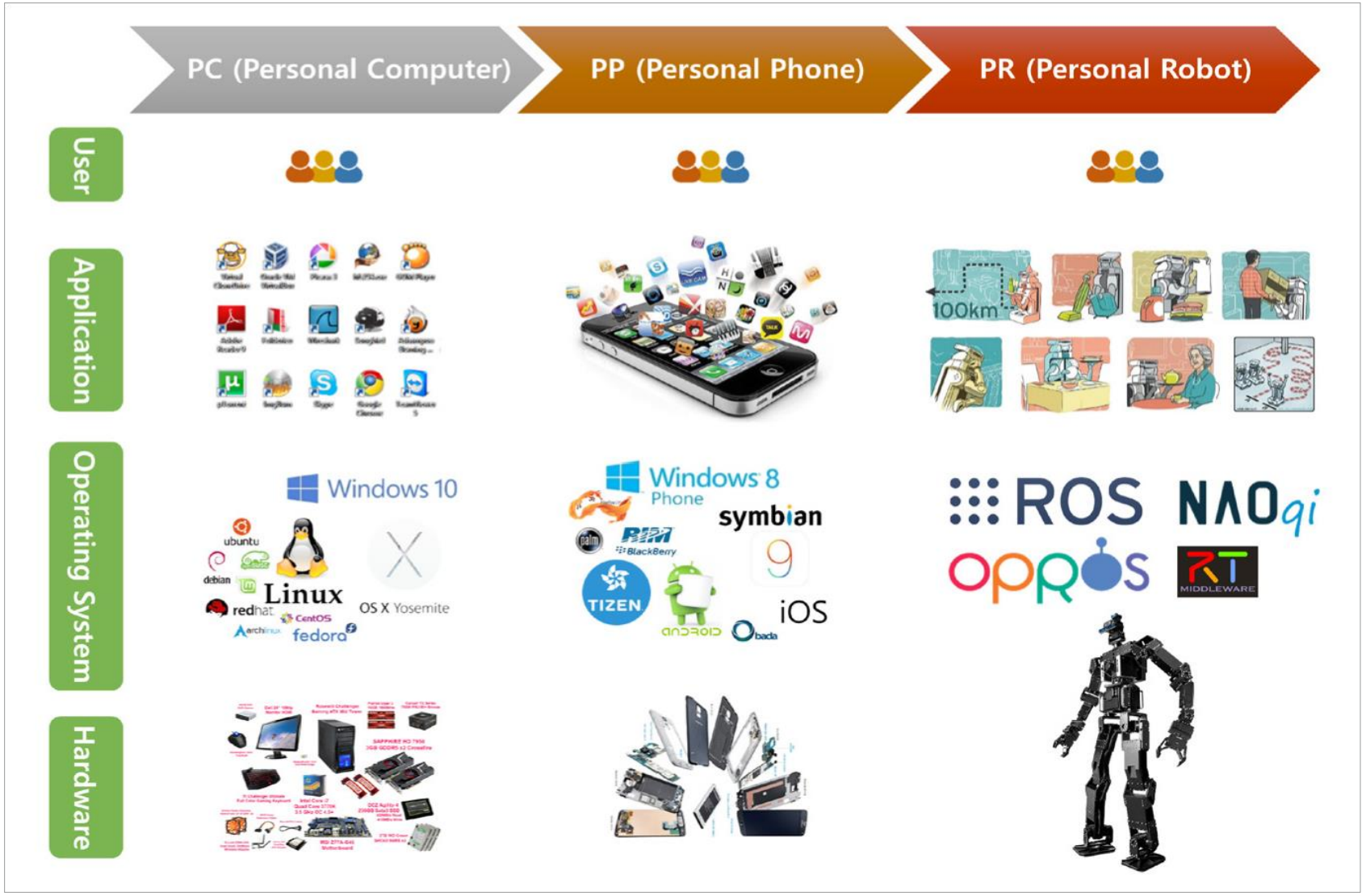
ROS

Robot Operating System

Robot Operating System



Components of the Software System Platform

ROS?

- ROS or Robot Operating System.
- Framework for robotic software development providing Operating System Like functionality, including hardware abstraction, low-level-device control, message-passing between processes, and package management.
- The origins lie in Stanford Artificial Intelligence Lab and was further developed at Willow Garage. Willow Garage is a Robot research lab.

- Available for all major operating systems

ROS?

- ROS is an open source, free for users.
- Specifically, it is a software framework rather than an operating system
- It provides the services you would expect from an operating system including hardware abstraction, low level device control, implementation of commonly used functionality, message passing between processes and package management.
- It also provides tools and libraries for obtaining, building, writing and running code across multiple computers.

ROS

- It is not a traditional operating system like Windows, Linux and Android. Rather ROS uses the traditional operating system (Linux, Windows, Android)

- It uses the existing operating system's process management system, file system, user interface, program utilities (compiler, thread, model etc.). In addition, it provides essential functions for developing robot application software such as data transmission/reception, scheduling and error handling among many different types of hardware in a library form.

- In addition, it develops, manages and provides various application programs based on the robot software framework and distributed packages developed by users.

ROS: Software Framework

- Software framework for developing robot software

❑ It is possible to jointly develop complex programs by finely dividing them with message exchanging method between nodes.

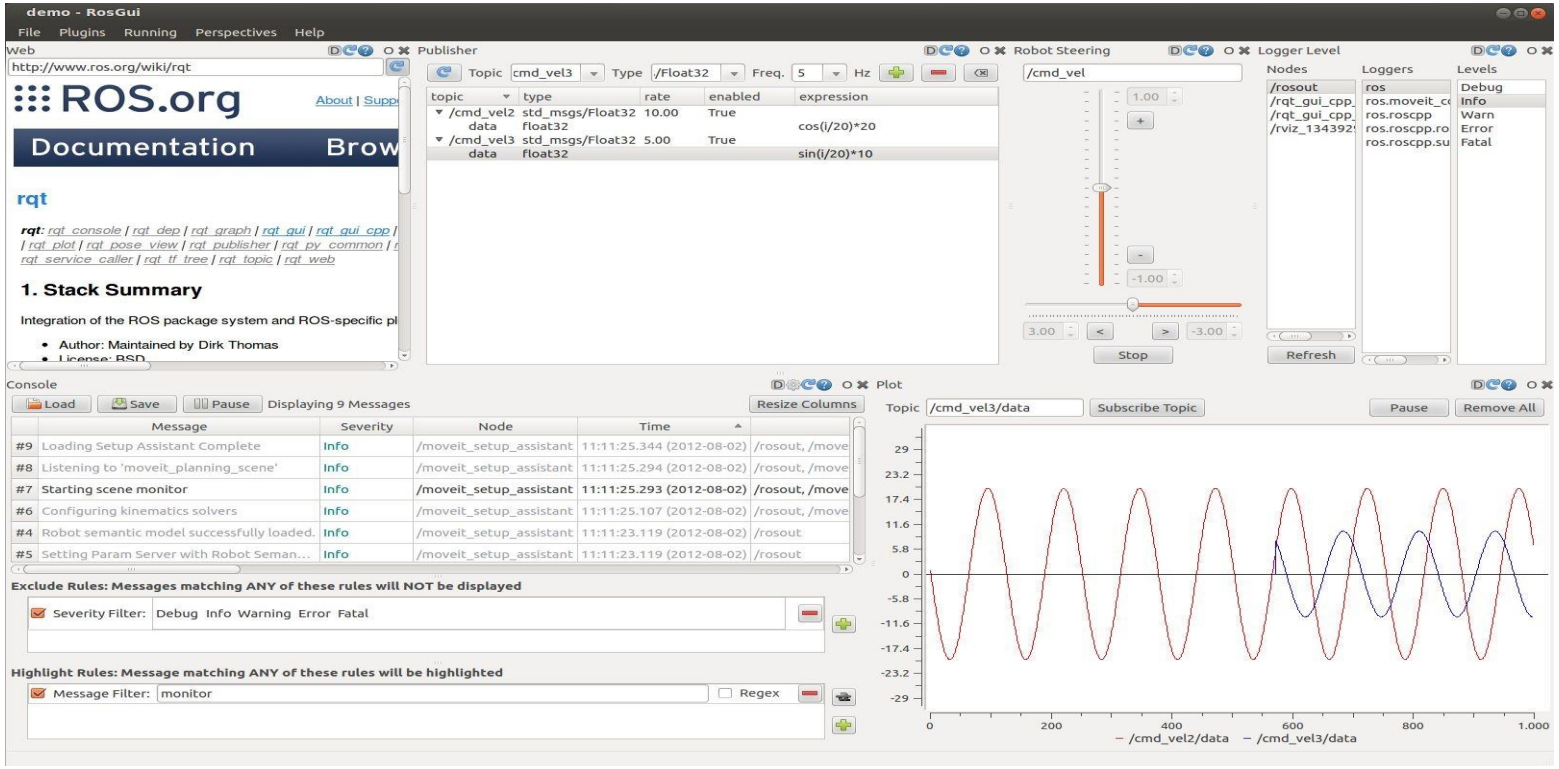
❑Supports command tools, Visualization tool Rviz, GUI toolbar rqt, 3D simulator Gazebo

❑Supports modeling, sensing, recognition, navigation and manipulation functions commonly used in robotics

ROS Add Ons

rqt

- **rqt** is a software framework of ROS that implements the various **GUI tools** in the form of plugins. One can run all the existing **GUI tools** as dockable windows within **rqt**! The **tools** can still run in a traditional standalone method, but **rqt** makes it easier to manage all the various windows on the screen at one moment. A QT based GUI developed for ROS



rqt

- A QT based GUI developed for ROS
- Lots of different plugins.

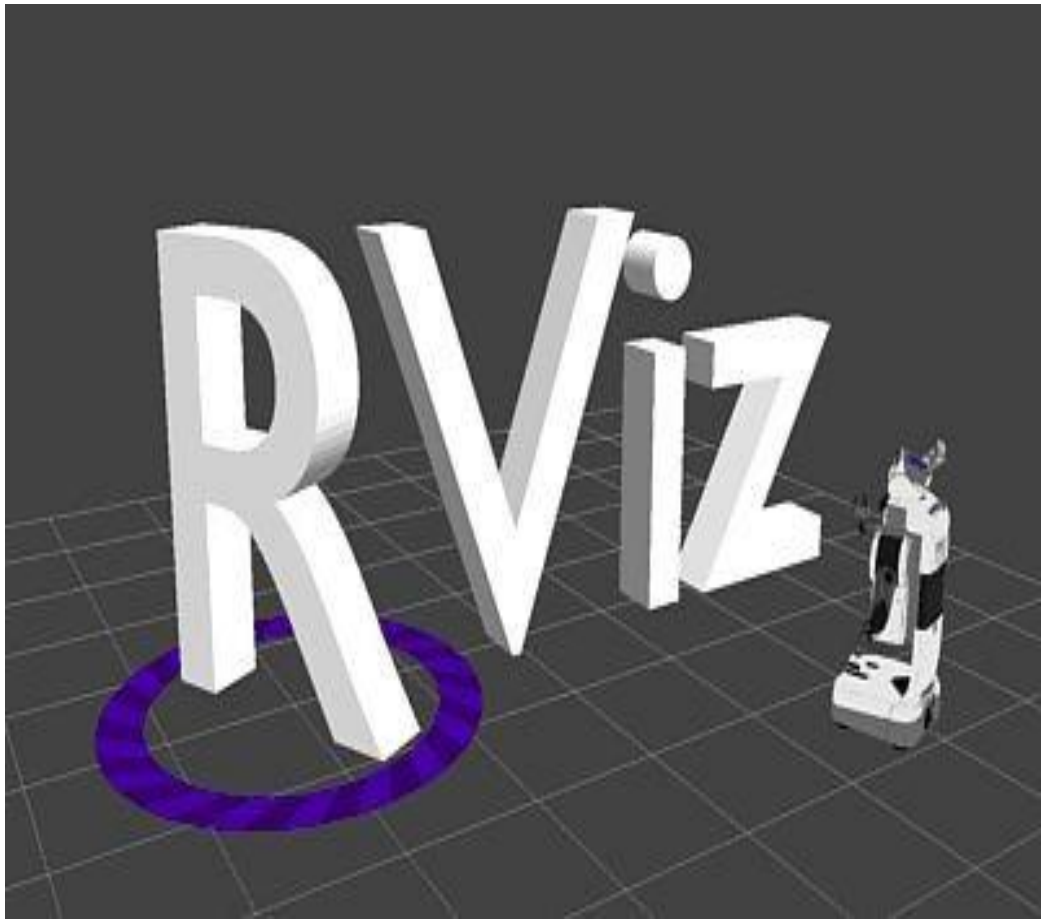
❑ `rqt_graph`, `rqt_image_view`, `rqt_console` to name a few.

❑`rqt_graph` can be used to view the graph structure of the system, i.e the nodes, and how are they related etc.

- Multiple plugins can be run simultaneously.
- Anyone can add more custom plugins written in C++ or Python.

RVIZ : ROS

- A Powerful tool for 3D Visualization in ROS
- Modular state and sensor visualization
- Excellent community support.



Gazebo Simulator

- Simulate 3D rigid body dynamics
- Simulate a variety of different sensors, many of them are available online ready to use.
- Has many environments and robots pre-implemented.
- With ROS interface, it can be used to directly to test the applications inside a simulation.
- Has even more plugins available.

ROS-enabled operating system

- **Traditional operating system**

❑Ubuntu, Windows, Fedora, OS X, Gentoo, OpenSUSE, Debian, Raspbian, Arch, QNX Realtime OS. (There can be functional limitations in some OS)

❑Partially available for Android and iOS, smartphone operating system

❑In case of microcontroller unit (MCU) which can not be equipped with OS, it provides a library to communicate via serial communication, Bluetooth and LAN.

❑Basically it is recommended to run on Ubuntu

Philosophy of ROS

- **Peer to Peer:** Individual programs (nodes) communicate over ROS API (messages, etc.)
- **Distributed:** Nodes can communicate over a network.
- **Multilingual:** Native support for C++, Python and Lisp, Experimental support exist for Java and Lua too. Client Libraries for MATLAB etc.
- **Light Weight:** Doesn't slow the programs or even hinder their ability to work with other frameworks.
- **Free and Open Source**

What ROS isn’t?

- An actual Operating System
- A programming Language
- A programming environment/IDE
- A hardware

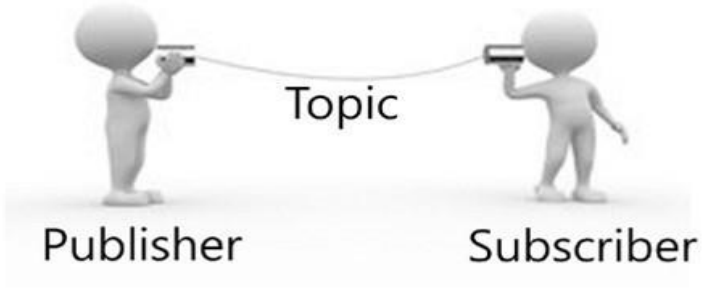
ROS terms

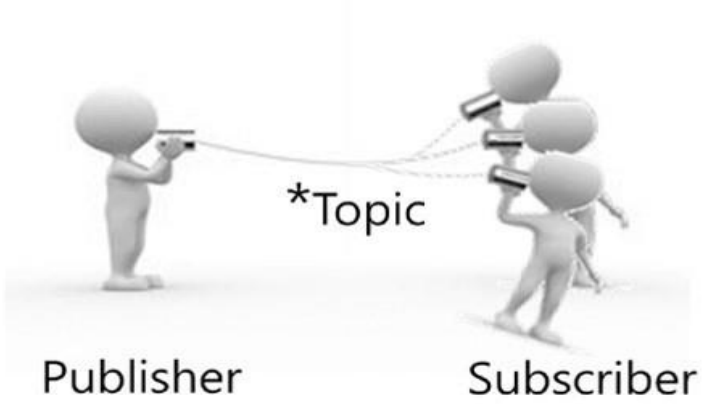
- Node: The smallest unit of executable processors i.e. sensors, actuators etc. It can be regarded as single executable program. In ROS, a system is consist of many nodes. Each node transmits and receives data by message communication

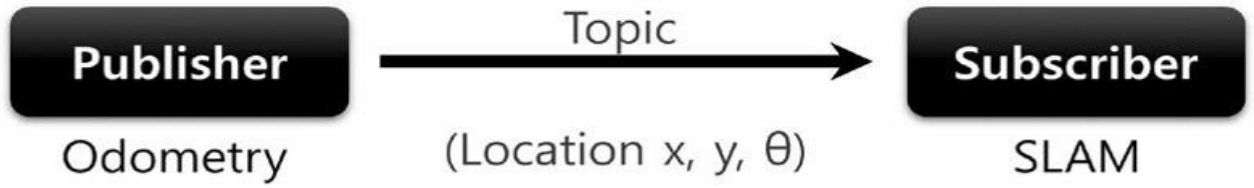
- Package: One or more nodes, information for node execution . Also bundles of packages are called as metapackages.

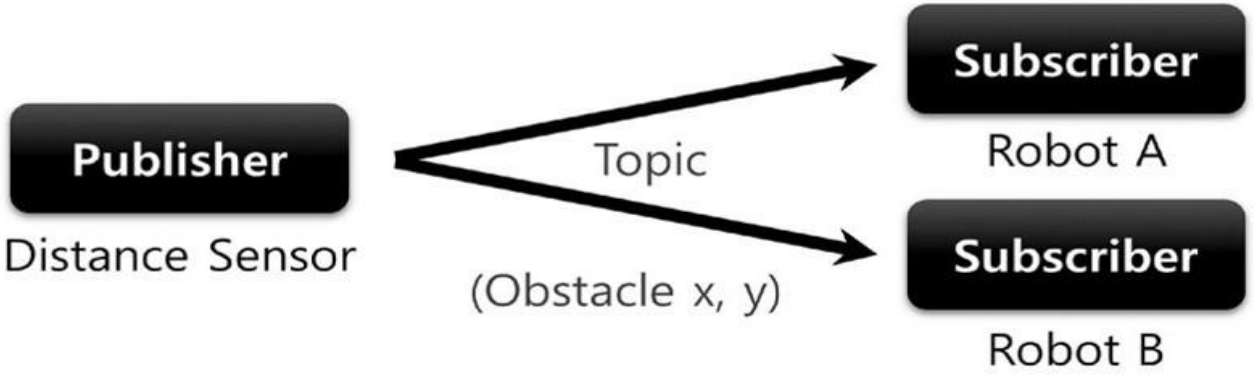
- Message: Data is transmitted and received through message between nodes. Messages can have various types such as integer, floating point and Boolean. You can also use structures such as a simple data structure and an array of messages that hold messages in the message.

ROS terms Topic, Publisher, Subscriber

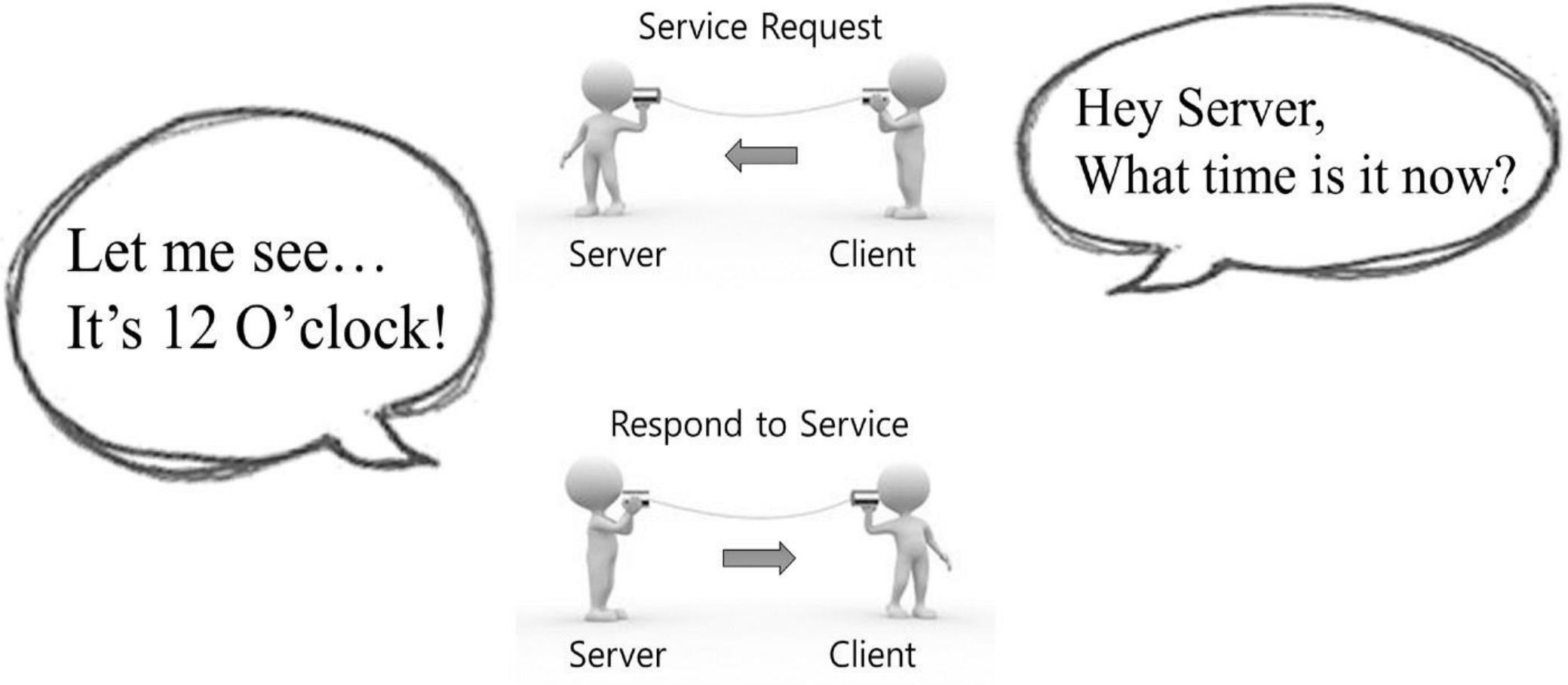




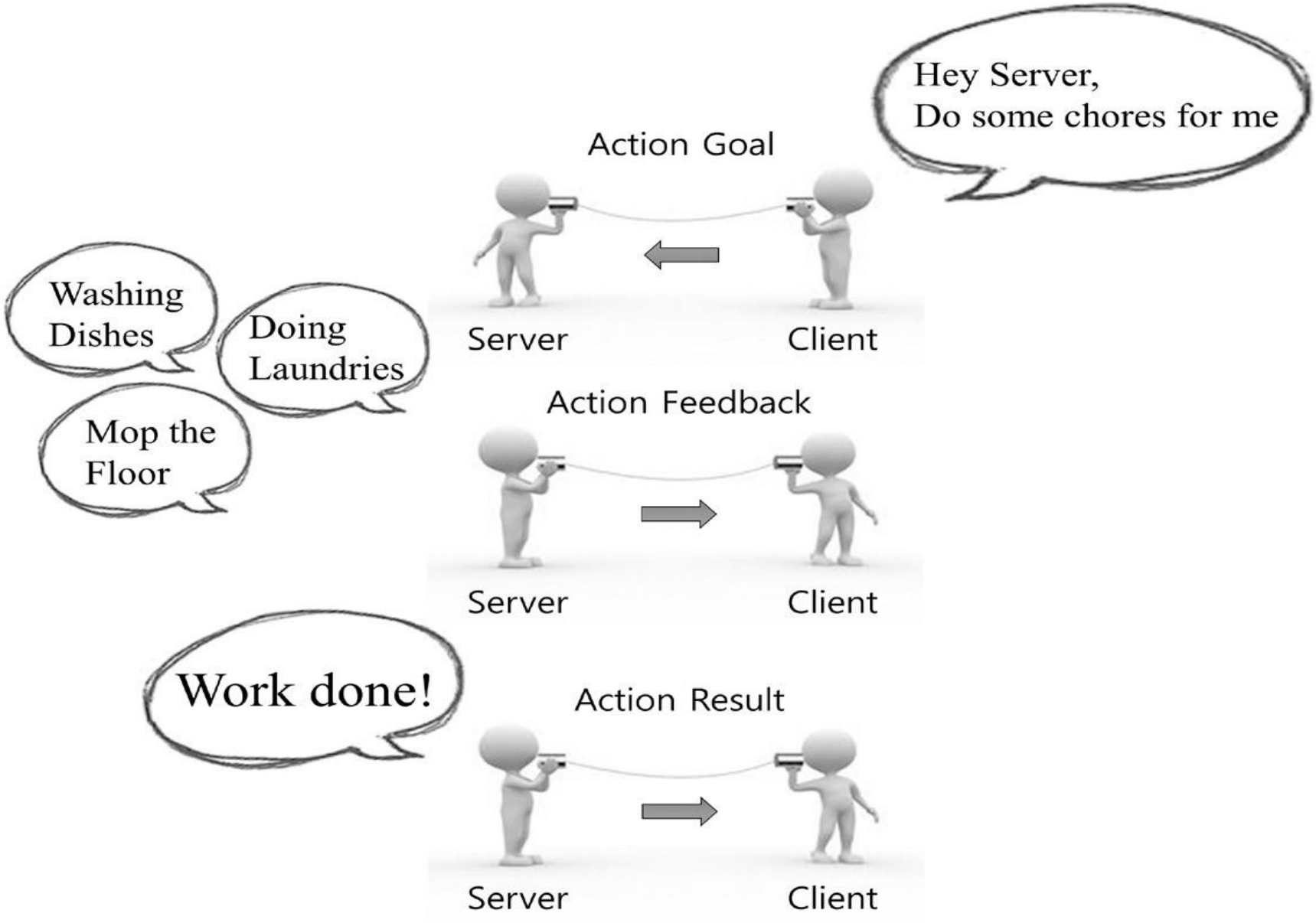




ROS terms Service, Service server, Service Client



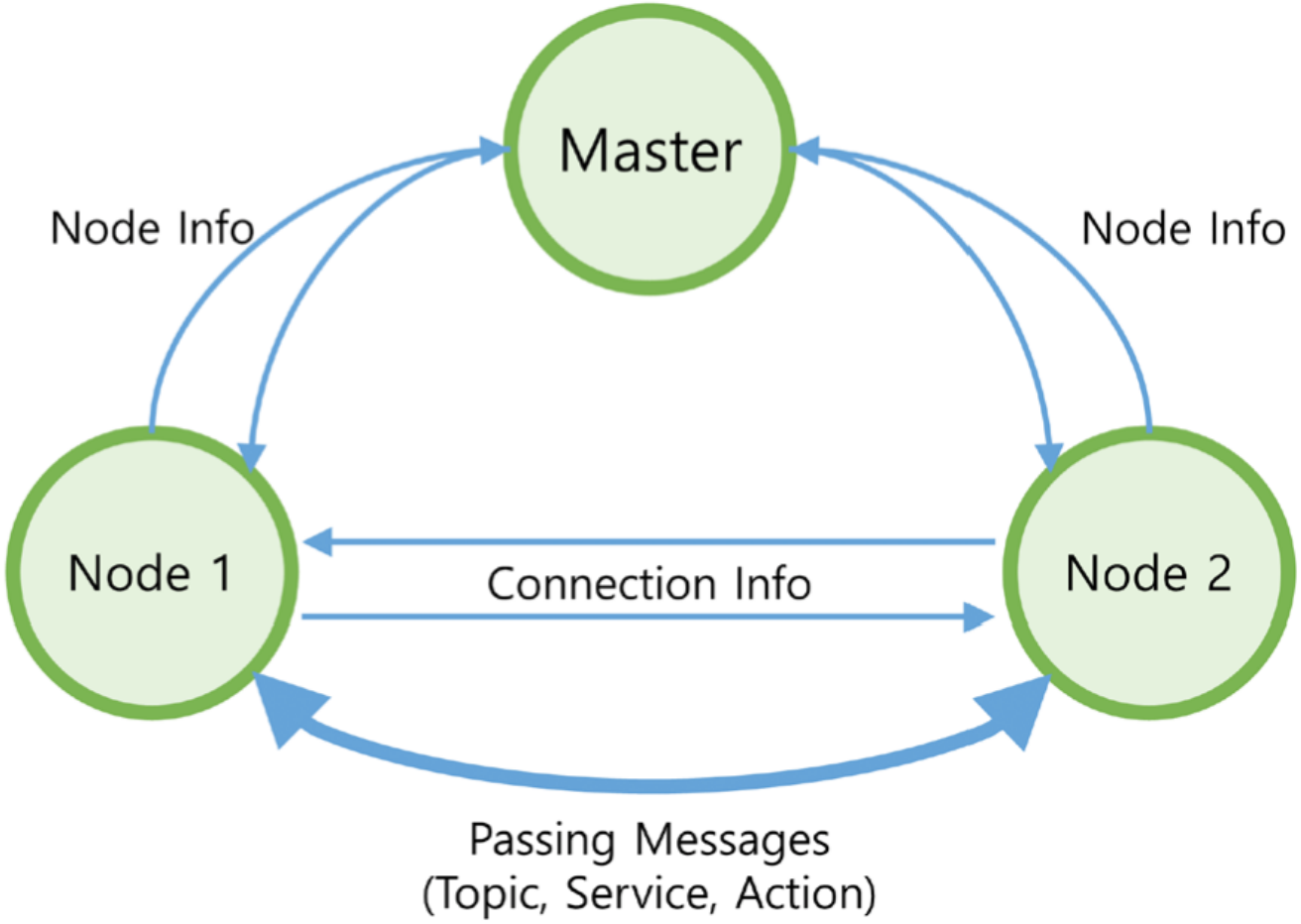
ROS terms Action, Action server, Action Client



Message Communication

Understanding Message Communication

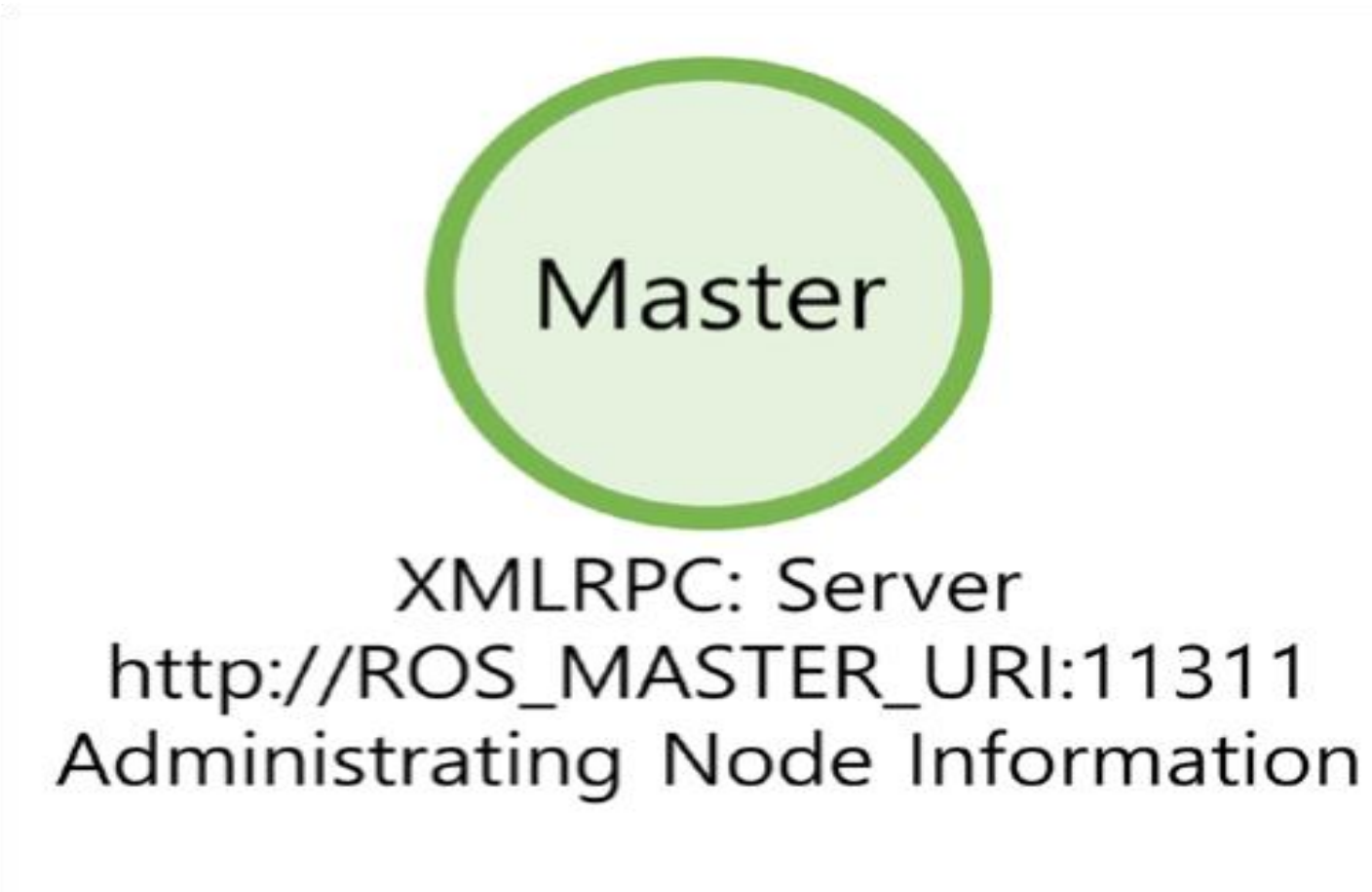
- The most fundamental technical point of ROS: message communication among nodes



Understanding Message Communication

1. Run Master: XMLRPC(XML-Remote Procedure Call)

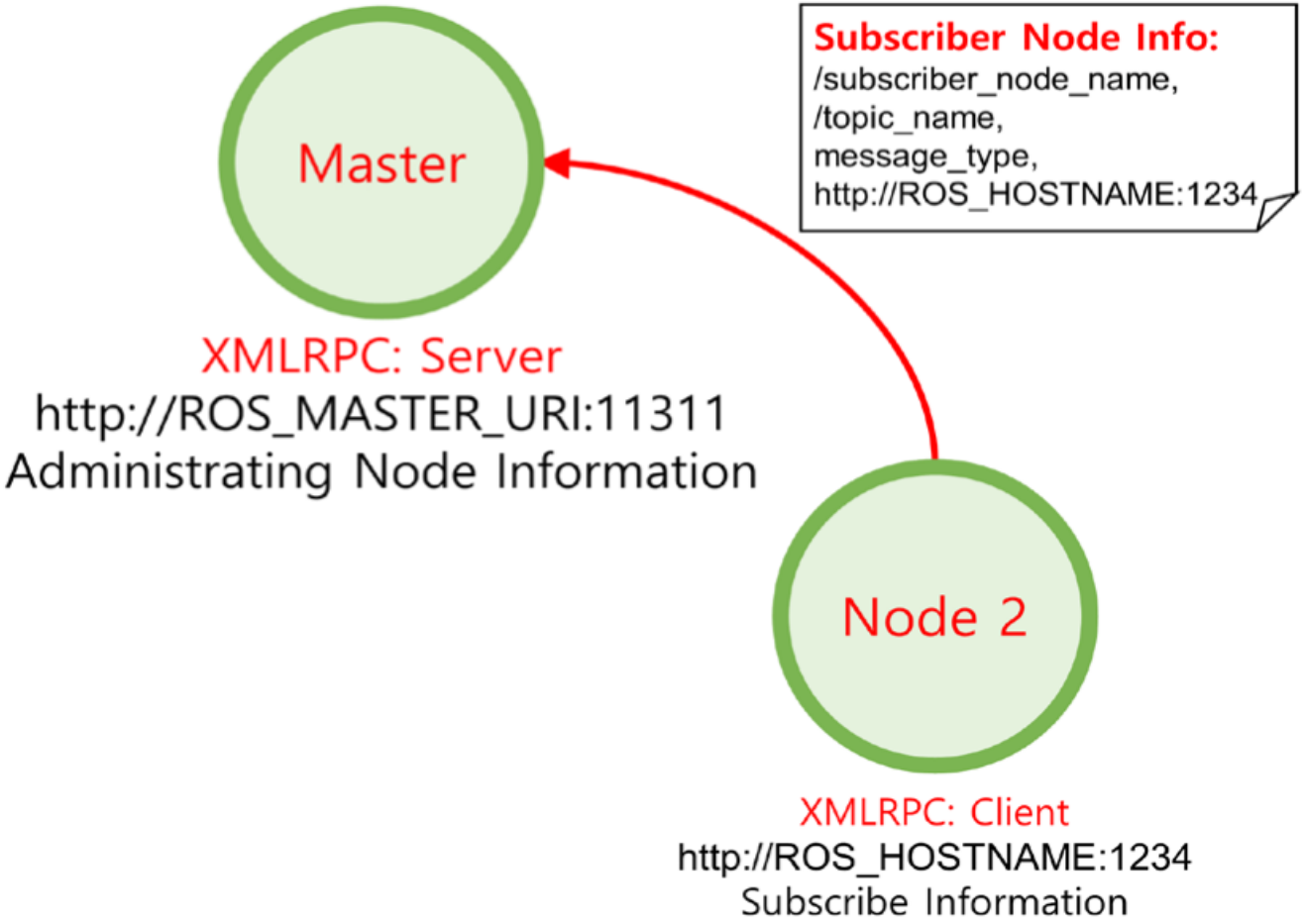
- $ roscore



Understanding Message Communication

2. Run Subscriber node

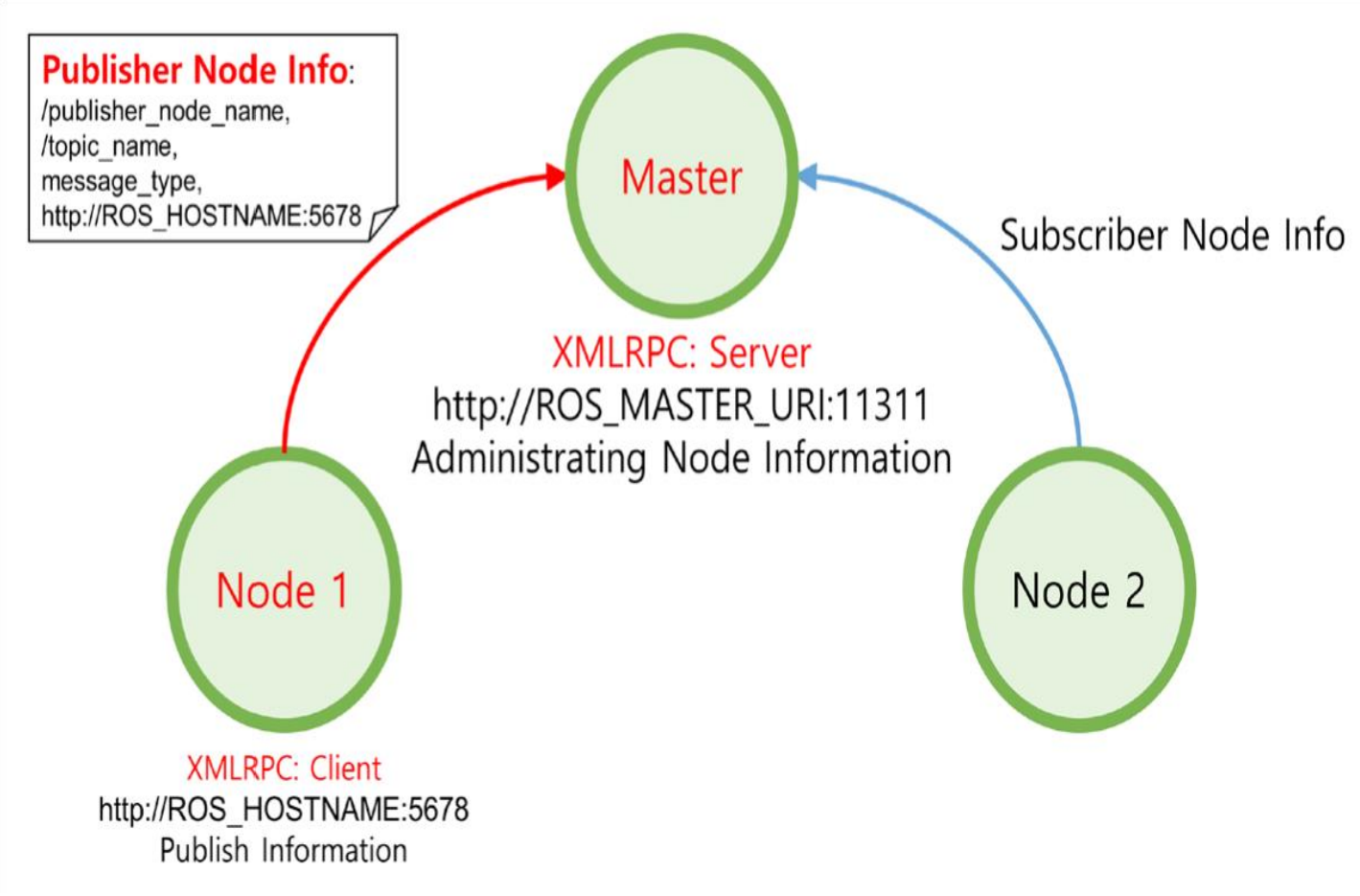
- $rosrun packagename nodename



Understanding Message Communication

3. Run Publisher node

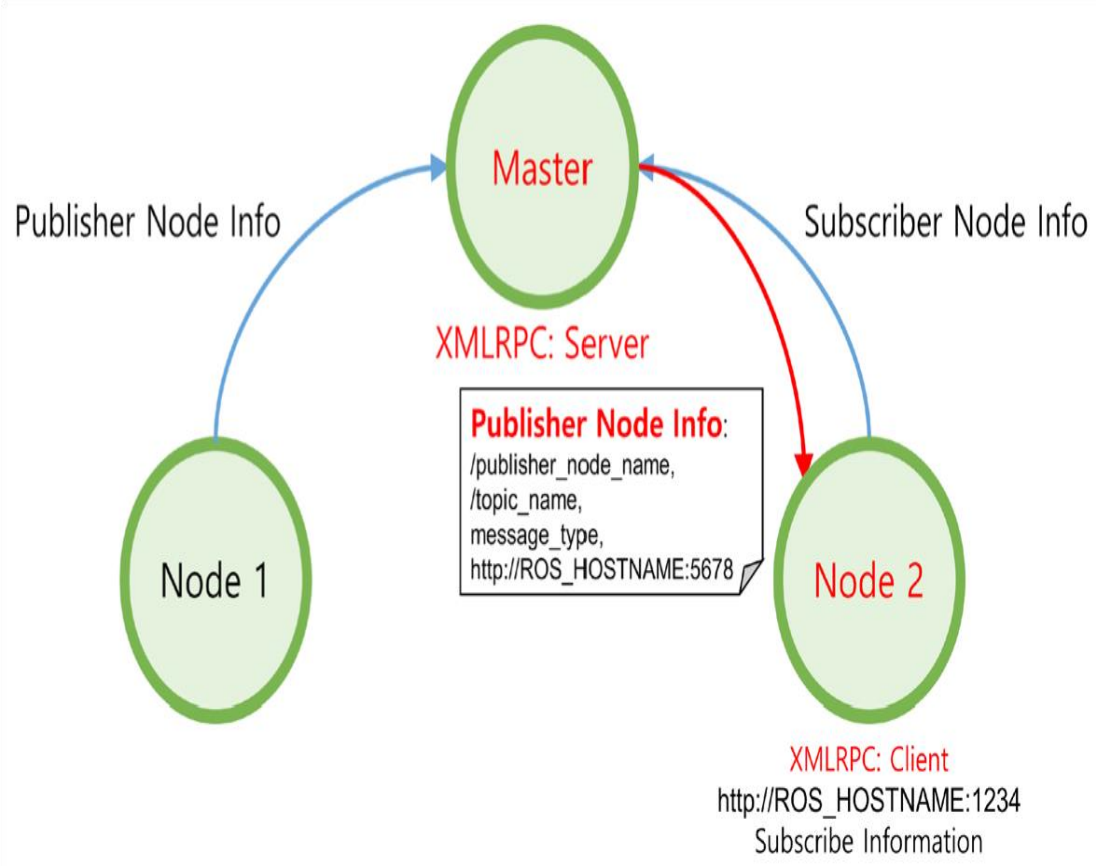
- $rosrun packagename nodename



Understanding Message Communication

4. Publisher Information

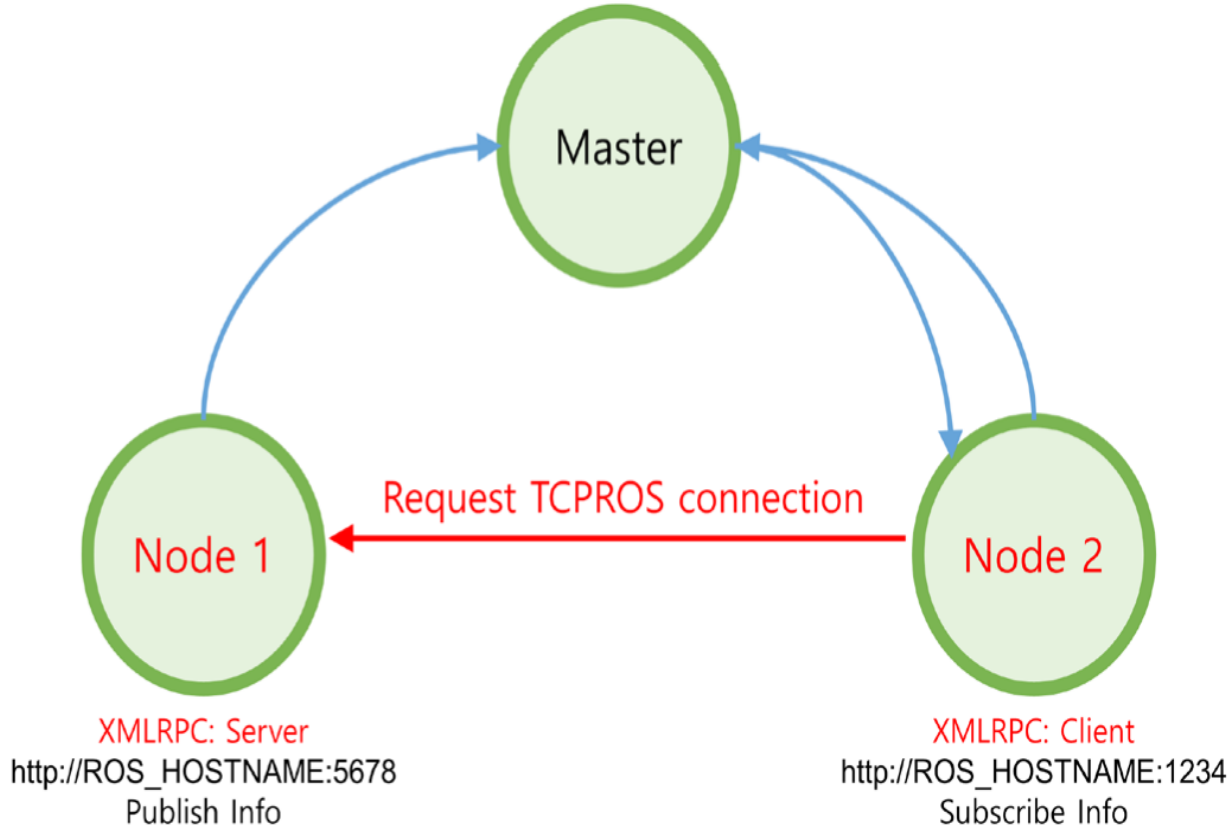
- The master informs the subscriber node of the new publisher information



Understanding Message Communication

5. Request access to the publisher node

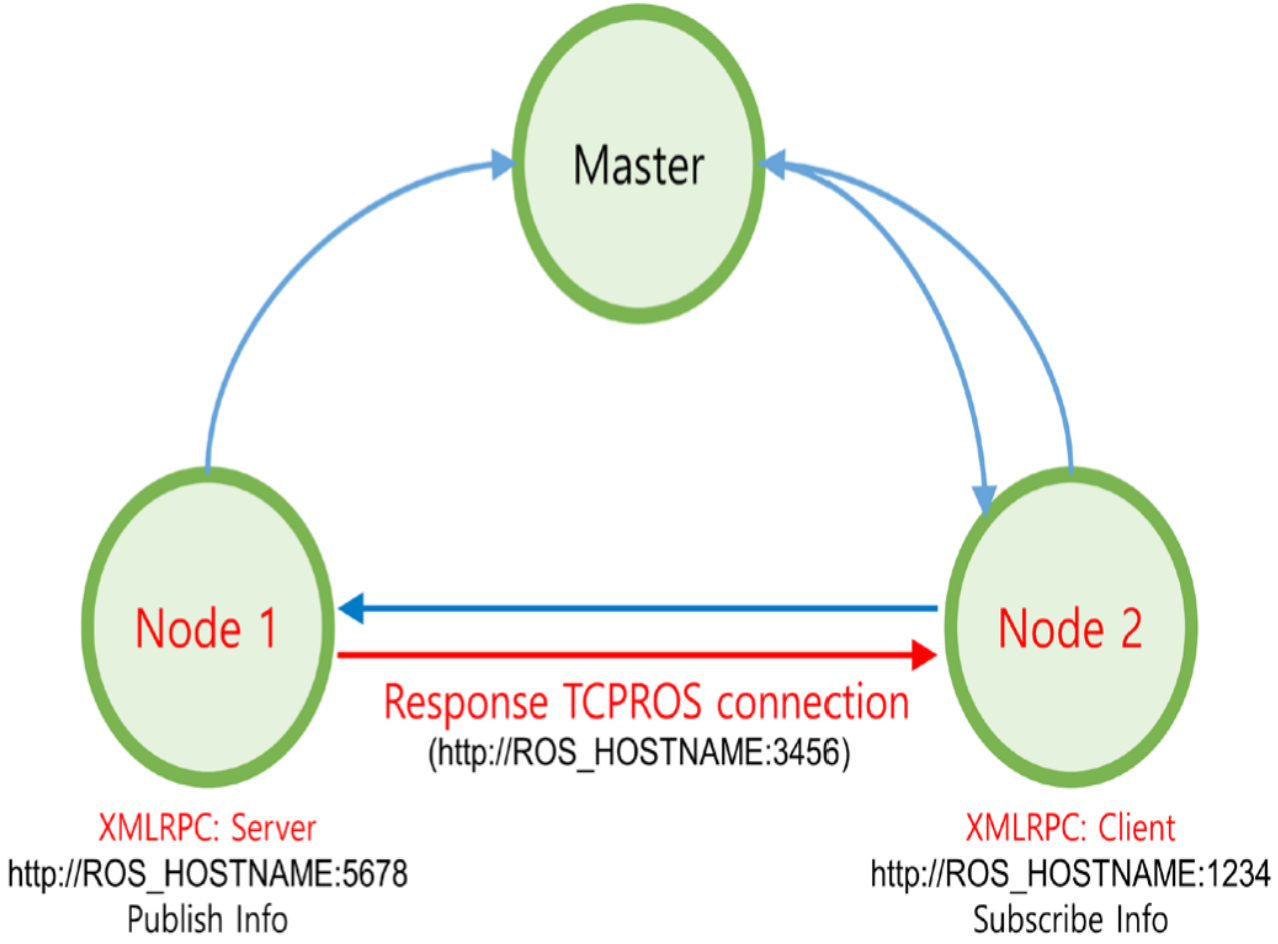
- Request TCPROS connection using the publisher information from the master



Understanding Message Communication

6. Connection response to subscriber node

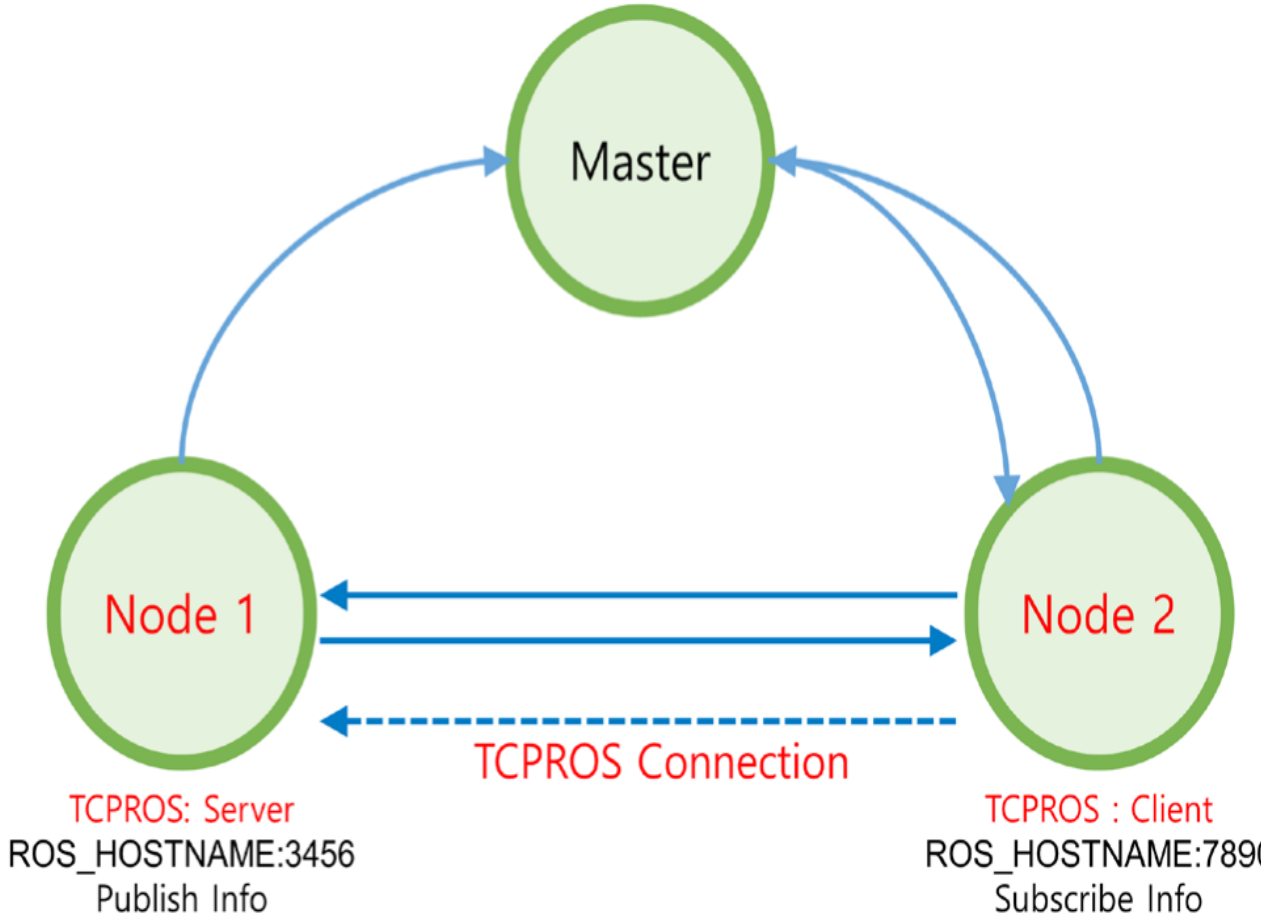
- Return TCP URL and port number corresponding to the connection response



Understanding Message Communication

7. TCP Connection

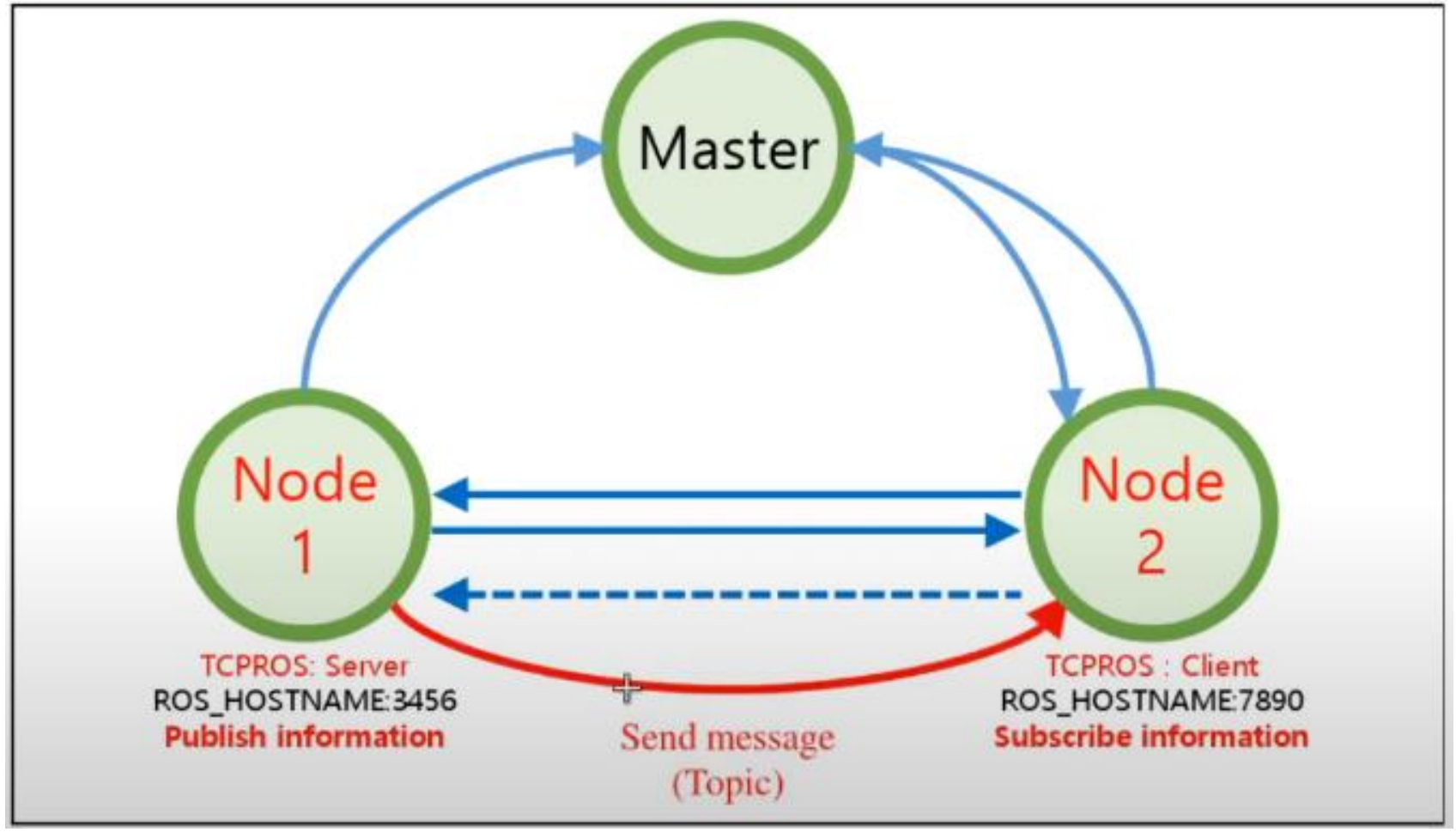
- Establish connection with the publisher node using TCPROS



Understanding Message Communication

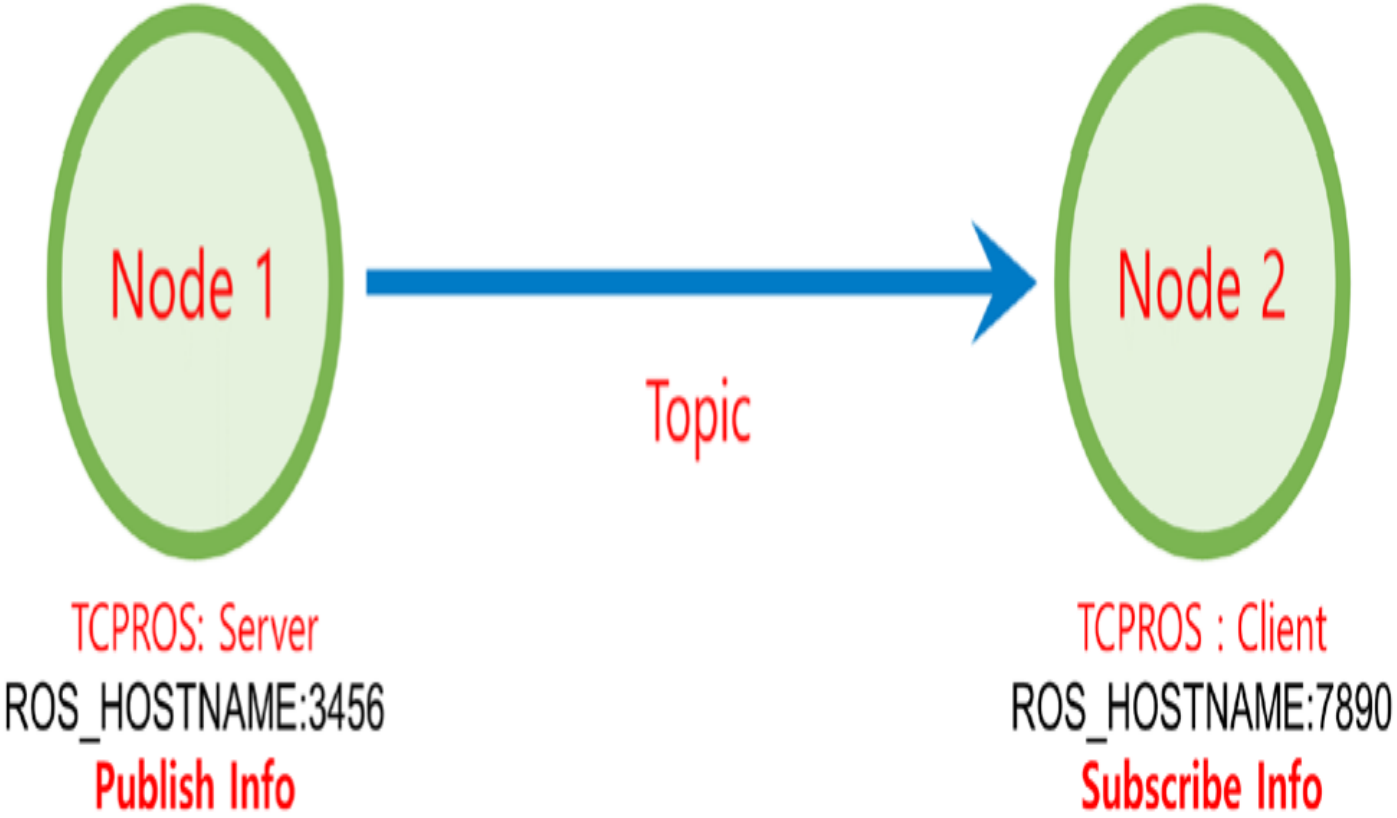
8. Send Message

- The publisher node sends a message to the subscriber node (topic)



Understanding Message Communication

- In Topic mode, message are continuously transmitted unless the connection is terminated. That is continuity.

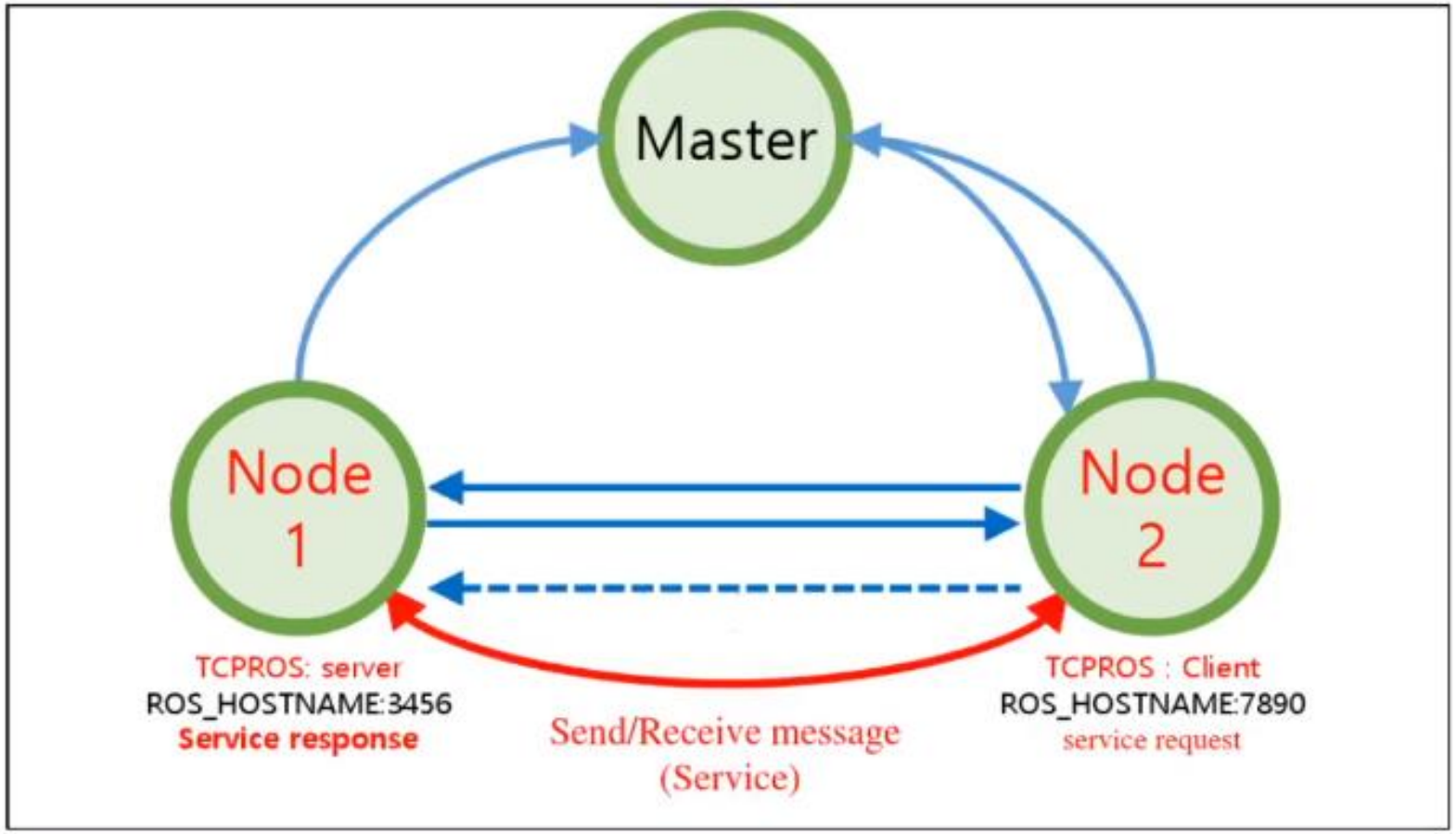


Topic Message Transmission

Understanding Message Communication

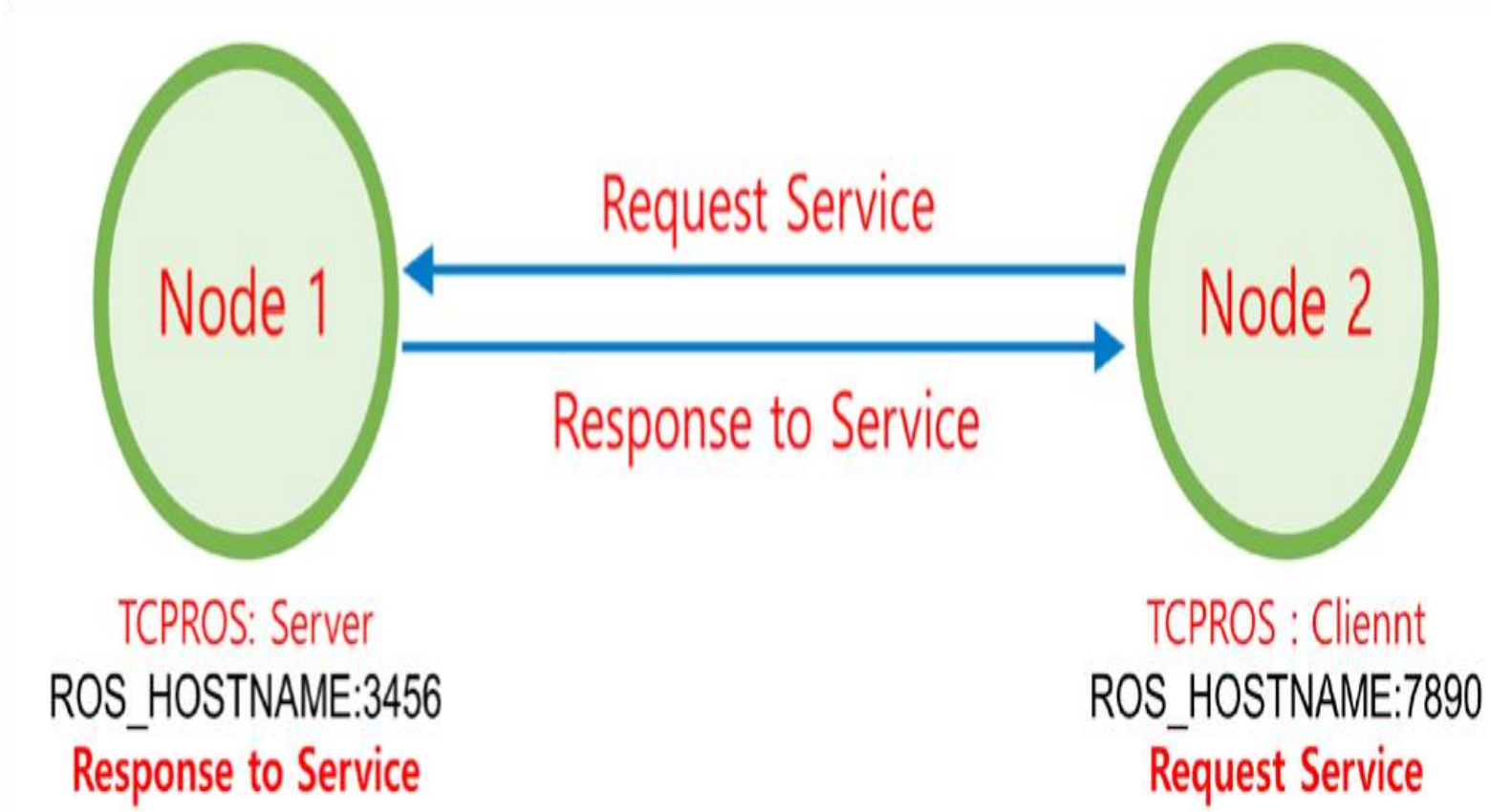
9. Service Request and Response

- For only once, service request and service response are performed and disconnected from each other



Understanding Message Communication

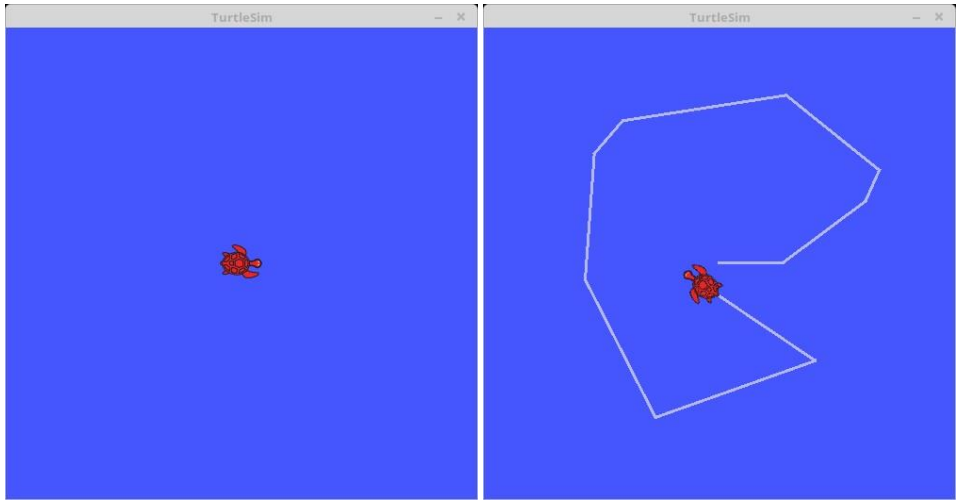
- Unlike the topic, the service connects only once and disconnected after a service request and a service response are performed. That is, it is one-time.

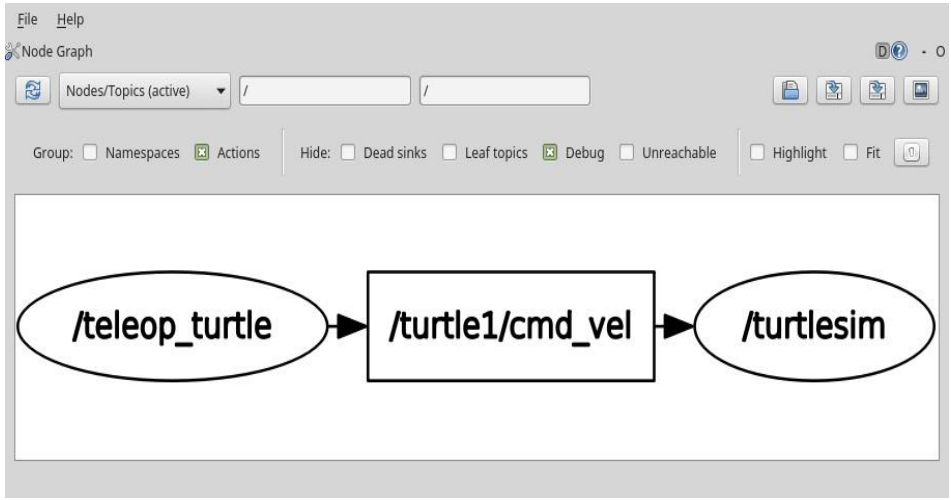


Summing up again

Understanding the message communication concept

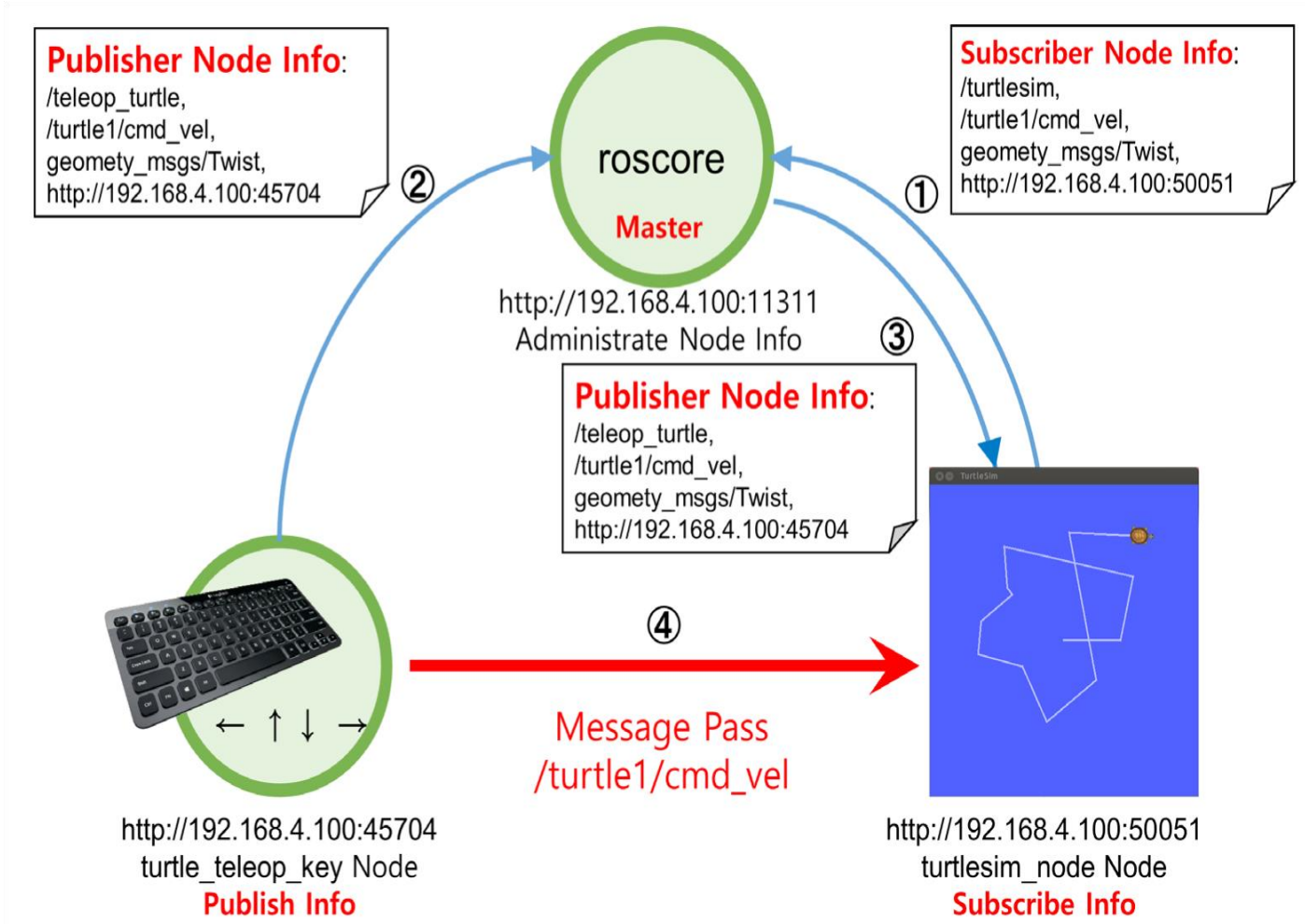
- turtlesim package
- roscore
- rosrun turtlesim turtlesim_node
- rosrun turtlesim turtle_teleop_key
- rosrun rqt_graph rqt_graph





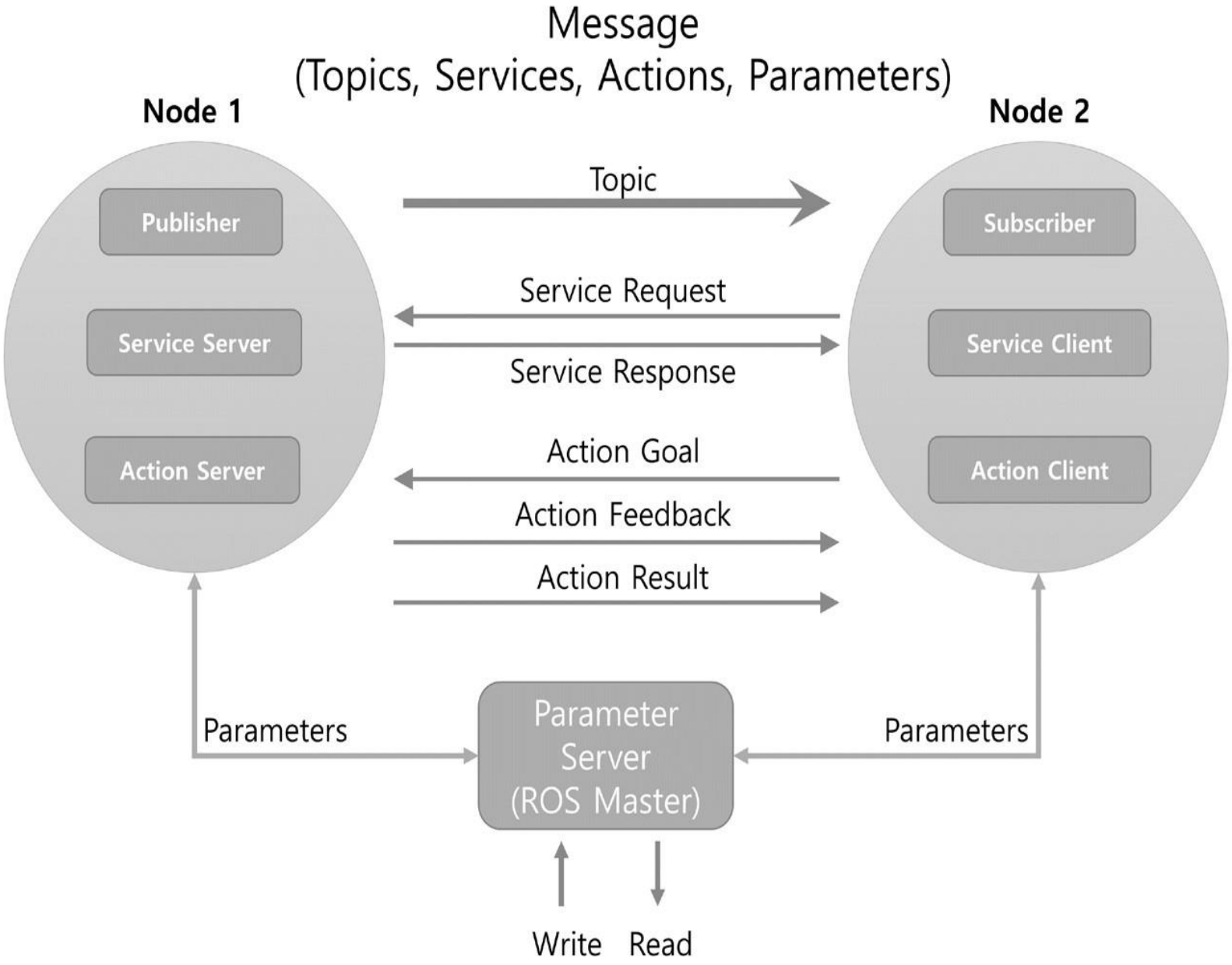
Catching the message communication concept

10. Example! turtlesim



Message

ROS Message


Message
(Topics, Services, Actions, Parameters)
Node 1
Node 2
Publisher
Subscriber
Topic
Service Server
Service Client
Service Request
Service Response
Action Server
Action Client
Action Goal
Action Feedback
Action Result
Parameters
Parameter Server (ROS Master)
Parameters
Write
Read

ROS shell commands

Commands	Command description	Detailed Description
roscd	ros+cd (changes directory)	Move to the directory of the specified ROS package
rosls	ros+ls(lists file)	Check the list of files in the ROS package
rosed	ros+ed (editor)	Edit file in the ROS package
roscp	ros+cp(copies files)	Copy file in the ROS package
rospd	ros+pushd	Add Directories to the ROS Directory Index
rosd	ros+directory	Check the ROS directory index

ROS shell commands

Commands	Command description	Detailed Description
roscore	ros+core	-master (ROS name service) -rosout (record logs) -parameter server (Manage parameters)
rosrun	ros+run	Run the Node
roslaunch	ros+ed (editor)	Run multiple nodes with options
rosclean	ros+clean	Check or delete ROS log files
rostopic	ros+topic	Check ROS topic information
rosservice	ros+service	Check ROS service information

ROS shell commands

Commands	Command description	Detailed Description
rosnode	ros+node	Check ROS node information
rosparam	ros+param(parameter)	Check and modify ROS parameter information
rosbag	ros+bag	ROS message recording, playback
rosmsg	ros+msg	Check ROS message information
rossrv	ros+srv	Check ROS service information
roswtf	ros+wtf	ROS system Inspection
rosversion	ros+version	Check ROS Package, Release Version information

ROS shell commands

Commands	Detailed Description
catkin_create_pkg	Automatically generates a package
catkin_make	Builds based on the Catkin build system
catkin_eclipse	Converts the package created by the Catkin build system for Eclipse
catkin_prepare_release	Log cleanup and version Lagging for release
catkin_generate_changelog	Creates or Updates the CHANGELOG.rst file for release
catkin_init_workspace	Initializes the workspace of Catkin build system
catkin_find	Search catkin
catkin_make clean	Clean all projects in workspace

ROS shell commands

Commands	Command description	Detailed Description
rospack	ros+pack(package)	Displays information related to the ROS package
rosinstall	ros+install	Installs additional package for ROS
rosdep	ros+dep(dependicies)	Install dependency files for the pacakage
roslocate	ros+locate	Acquires information regarding ROS package
roscreate-pkg	ros+create-pkg	Automatically generates ROS package(used in old rosbuild system)
rosmake	ros+make	Builds the ROS package (formerly used by the rosbuild system)