# 1. Концепция файловой системы ROS

**Package**

Основная единица организации программного обеспечения в ROS. В основном пакет содержит выполняемые процессы ROS (узлы или ноды - nodes), библиотеки на основе ROS, наборы данных, конфигурационные файлы и прочие полезные данные. Пакет – это минимальная единица для компиляции и релиза в ROS.

**Package Manifests**

(package.xml) содержит данные о пакете, включая имя, версию, описание, информацию о лицензировании, зависимости и прочую информацию, такую как экспортируемые пакеты.

**Message types**

Описание сообщений, хранится в my\_package/msg/MyMessageType.msg, определяет структуры данных для сообщений, передаваемых в ROS.

**Service types**

Описание сервисов, хранится в my\_package/srv/MyServiceType.srv, определяет структуры данных для запроса и ответа сервисов в ROS.

# **2.** Граф вычислений ROS – это одноранговая сеть процессов ROS, обрабатывающих данные.

## 2.1. Элементы графа

* **Nodes** – это процессы, выполняющие вычисления. Системы использующие ROS состоят из модулей, система управления роботом включает в себя множество нод. Например, одна нода управляет лазерным дальномером, другая – моторами колёс, третья нода определяет положение в пространстве, четвёртая планирует траекторию движения, пятая предоставляет графическое представление системы, и т.д. ROS ноды разрабатывают с использованием клиентских библиотек ROS, таких как roscpp или rospy.
* **Master -** мастер-процесс ROS обеспечивает регистрацию имён и наблюдение за всем вычислительным графом. Без мастер-процесса ноды не смогли бы найти друг друга, обмениваться сообщениями или вызывать сервисы.
* **Parameter** **Server -** сервер параметров позволяет хранить данные с доступом по ключу в централизованном хранилище. В настоящее время Сервер Параметров является частью Мастера.
* **Topics -** сообщения передаются через транспортную систему через механизм публикации/подписки. Нода отправляет сообщение, публикуя (publish) его в определённом Топике. Топик – это имя, идентифицирующее содержание сообщения. Нода, заинтересованная в определённых данных, осуществляет подписку (subscribe) на соответствующий Топик. Для одного топика может существовать несколько параллельно публикующие/подписанных на него Нод, равно как и одна Нода может публиковать сообщения в и/или подписываться на несколько Топиков. В общес случае, публикаторы/подписчики не оказывают влияния друг на друга. Идея заключается в отделении производства информации от её использования. Логически Топик может быть представлен как строго типизированная шина сообщений. У каждой шины есть наименование, и любой элемент может подсоединиться к шине для получения и отправки сообщений соответствующего типа.
* **Services -** модель публикации/ подписки является очень гибкой, но её схема односторонней передачи сообщений «многие-многим» не подходит для взаимодействий типа «запрос-ответ», которые часто нужны в распределённой системе. Механизм «запрос-ответ» реализован через Сервисы. Сервис определяется парой структур сообщений – одна для запроса и одна для ответа. Нода предоставляет сервис, используя определённое Имя сервиса, клиент использует сервис, отправляя сообщение-запрос и ожидая ответа. Клиентские библиотеки ROS обычно представляют это взаимодействие для программиста в виде вызова удалённой процедуры.
* **Bags -** контейнеры предоставляют форматы для записи и воспроизведения потоков ROS-сообщений. Контейнеры являются важным механизмом для записи данных, например, данных с сенсоров, которые трудно собрать, но необходимо сохранять для разработки и тестирования алгоритмов.

## 2.2. Пример графа

Мастер ROS играет роль сервера имён в Вычислительном графе ROS. Он хранит информацию о Топиках и Сервисах для ROS-нод. Ноды сообщают Мастеру свою регистрационную информацию. В процессе коммуникации с Мастером Ноды могут получать информацию о других зарегистрированных Нодах, устанавливать с ними связь. Мастер также осуществляет обратные вызовы к Нодам, когда регистрационная информация меняется, что позволяет одам динамически устанавливать связи по мере запуска новых Нод.

Ноды связываются с другими Нодами напрямую. Мастер только предоставляет информацию для поиска, подобно DNS серверу. Ноды, которые подписываются на топик, запрашивают связь с Нодами, которые публикуют данные в этот топик, и устанавливают эту связь через соответсвующий согласованный протокол. Наиболее часто используемый протокол – TCPROS, он использует стандартные TCP/IP сокеты.

ГРАФ

Изображение выглядит как диаграмма

Автоматически созданное описание

# 3. Установка окружающей среды

Инструкция на официальной странице:

https://docs.ros.org/en/galactic/Installation/Ubuntu-Install-Debians.html

**4. Примеры**

EXAMPLE 1: Create a node on C++

**1. create dirs for workspace**

|  |
| --- |
| mkdir ros2\_ws  cd ros2\_ws  mkdir src  cd src |

**2. source**

|  |
| --- |
| source /opt/ros/galactic/setup.bash |

**3. init package**

|  |
| --- |
| ros2 pkg create ${pkg\_name} --build-type ament\_cmake |

ex:

|  |
| --- |
| ros2 pkg create ros2\_cpp\_pkg --build-type ament\_cmake --dependencies rclcpp |

**4. create .cpp files in ../ros2\_cpp\_pkg/src/**

*see appendices*

**5. set up cmake**

|  |
| --- |
| add\_executable (simple\_node src/simple.cpp)  ament\_target\_dependencies (simple\_node rclcpp)  install (TARGETS  simple\_node  DESTINATION lib/$(PROJECT\_NAME}  ) |

Instead of simple.cpp write name of .cpp file and instead of simple\_node a name of executable file

EXAMPLE 2.1: Create a publisher and subscriber on C++

https://docs.ros.org/en/foxy/Tutorials/Beginner-Client-Libraries/Writing-A-Simple-Py-Publisher-And-Subscriber.html

EXAMPLE 2.2: Create a publisher and subscriber on Python

https://docs.ros.org/en/foxy/Tutorials/Beginner-Client-Libraries/Writing-A-Simple-Py-Publisher-And-Subscriber.html