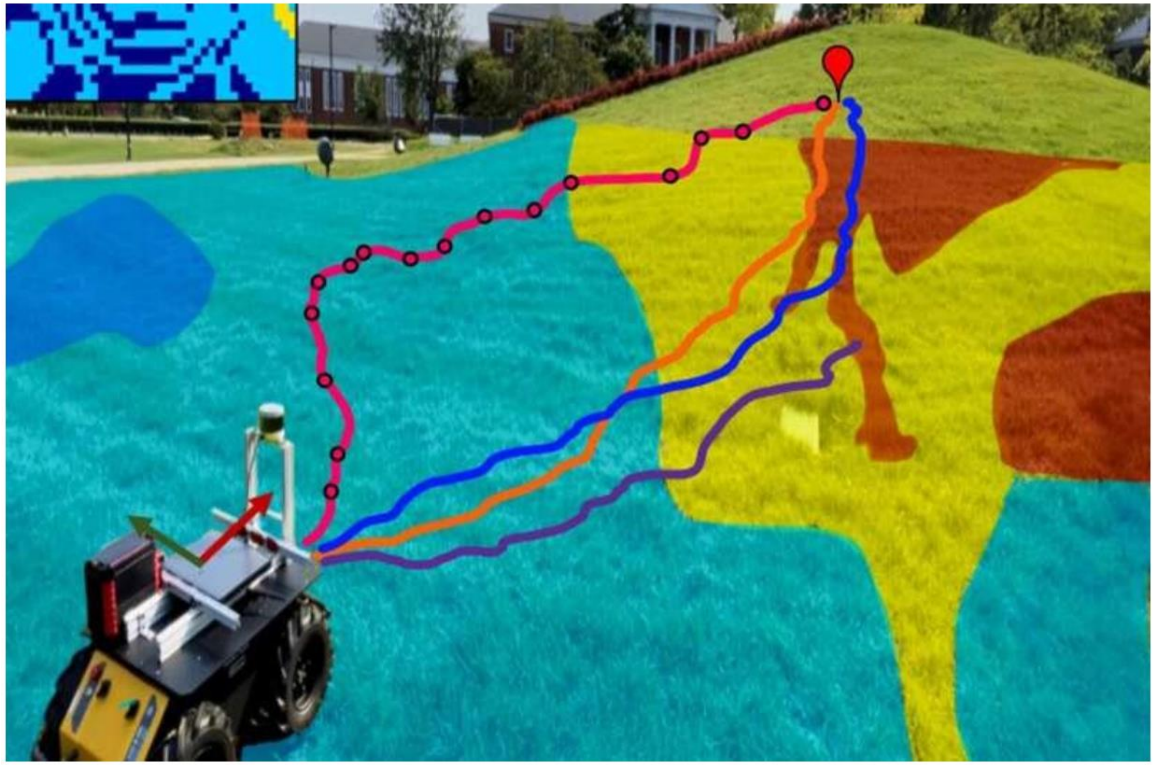
- Useful for visualizing ROS graph
- Run with

  ros2 run rqt_graph rqt_graph

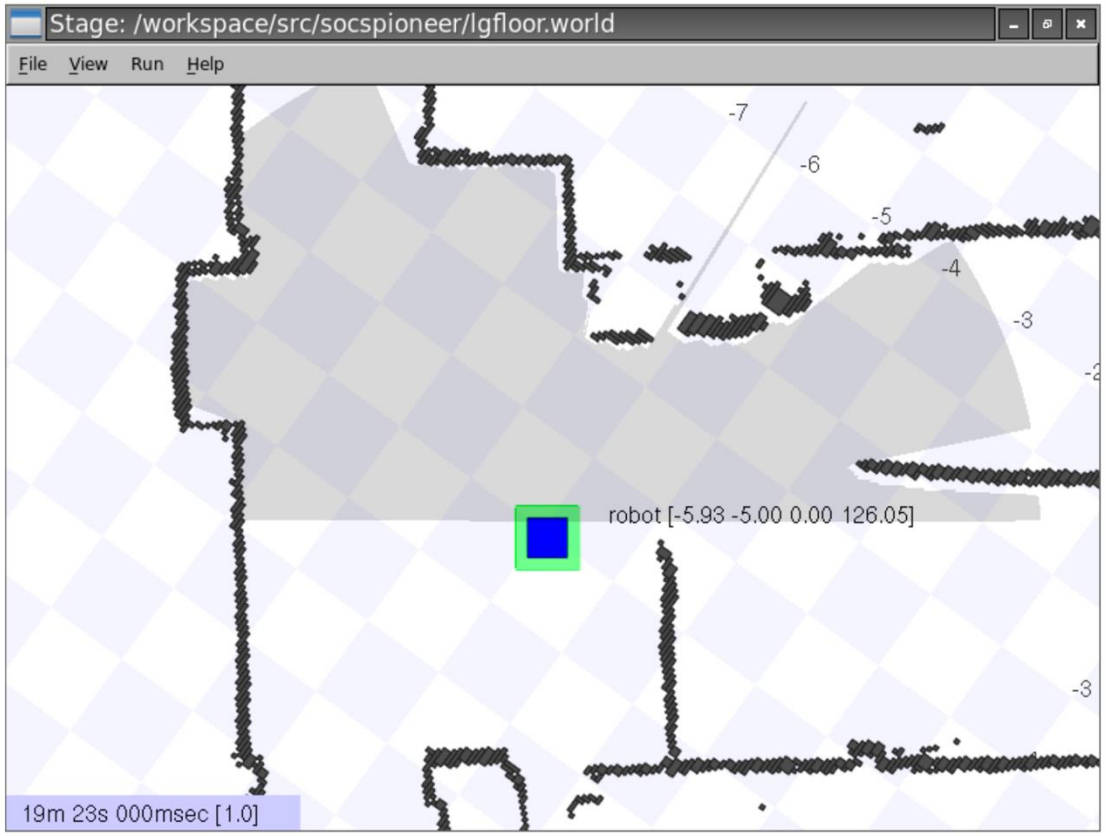


ROS2 package - Nav2

- Mobile robot navigation tool
- Map Server reads a map from a file and publishes it
- Lifecycle Manager orders nodes
- More at: https://docs.nav2.org/



ROS2 package - Stage



- Basic 2D simulator
- Load custom maps
- Control robot around
- Publish laser scanner

ROS community

- Core ROS is maintained by Open Robotics
- Packages by open-source community
- ROS index
- ROSCon