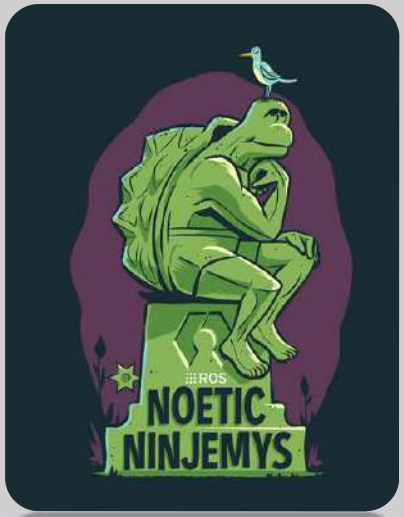
Installing ROS on Ubuntu

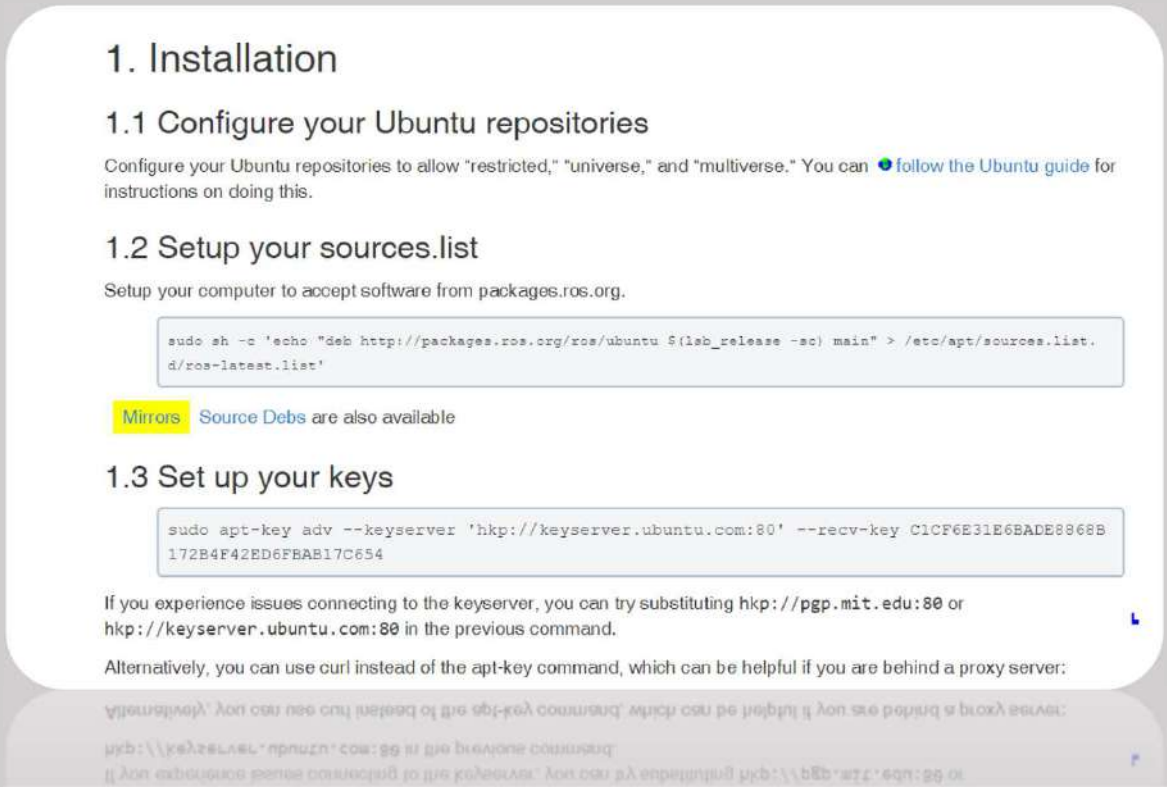
- Ubuntu Version : 20.04 LTS 64 bit, Focal
- ROS Distribution : ROS Noetic Ninjemys(Stable)



+

Installing ROS on Ubuntu

- Installation link : http://wiki.ros.org/noetic/Installation



1. Installation

1.1 Configure your Ubuntu repositories

Configure your Ubuntu repositories to allow "restricted," "universe," and "multiverse." You can follow the Ubuntu guide for instructions on doing this.

1.2 Setup your sources.list

Setup your computer to accept software from packages.ros.org.

```
sudo sh -c 'echo "deb http://packages.ros.org/ros/ubuntu $(lsb_release -sc) main" > /etc/apt/sources.list.d/ros-latest.list'
```

Mirrors Source Debs are also available

1.3 Set up your keys

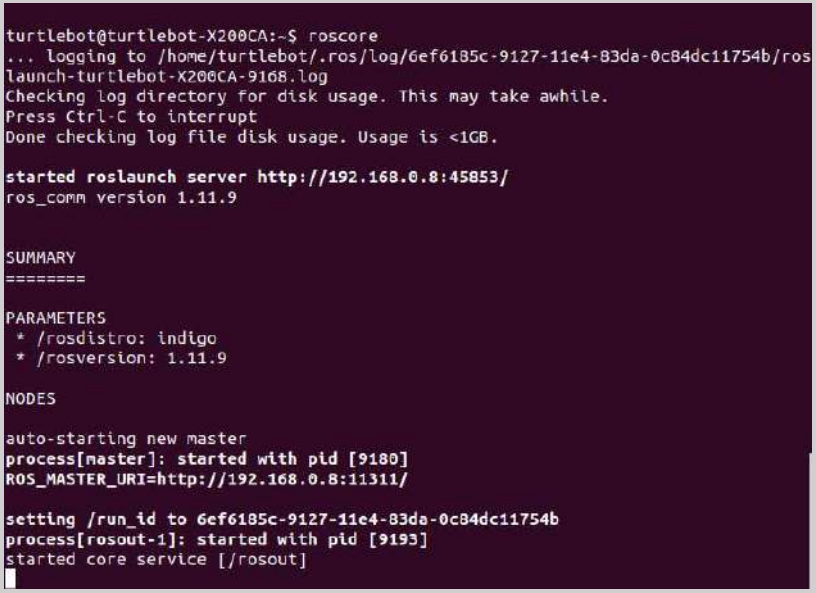
```
sudo apt-key adv --keyserver 'hkp://keyserver.ubuntu.com:80' --recv-key C1CF6E31E6BADE8868B172B4F42ED6FBAB17C654
```

If you experience issues connecting to the keyserver, you can try substituting `hkp://pgp.mit.edu:80` or `hkp://keyserver.ubuntu.com:80` in the previous command.

Alternatively, you can use curl instead of the apt-key command, which can be helpful if you are behind a proxy server:

Testing the ROS installation

- $ roscore – Test this command in terminal to verify ROS installation



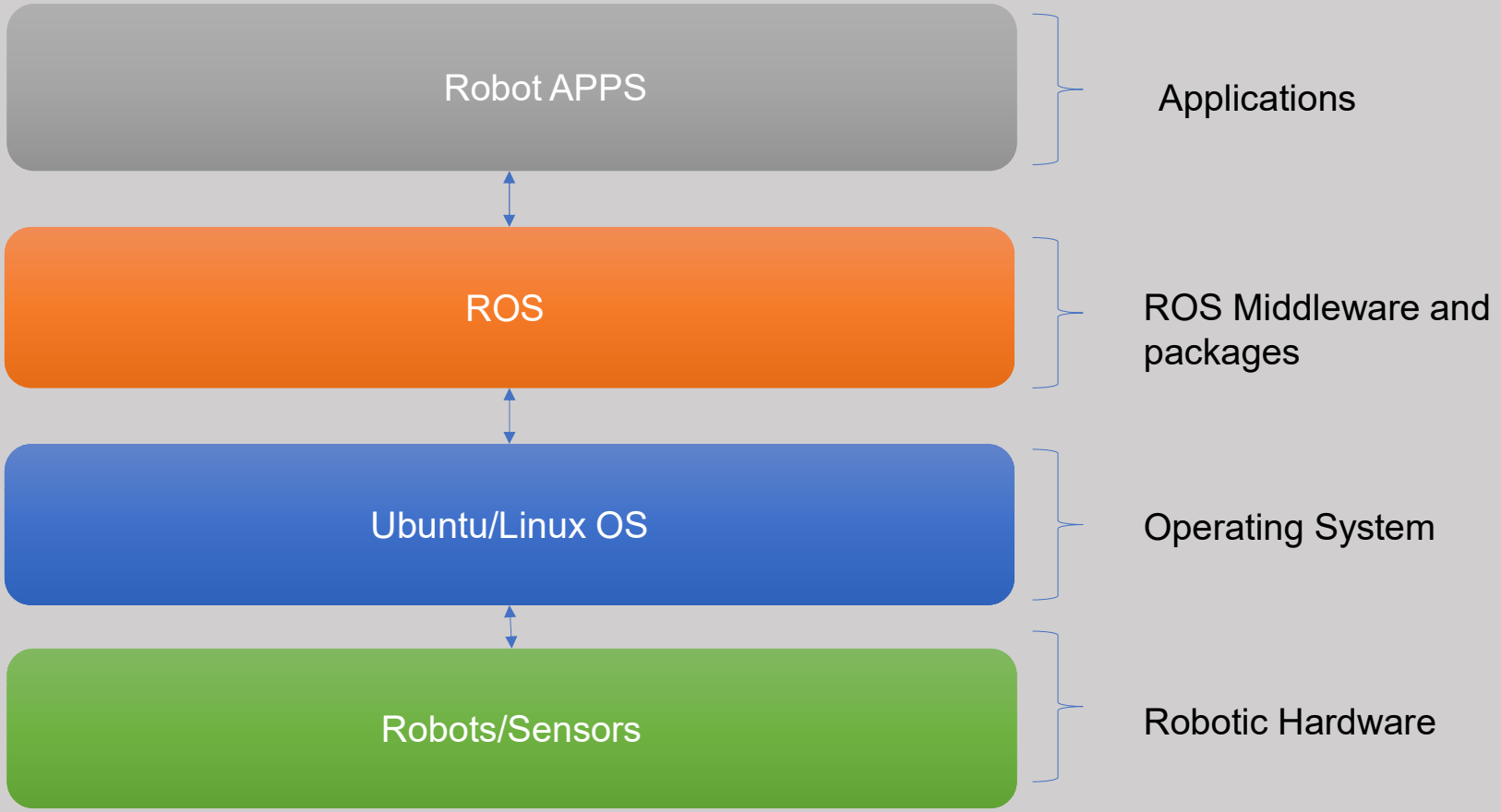
The ROS Concepts

Understanding what's inside ROS

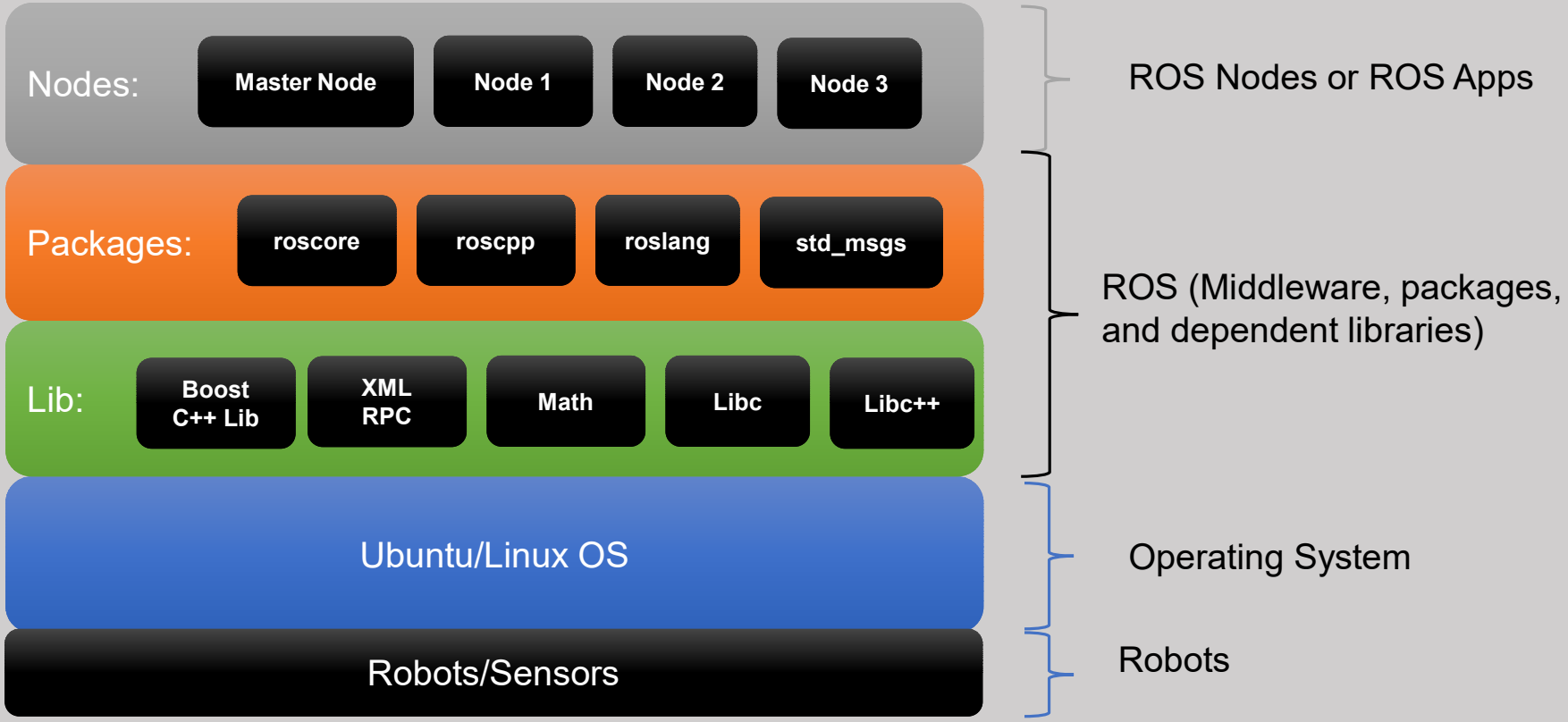
The ROS Architecture

Understanding what's inside ROS

ROS Software Architecture



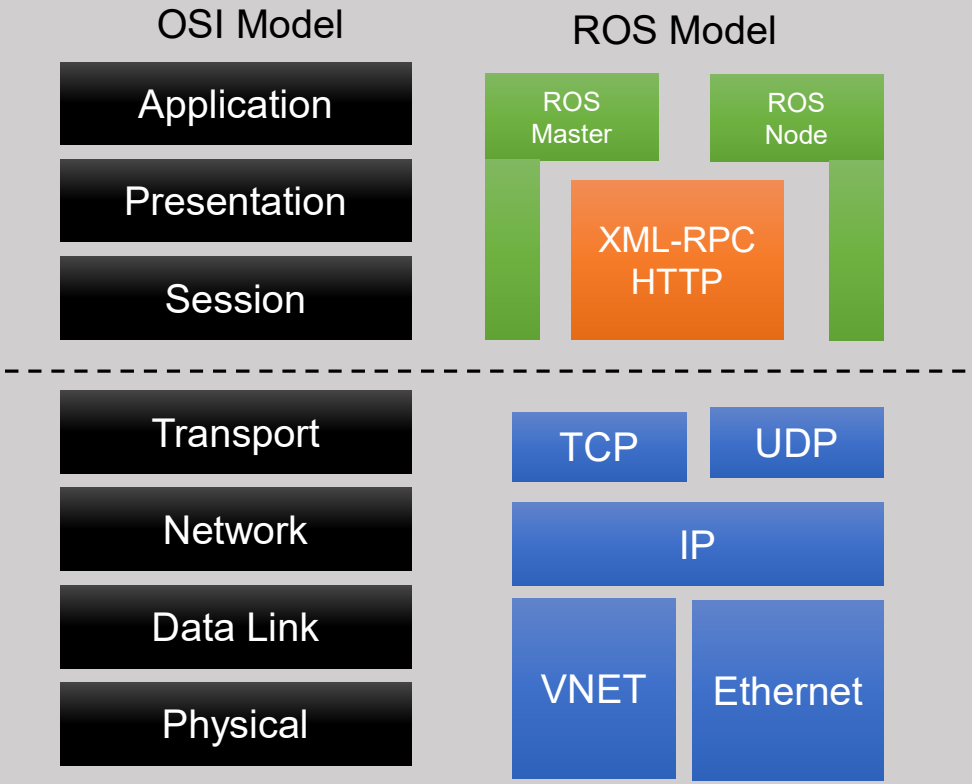
ROS Software Architecture



ROS Software Architecture -Terminology

- **Middleware:** A bridge between operating system and user applications.
  - In case of ROS, it is for inter process communication application
- **XML RPC**: XML Remote Procedure Call
  - A protocol which uses XML to encode its calls and HTTP as a transport mechanism
- **ROS Packages**: Software in ROS is organized in packages
  - It may contain set of ROS node, ROS independent library, dataset, configuration files, third-party piece of software or anything else that logically constitutes a useful module [ROS Wiki]

ROS Communication Model



* **OSI: Open Systems Interconnection**

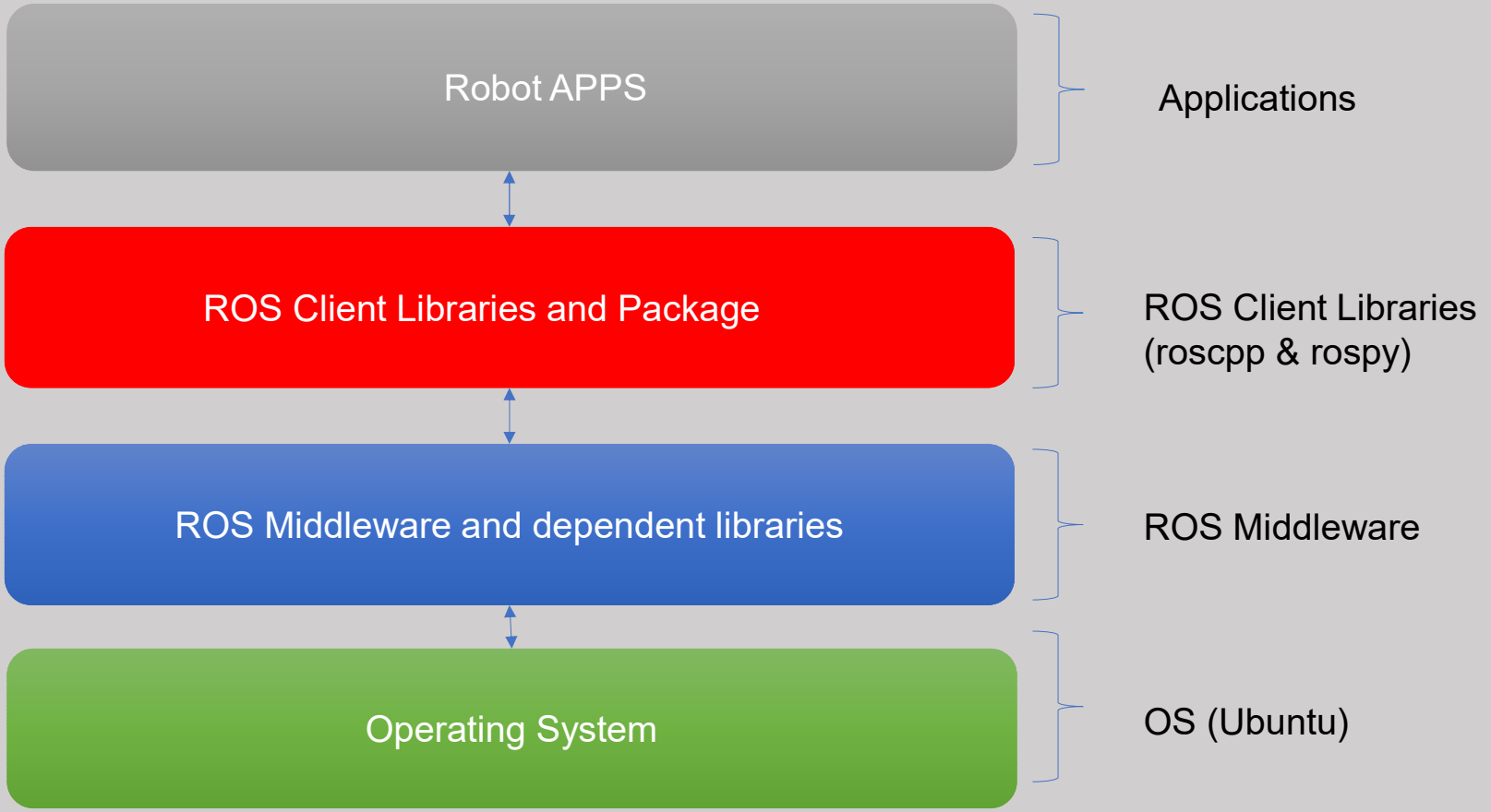
ROS Communication Model -Terminology

- **OSI Model: (**Open Systems Interconnection model)
  - https://en.wikipedia.org/wiki/OSI_model
- **TCP**: (Transmission Control Protocol)
  - https://en.wikipedia.org/wiki/Transmission_Control_Protocol
- **UDP**: (User Datagram Protocol):
  - https://en.wikipedia.org/wiki/User_Datagram_Protocol
- **VNET: (**Virtual Network**):**
  - https://en.wikipedia.org/wiki/Network_virtualization

What are ROS Client Libraries?

Discussing the basics of ROS client libraries

ROS Client Libraries



What are ROS client libraries?

- It is a software library used to develop ROS based applications or nodes.
- The various ROS concepts are already implemented in the client libraries.
- The client libraries are available for various computer languages.
- Popular ROS Client libraries: roscpp (for C++ developers), rospy (for Python developers)

What are ROS client libraries?

- It provides easy APIs to the developer to implement ROS concepts like ROS Topics, Service, Action etc.
- ROS Client libraries can be implemented in any programming languages
- http://wiki.ros.org/Client%20Libraries

Main ROS client libraries

- roscpp:
  - C++ client library for ROS
  - Used for high performance applications
  - http://wiki.ros.org/roscpp

- rospy:
  - Python client library for ROS
  - Favors development time over runtime performance
  - http://wiki.ros.org/roscpp

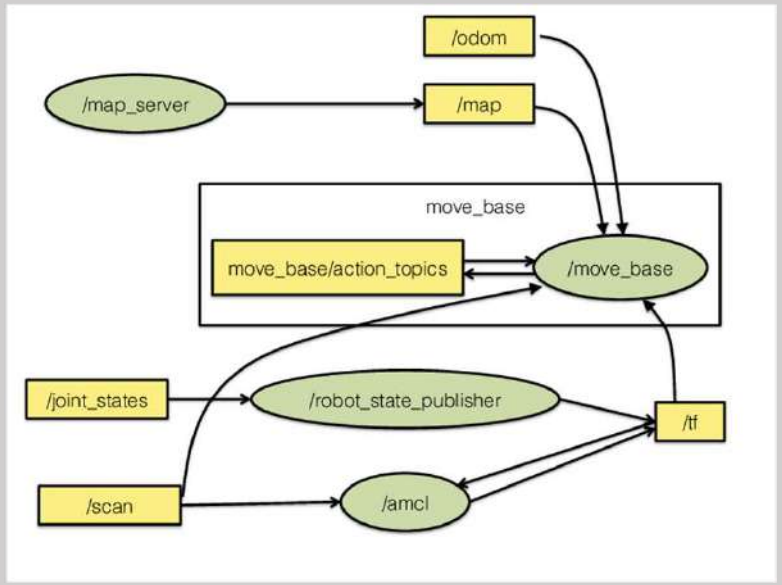
Experimental client libraries

- **roscs**: Client library for Mono/.NET

- **rosgo**: Implementation in Go programming
- **rosjava**: Pure java implementation with Android support

- **rosnodejs**: ROS Javascript client library for Node.js

- **roslua**: ROS clien library for Lua

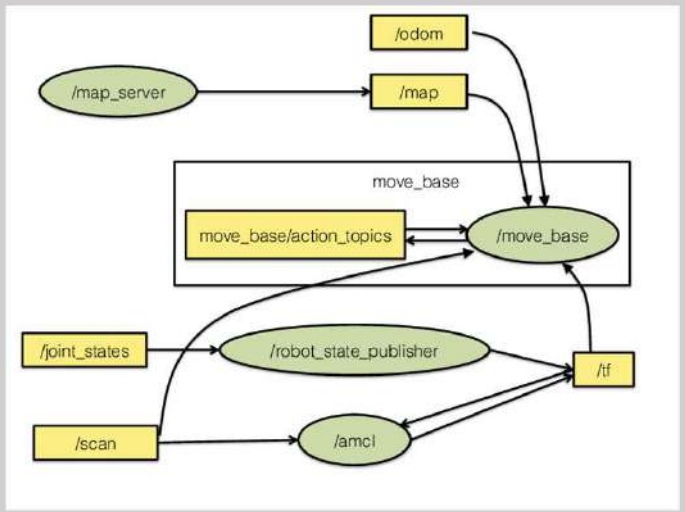


What are the different ROS Concepts?

Discussing the important ROS concepts

List of Topics

- ROS Computation Graph
- Important ROS terminologies
  - ROS Nodes
  - ROS Master
  - ROS Messages
  - ROS Topics
  - ROS Parameter Server
  - ROS Services
  - ROS Action
  - ROS Bags
- Secret of ROS inter-process communication

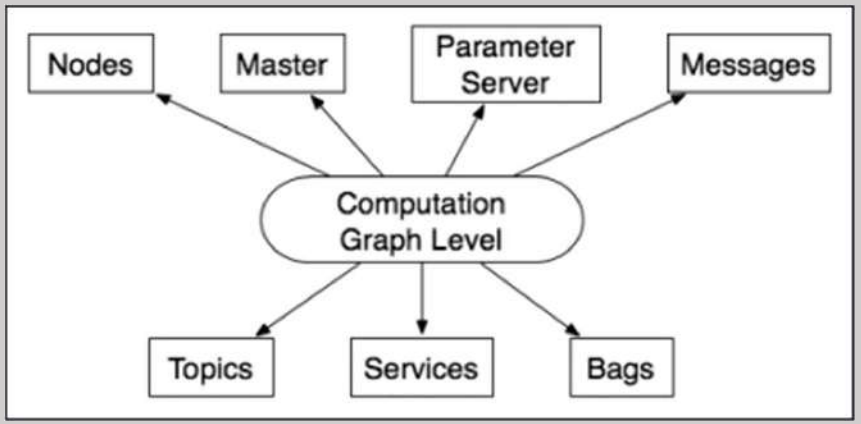


What is a ROS Computation Graph?

What is a ROS Computation Graph?

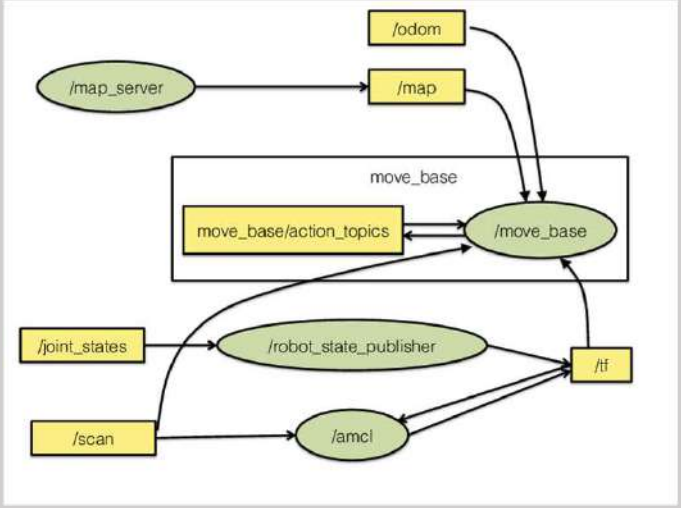
- **ROS Computation Graph:**
  - *It is the network of ROS process which is doing the computation in ROS together by connecting peer-to-peer*

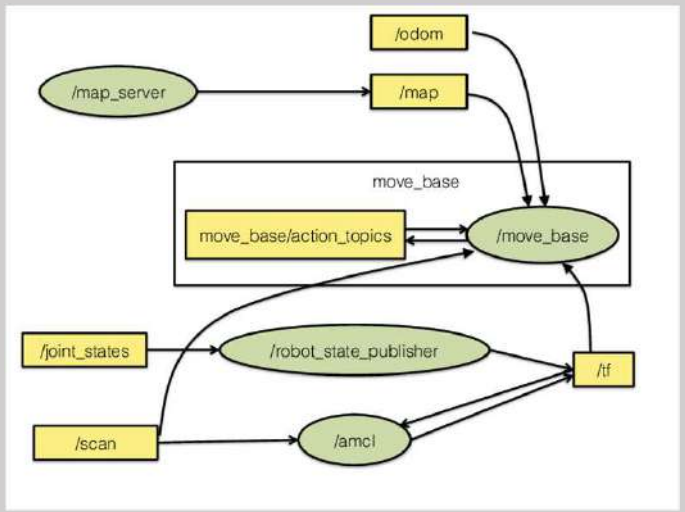
  - Various concept in the graph are
    - **Nodes**
    - **Master**
    - **Parameter Server**
    - **Messages**
    - **Topics**
    - **Services**
    - **Bags**



What is a ROS Computation Graph?

- **ROS Computation Graph:**
  - The ROS graph concepts are implemented in ***ros_comm*** repository
    http://wiki.ros.org/ros_comm
  - E.g.: ROS graph diagram



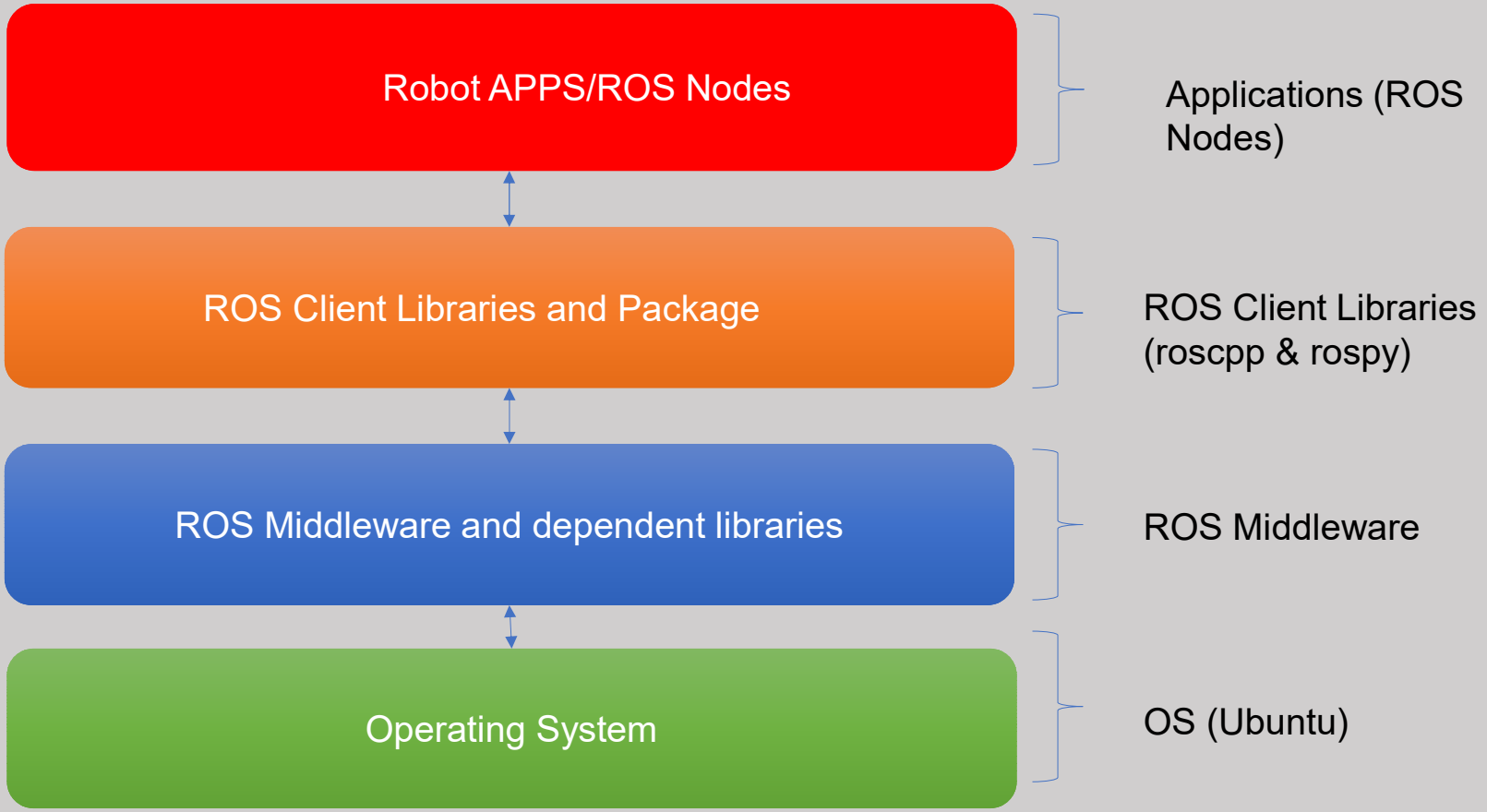


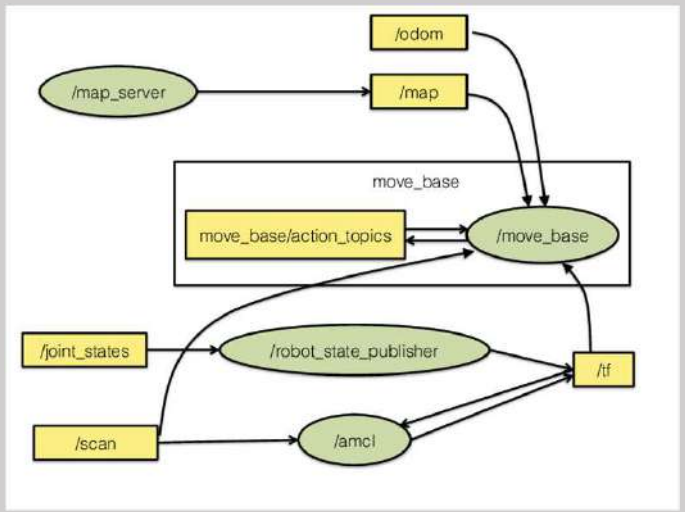
What is a ROS Node?

What is a ROS Node?

- A Node is a executable program which perform computation
- Nodes are ROS program created using ROS Client libraries like **roscpp** & **rospy**
- Nodes are the atomic computing unit of ROS framework
- The Nodes can communicate each other nodes using ROS concepts like Topics, services etc.

ROS Nodes

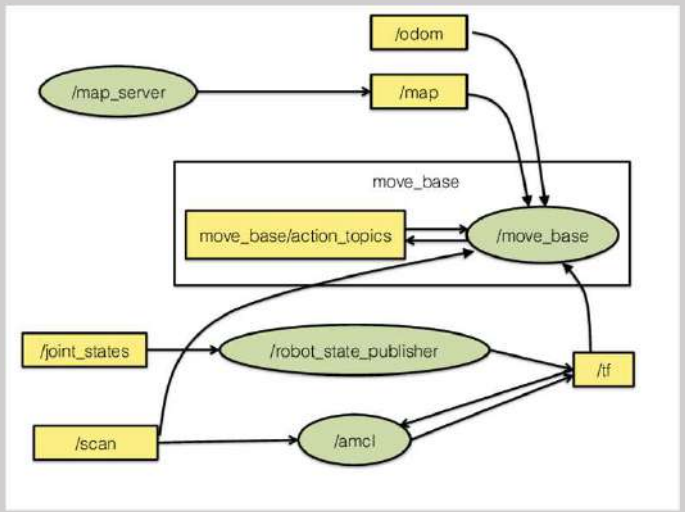

Robot APPS/ROS Nodes
ROS Client Libraries and Package
ROS Middleware and dependent libraries
Operating System
Applications (ROS Nodes)
ROS Client Libraries (roscpp & rospy)
ROS Middleware
OS (Ubuntu)



What is a ROS Master?

What is a ROS Master?

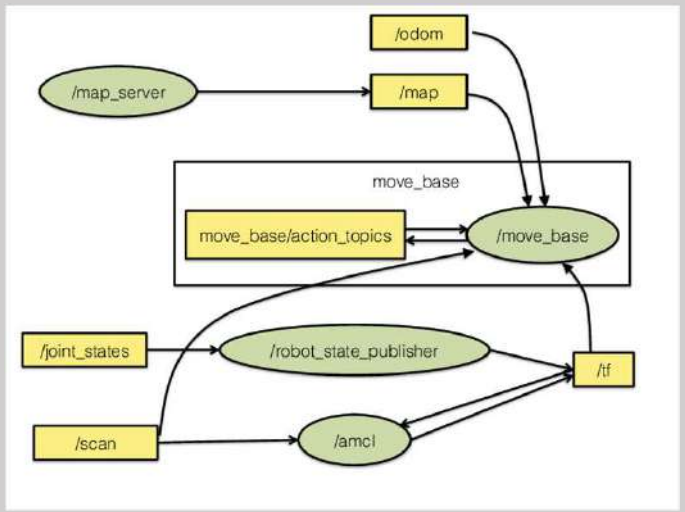
- **ROS Master:**
  - **Helps the ROS nodes to communicate and exchange messages**
  - Keep on tracking all ROS nodes and act as a bridge between two nodes to find each other
  - After the discovery of nodes, the two nodes will connect each other
  - The exchange of message between two nodes start after the connection.



What is a ROS Messages?

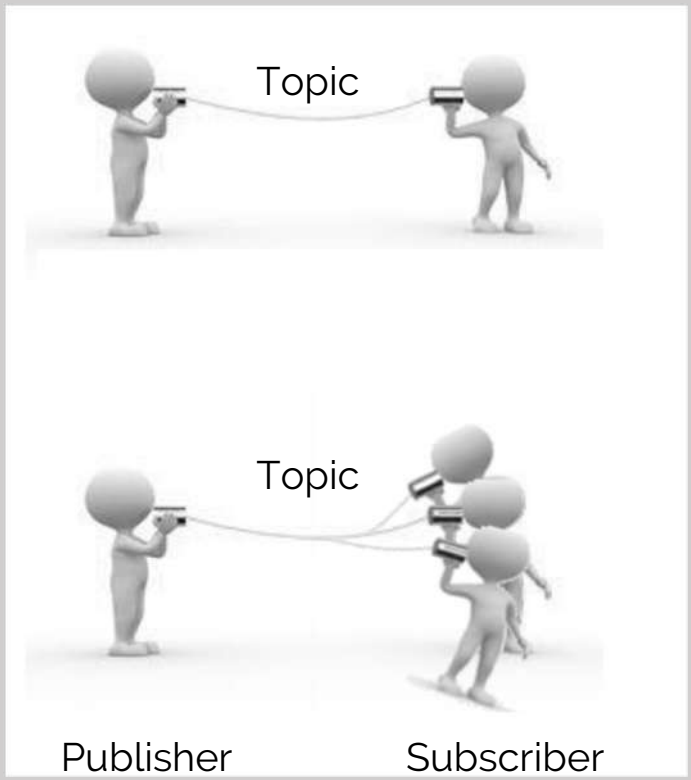
What is a ROS Message?

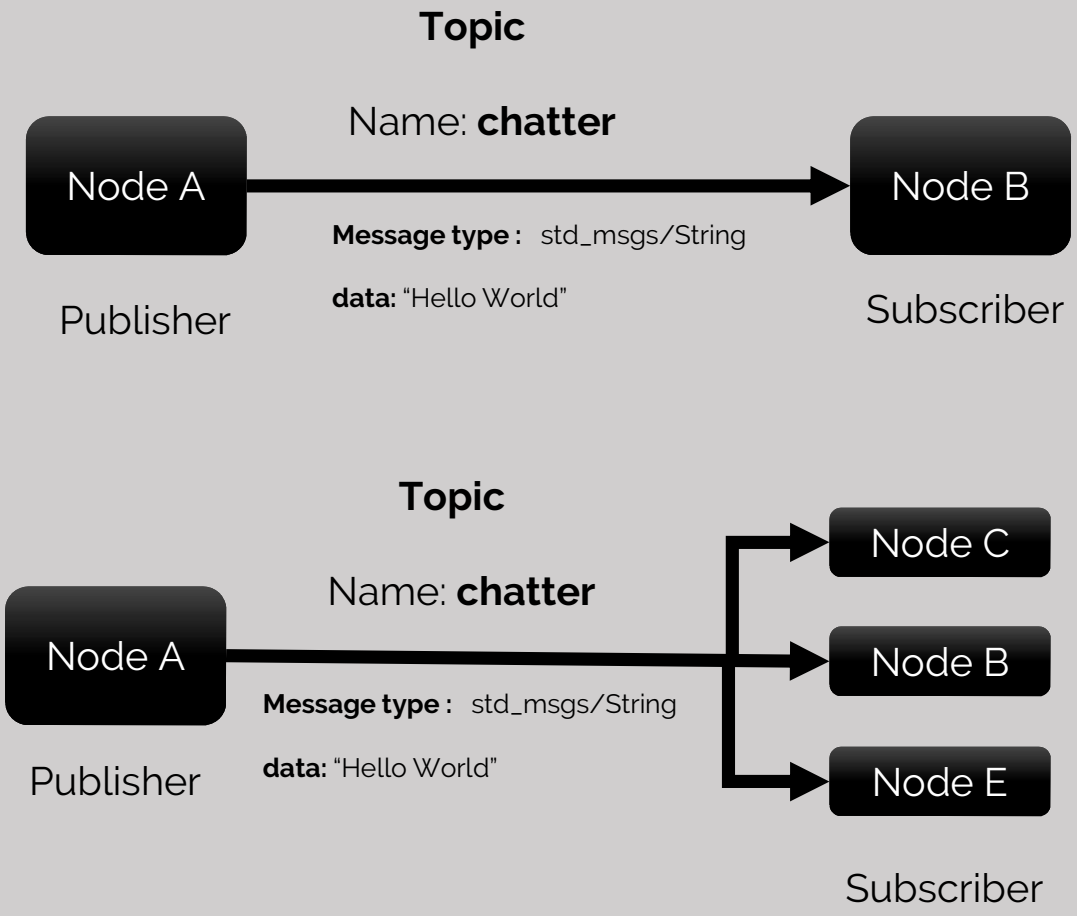
- **ROS Message:**
  - The ROS nodes are communicating with each other by passing ROS messages.
  - ROS message (**.msg**) is a data structure which can be of different types such as *Integer*, *Float*, *Boolean* etc.
  - It can hold same or different types of data type in a single message.
  - It can even hold an array of values of different data types.
  - We can use prebuilt messages in ROS or can create our own ROS messages
  - **E.g.: std_msgs/Int32, std_msgs/String, sensor_msgs/Image**
  - http://wiki.ros.org/Messages



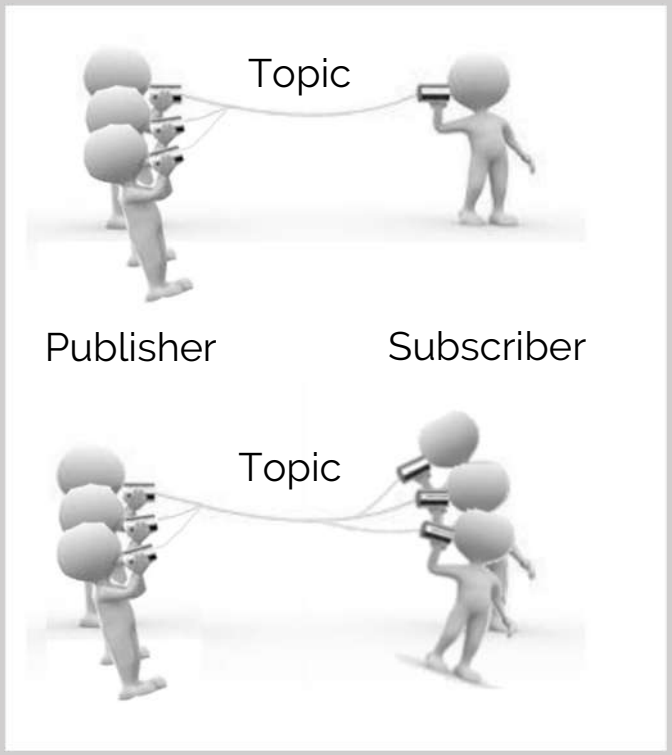
What is a ROS Topic?

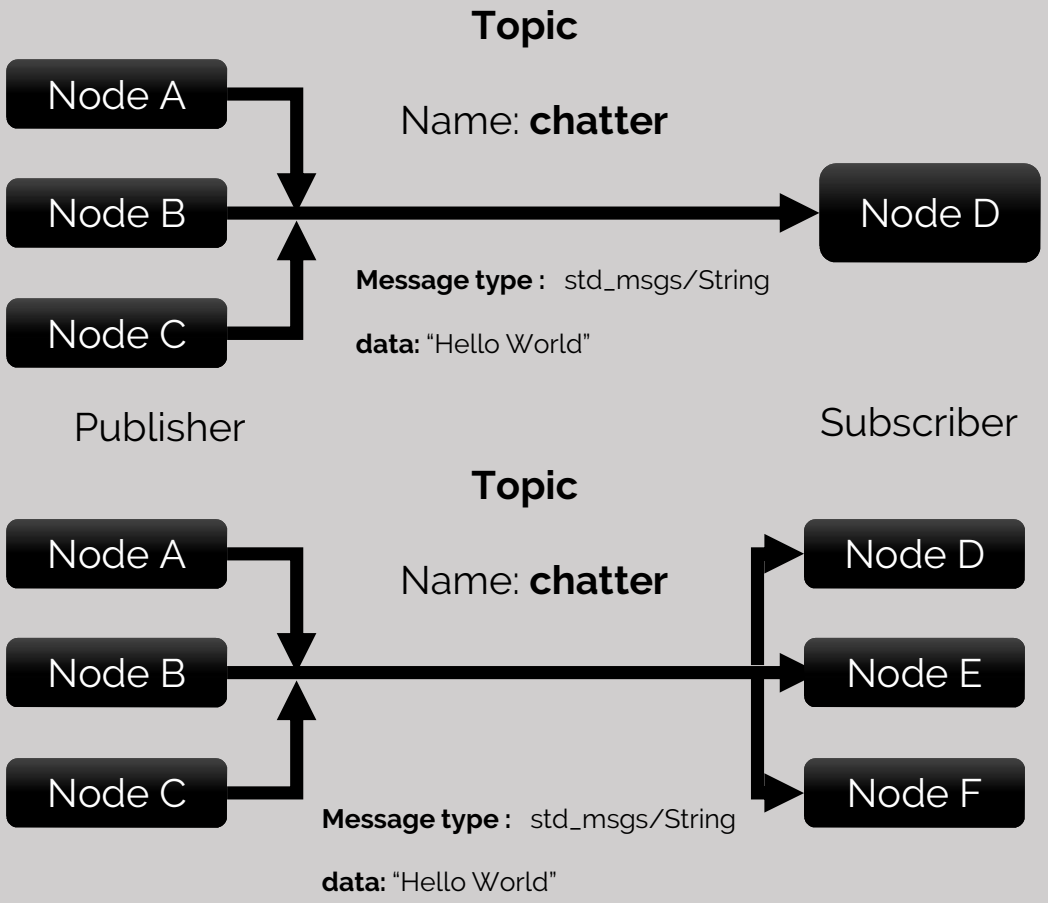
What is a ROS Topic?





What is a ROS Topic?



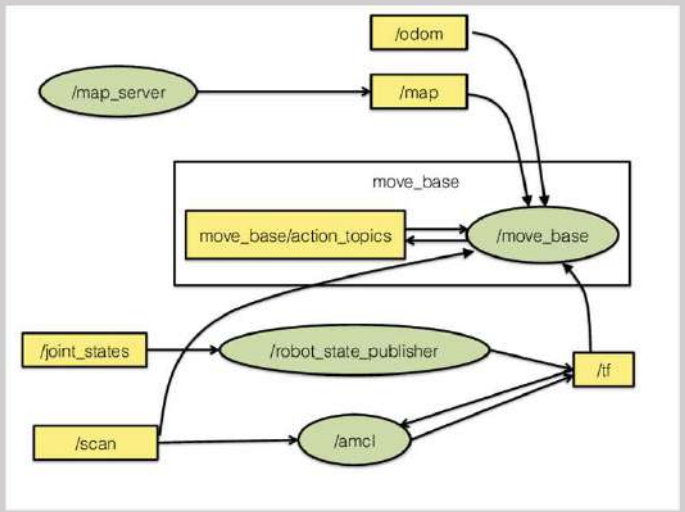


What is a ROS Topic?

- **ROS Topic:**
  - Topic is a named data bus in which ROS nodes send/receive ROS messages
  - Sending a message is called Publishing/Advertising
  - Receiving a message is called Subscribing
  - A ROS topic is identified by a name called Topic name
  - A topic can be created in a ROS node with a specific name and the type of ROS message it should transport

What is a ROS Topic?

- **ROS Topic:**
  - Once a topic is created, multiple nodes can send or receive the ROS message transporting through it
  - The *publishing* nodes and *subscribing* nodes can only exchange a message when they handle same message type, which is given during the creation of ROS Topic.
  - A ROS node can publish or subscribe any number of topics.
  - The publisher nodes and subscriber nodes are not aware of other's existence

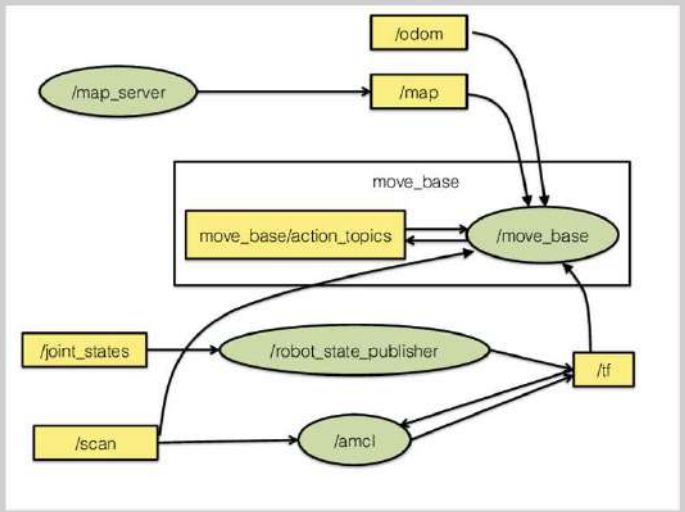


What is a ROS Parameter Server & Parameter?

What is a ROS Parameter Server?

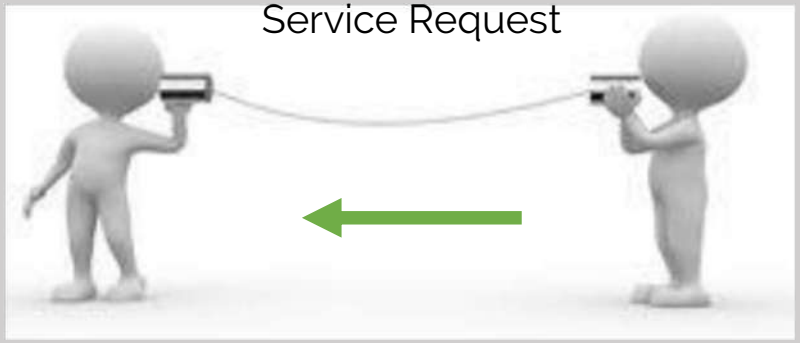
- **ROS Parameter Server:**
  - It is a server program running inside the ROS Master
  - It enables the ROS nodes to store and retrieve variables at runtime
  - These variables are called ***ROS Parameters***.
  - The parameters can be accessed by all nodes in the ROS network
  - We can create private and public ROS parameters
  - It is implemented using XML RPC libraries

  - Parameter types: 32-bit integers, Booleans, strings, doubles, lists etc.
  - E.g. **/P: 10.0**
    /I: 1.0
    /D: 0.1



What is a ROS Service?

What is a ROS Service?



Hey Server, What is 2+2

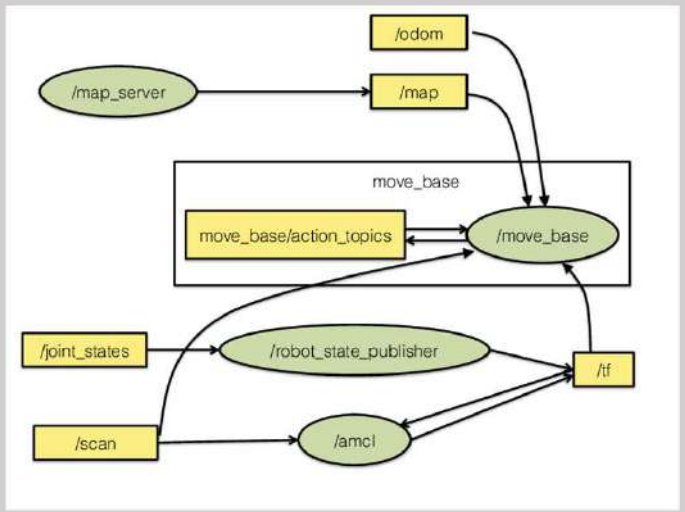
Thanks for contacting me, The answer is 4

What is a ROS Service?

- **ROS Service:**
  - The ROS topics are sending message in many-to-many one direction using publish/subscribe mechanism
  - We may need a *two way* communication like **request/reply** in a distributed system.
  - Using ROS services, we can implement a **request/reply** system in our nodes
  - Node which sending the request is called ***Service client node***.
  - Node which receive the request and execute the service is called ***ROS Service server.***
  - After the execution, the server node send the results back to the ROS client.

What is a ROS Service?

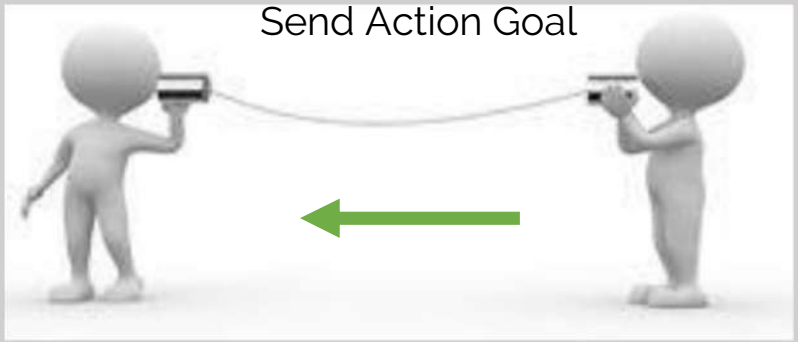
- **ROS Service:**
  - The node is sending request/reply using a *ROS service message*.
  - The service message (**.srv**) will have the definition of ***request*** and ***reply*** datatype
  - The service message type is similar to ROS messages but have different fields for request and reply.
  - E.g.: **std_srvs/SetBool**
  - http://wiki.ros.org/rosservice



What is a ROS Action?

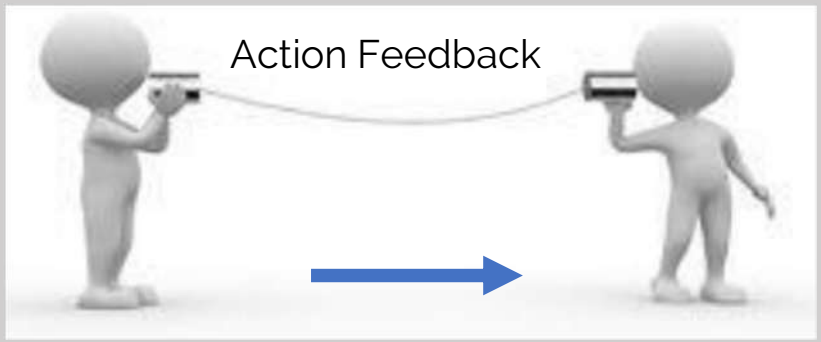
What is a ROS Action?

1)



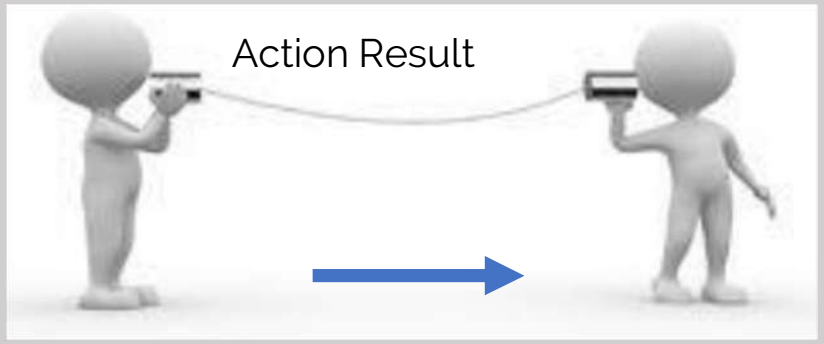
Server Client

2)



Server Client

3)



Server Client

What is a ROS Action?

- **ROS Action:**
  - Similar to ROS Service
  - There is a Action Server and Client
  - In ROS service, we can't cancel the service once it requested
  - Using ROS action, we can cancel the current service request
  - The action server can also send feedback of the current goal execution.
  - After completing the goal, the action server will send the final result to the client

  - **ROS Action = ROS service + feedback + state + status + cancel goal**

What is a ROS Action?

- **ROS Action:**
  - Similar to ROS service, there is ROS Action messages
  - The ROS Action message (**.action**) contain
    - **Goal message type**
    - **Feedback message type**
    - **Result message type**

What is a ROS Action?