

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕТРОЗАВОДСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Институт математики и информационных технологий

Отчет о прохождении производственной практики

Выполнил студент(ка) группы 22207

Анисимов Никита Николаевич

(ФИО)

Направление подготовки:

Системное и прикладное программное
обеспечение

Место прохождения практики:

Кафедра ИМО, ЦИИ

Сроки прохождения практики:

30.05.2022 – 09.06.2022

Руководитель практики:

к.т.н., доцент

Богоявленская Ольга Юрьевна

Оценка _____

Дата _____

Петрозаводск 2022 г.

Содержание

Введение	3
1 Система ROS	3
1.1 Теория	3
1.2 Практика	3
2 Publisher and subscriber	3
2.1 Теория	3
2.2 Практика	3
3 Использование в проектах	5
Заключение	5
Приложение А. Код издателя	6
Приложение Б. Код подписчика	7

Введение

Цель практики: изучить механизм publisher/subscriber в системе ROS.

Задачи практики:

- изучить систему ROS
- изучить концепцию подписчика и издателя
- написать подписчика и издателя
- протестировать работу программ
- рассмотреть использование механизма в проектах

1 Система ROS

1.1 Теория

Я работал с системой ROS (Robot Operating System), которая является ОС для программирования роботов. Помимо реализации всех стандартных служб операционной системы, ROS предоставляет набор программ для удобной работы с пакетами, их сборки и тестирования. В частности систему catkin для сборки пакетов.

1.2 Практика

Я поставил VirtualBox с Ubuntu, на которую планировал поставить ROS Noetic. Далее поставил саму систему по инструкции с официального сайта, а так же настроил catkin workspace.

2 Publisher and subscriber

2.1 Теория

ROS основан на архитектуре графов, где обработка данных происходит в программах, называемых узлами (nodes), которые могут получать и передавать сообщения (messages) между собой. Одним из способов взаимодействия между ними в ROS является модель издатель-подписчик (publisher-subscriber). Ключевым объектом здесь является тема (topic), в которую издатель публикует сообщения, а подписчик считывает их оттуда. У одной темы может быть несколько как подписчиков, так и издателей. Тип сообщений можно использовать как стандартный, так и задать самому с помощью msg файлов.

2.2 Практика

Я создал пакет beginner_tutorials для моих программ (nodes). В манифест и CMakeLists добавил все необходимые зависимости. Так же создал msg файл Num.msg где описал формат сообщений. Далее написал две программы на языке C++:

- talker.cpp – издатель

- listener.cpp – подписчик

Обе программы взаимодействовали через тему chatter. Издатель публиковал значения увеличивающегося счетчика, а подписчик считывал их и выводил в консоль. Исходные коды находятся в приложении.

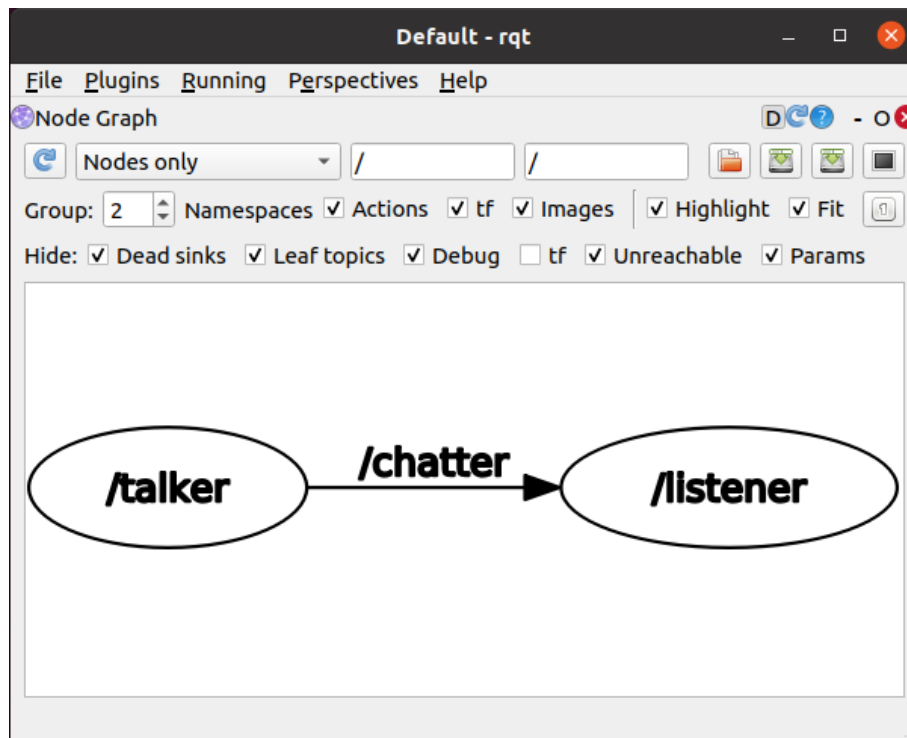


Рис. 1: Созданный с помощью rqt граф, изображающий происходящее в системе

```
nikita@nikita-VirtualBox:~/catkin_ws$ rosmmsg show Num  
[beginner_tutorials/Num]:  
int64 num
```

Рис. 2: Тип сообщений Num

```
nikita@nikita-VirtualBox: ~
[ INFO] [1654541167.579801267]: I said: [540] [ INFO] [1654541167.580272641]: I heard: [540]
[ INFO] [1654541167.679261709]: I said: [541] [ INFO] [1654541167.679782604]: I heard: [541]
[ INFO] [1654541167.779793640]: I said: [542] [ INFO] [1654541167.780592316]: I heard: [542]
[ INFO] [1654541167.879439374]: I said: [543] [ INFO] [1654541167.879993675]: I heard: [543]
[ INFO] [1654541167.979961594]: I said: [544] [ INFO] [1654541167.980535210]: I heard: [544]
[ INFO] [1654541168.079413174]: I said: [545] [ INFO] [1654541168.079847421]: I heard: [545]
[ INFO] [1654541168.179905853]: I said: [546] [ INFO] [1654541168.180671393]: I heard: [546]
[ INFO] [1654541168.285933419]: I said: [547] [ INFO] [1654541168.287439024]: I heard: [547]
[ INFO] [1654541168.380155876]: I said: [548] [ INFO] [1654541168.380639519]: I heard: [548]
[ INFO] [1654541168.479506570]: I said: [549] [ INFO] [1654541168.480071334]: I heard: [549]
[ INFO] [1654541168.579406570]: I said: [550] [ INFO] [1654541168.580154319]: I heard: [550]
[ INFO] [1654541168.679775143]: I said: [551] [ INFO] [1654541168.680563850]: I heard: [551]
[ INFO] [1654541168.779308049]: I said: [552] [ INFO] [1654541168.780381594]: I heard: [552]
[ INFO] [1654541168.879624050]: I said: [553] [ INFO] [1654541168.880105597]: I heard: [553]
[ INFO] [1654541168.980163050]: I said: [554] [ INFO] [1654541168.980702979]: I heard: [554]
[ INFO] [1654541169.079679303]: I said: [555] [ INFO] [1654541169.080158934]: I heard: [555]
[ INFO] [1654541169.179246860]: I said: [556] [ INFO] [1654541169.179722841]: I heard: [556]
[ INFO] [1654541169.284424019]: I said: [557] [ INFO] [1654541169.284738773]: I heard: [557]
[ INFO] [1654541169.379331526]: I said: [558] [ INFO] [1654541169.379812590]: I heard: [558]
[ INFO] [1654541169.479687115]: I said: [559] [ INFO] [1654541169.480277786]: I heard: [559]
[ INFO] [1654541169.579293164]: I said: [560] [ INFO] [1654541169.579977585]: I heard: [560]
[ INFO] [1654541169.680077322]: I said: [561] [ INFO] [1654541169.680591303]: I heard: [561]
[ INFO] [1654541169.779284385]: I said: [562] [ INFO] [1654541169.779937030]: I heard: [562]
```

Рис. 3: Вывод программ

3 Использование в проектах

При разработки роботов концепция издатель-подписчик очень важна, т.к. обеспечивает обмен информацией между отдельными компонентами системы. Например узел А отвечает за датчики робота, и ему необходимо передовать данные о препятствиях на пути узлу Б, который отвечает за движение робота. Узел А является издателем и публикует сообщение об обстановке в тему "препятствие". Узел Б считывает оттуда сообщения и при уведомлении о препятствии останавливает движение робота и перестраивает маршрут.

Заключение

В данной практике была изучена система ROS и механизм publisher/subscriber, который в ней используется. Были написаны и протестированы программы, реализующие этот механизм, а также рассмотрен пример его использования в проекте робота.

Приложение А. Код издателя

```
1  #include "ros/ros.h"
2  #include "beginner_tutorials/Num.h"
3
4  int main(int argc, char **argv)
5  {
6      // Инициализация узла
7      ros::init(argc, argv, "talker");
8
9      // Интерфейс для создания узлов
10     ros::NodeHandle n;
11
12     // Создание издателя, который будет публиковать сообщения
13     // в тему chatter в формате Num
14     ros::Publisher chatter_pub =
15         n.advertise<beginner_tutorials::Num>("chatter", 1000);
16
17     // Частота цикла
18     ros::Rate loop_rate(10);
19
20     // Счетчик
21     int count = 0;
22
23     // Пока нет ошибок публикуем сообщения
24     while (ros::ok())
25     {
26         // Создание сообщения
27         beginner_tutorials::Num msg;
28         msg.num = count;
29
30         // Вывод в консоль
31         ROS_INFO("I said: [%ld]", msg.num);
32
33         // Публикация сообщения
34         chatter_pub.publish(msg);
35
36         // Задержка
37         loop_rate.sleep();
38
39         // Увеличение счетчика
40         ++count;
41     }
42     return 0;
43 }
```

Приложение Б. Код подписчика

```
1  #include "ros/ros.h"
2  #include "beginner_tutorials/Num.h"
3
4  void chatterCallback(const beginner_tutorials::Num::ConstPtr& msg)
5  {
6      // Консольный вывод
7      ROS_INFO("I heard: [%ld]", msg->num);
8  }
9
10 int main(int argc, char **argv)
11 {
12     // Инициализация узла
13     ros::init(argc, argv, "listener");
14
15     // Интерфейс для создания узлов
16     ros::NodeHandle n;
17
18     // Создание подписчика темы chatter, который при считывании сообщения
19     // вызывает функцию chatterCallback с сообщением в параметре
20     ros::Subscriber sub = n.subscribe("chatter", 1000, chatterCallback);
21
22     // Вход в цикл считыванию сообщений
23     ros::spin();
24     return 0;
25 }
```
