University of California, Riverside

EE106 - Spring 2025

Name: Alex Totah **SID**: 862368232 **Section**: 021

Lab 1 Report - Apr 24, 2025

1. Problem Statement

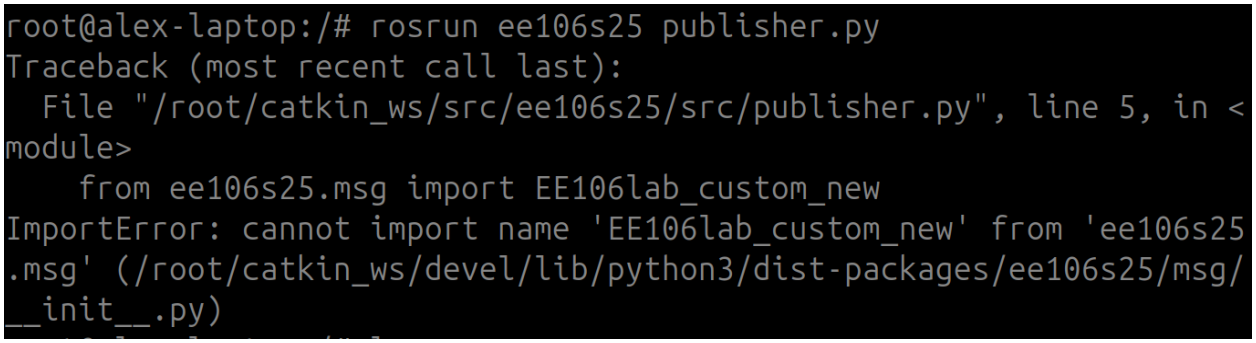
In this lab, we will explore the fundamentals of creating a simple ros publisher and subscriber system using ros1 noetic. We will develop the environment and create our own custom messages to send two random integers from the publisher node and add and print them on the subscriber node.

2. Design Idea

The idea of this lab is to create a new custom topic to publish the data of two randomly generated integers to a subscriber. The subscriber will then read in those integers and print the sum of the integers to the screen.

3. Problems and struggles

While working on this lab, myself, and many others had issues when running the subscriber and publisher files. We received the following error, “ImportError: cannot import name 'EE106lab_custom_new' from 'ee106s25.msg'”. After debugging the issue with the TA and many other students during the lab session, we learned that our error came from not including the file in the cmake lists. Therefore, when we ran the catkin_make command, the proper file was not being included in the compilation. After adding the file and making a few adjustments to the code to account for different variable names, the system ran as intended.



root@alex-laptop:/# rosrun ee106s25 publisher.py
Traceback (most recent call last):
File "/root/catkin_ws/src/ee106s25/src/publisher.py", line 5, in <module>
from ee106s25.msg import EE106lab_custom_new
ImportError: cannot import name 'EE106lab_custom_new' from 'ee106s25.msg'
(/root/catkin_ws/devel/lib/python3/dist-packages/ee106s25/msg/__init__.py)

3. Results

In ros, all software components must be running in complete coordination with each other. There are a lot of files and a lot of commands to keep track of so this lab was good practice for me to learn the basic foundations of building and running a ros package. Every little detail, like including the files, changing variables, and moving files need to be done very carefully while taking into consideration the delicate structure of the package. Also, I am currently running ros noetic out of a docker container. This is not ideal and is testing my knowledge of linux CLI. I might move to creating a virtual machine in the future, but may stick with this option to continue to improve my skills with the command line.

Code and screenshots:

GitHub: https://github.com/nander100/ee106Lab1

Publisher Code (before change):

```
#!/usr/bin/env python3

import rospy
from ee106s25.msg import EE106lab_custom

def talker():
        rospy.init_node('talker')
        pub = rospy.Publisher('custom_topic', EE106lab_custom, queue_size = 10)
        rate = rospy.Rate(10) # 10hz

        while not rospy.is_shutdown():
        msg = EE106lab_custom()
        msg.header.stamp = rospy.Time.now()
        msg.int_data = 5
        msg.float_data = 0.5
        msg.string_data = 'Hello'
        pub.publish(msg)
        rate.sleep()


if __name__ == '__main__':
        try:
        talker()
        except rospy.ROSInterruptException:
        pass
```

Subscriber Code:

```python
#!/usr/bin/env python3

from ee106s25.msg import EE106lab_custom
import rospy
from std_msgs.msg import String

def callback(data):
    rospy.loginfo(f "Got int = {msg.int_data}",
    f "float = {msg.float_data:.2f}",
    f "str = {msg.string_data}",
    f "time_delay = {(rospy.Time.now()-msg.header.stamp).to_sec():.3f}s"
    )

def listener_custom():
    rospy.init_node('listener_custom')
    rospy.Subscriber('custom_topic', EE106lab_custom, callback)
    rospy.spin()


if __name__ == '__main__':
    listener()
```

Create a new ROS message type

```
Header header
int32 int_rand_1
int32 int_rand_2
```

Update the subscriber

```python
#!/usr/bin/env python3
from ee106s25.msg import EE106lab_custom_new
import rospy
from std_msgs.msg import String

def callback(data):
    sum = data.int_rand1 + data.int_rand2
    # Combine all information into a single string
    log_msg = (f"random int 1 = {data.int_rand1}, "
               f"random int 2 = {data.int_rand2}, "
               f"sum = {sum}"
               f"time_delay = {(rospy.Time.now()-data.header.stamp).to_sec():.3f}s")
    rospy.loginfo(log_msg)

def listener_custom():
    rospy.init_node('listener_custom')
    rospy.Subscriber('custom_topic', EE106lab_custom_new, callback)
    rospy.spin()

if __name__ == '__main__':
    listener_custom()
```

Update the publisher:

```python
#!/usr/bin/env python3

import rospy
import random
from ee106s25.msg import EE106lab_custom_new

def talker():
    rospy.init_node('talker')
    pub = rospy.Publisher('custom_topic', EE106lab_custom_new, queue_size = 10)
    rate = rospy.Rate(10) # 10hz

    while not rospy.is_shutdown():
      msg = EE106lab_custom_new()
      msg.header.stamp = rospy.Time.now()
      msg.int_rand1 = random.randint(1, 100)
      msg.int_rand2 = random.randint(1, 100)
      pub.publish(msg)
      rate.sleep()


if __name__ == '__main__':
    try:
        talker()
    except rospy.ROSInterruptException:
        pass
```

Subscriber output:

root@alex-laptop:~/catkin_ws# rosrun ee106s25 subscriber.py
[INFO] [1745559632.560962]: random int 1 = 31, random int 2 = 34, sum = 65time_delay = 0.001s
[INFO] [1745559632.660773]: random int 1 = 64, random int 2 = 40, sum = 104time_delay = 0.001s
[INFO] [1745559632.760846]: random int 1 = 64, random int 2 = 31, sum = 95time_delay = 0.001s
[INFO] [1745559632.860655]: random int 1 = 28, random int 2 = 12, sum = 40time_delay = 0.001s
[INFO] [1745559632.960757]: random int 1 = 28, random int 2 = 53, sum = 81time_delay = 0.001s
[INFO] [1745559633.060787]: random int 1 = 99, random int 2 = 58, sum = 157time_delay = 0.001s

Full system running: