Week 12:- Robotic Operating System [ROS]

Ros:- Robotic Operating System

* It is a set of software libraries and tools made to ease the development of robotic applications.
* Ros is widely used in robotic companies, robotic research institutes for designing, building and simulating a robot model.
* It is a open source middleware framework to create a robotics standard.
* It is very useful in the field of automation & Robotics.
* A Ros is a BSD-licensed system for controlling robotic components from a PC.
* It uses a publish/subscribe communication model, where nodes (software modules) communicate by publishing messages to, and subscribing to, named topics.

Ros features:-

1. Ros is general:- The same base code and knowledge can be applied to many different kinds of robots [robotic arm, drone]
2. Ros packages for everything:- There are many Ros packages for almost any robotic applications [for joystick control, mapping navigation etc]
3. Ros is language agnostic [sub programs can be written in any language.
   We can easily communicate between a Python node & C++ node.
   lot of reusability allowed in Ros.
4. Ros has great simulation tools:- tools like Rviz & Gazerbo.
   provides robot design, debugging & simulation in a real 3D environments so its saves designer time and money.

5. You can Control multiple robots with Ros:- Ros can work with multiple Ros masters and all robots can communicate between each other.

6. Ros is Light:- You can quickly install the core packages and get started in a few minutes.

7. More Compatible Ros products:- It supports many robotic products like grippers, controller boad boards etc. For each and every product there is a separate packages available.

8. Ros is an open source project with permissive license It is open source frame work relased under BSD license. It allows you to modify and use the code for commercial purposes.

Ros Libraries and its uses

* ros_control - Ros main control loop
* URDF - represent a 3D model of your robot.
* Moveit - For path and motion planning.
* Navigate stack - to move mobile robot.
* Rviz - for 3D visualisation.
* Gazebo - powerful simulation tool.
* Rosbridge - TO Communicate between a Ros and non-Ros environment.
* roscpp - for creating, managing & interacting with Ros nodes.
* Sensor_msgs - for Senser related data such as imager. laser scans etc

Ros core and Communication tools :-

Ros packages containing small programs, called nodes.

How to make those programs communicate between each other?

Here, Ros comes with 3 main communication tools.

* Topics :- Those will be used mainly for sending data streams between nodes.
  example :- You are monitoring the temperature of a motor on the robot. The node monitoring this motor will send a data stream with the temperature. Now, any other node can subscribe to this topic and get the data.

* Services :- They will allow you to create a simple synchronous client/server communication between nodes. Services can send inputs and receive a reply. The service node that sends the request is called a "service client" and the one that sends the response is called a "service server".

* Actions :- A little bit more complex, they are in fact based on topics. They exist to provide you with an asynchronous client/server architecture, where the client can send a request that takes a long time. The client can asynchronously monitor the state of the server and cancel the request any time.

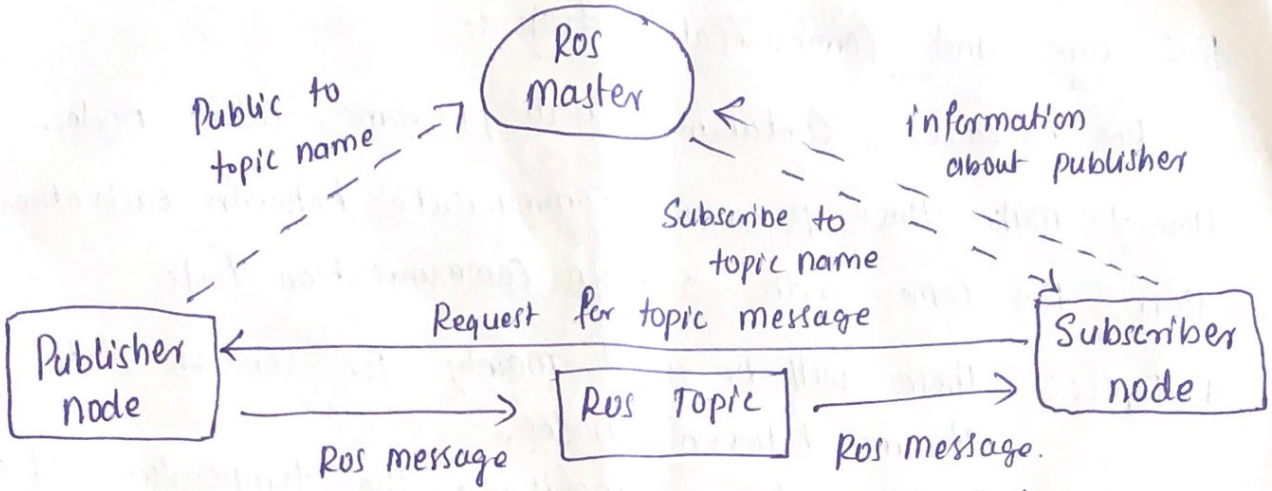

fig :- Ros Communication tools.

Recommended PC requirements for installing Ros

* Memory – 1 GB.
* Graphic Card – Nvidia Geforce 7200 GS
* CPU – Intel Core i3 / Inter Core 2 duo
* File Size – 15MB
* OS – Windows XP, Vista, 7, 8, 10

Procedure to install Ros

Step 1 :- Install Virtual Box [www. Virtualbox. org]
Step 2 :- Configurating the Virtual box like VMware.
Step 3 :- Booting and installing Ubuntu on Virtual box.
Step 4 :- Install Ros kinetic kame.
Step 5 :- Install Arduino IDE
step 6 :- Including Ros library.