Introduction ROS - 1

Jaeseok Kim
The BioRobotics Institute - Service Robotics and Ambient Assisted Living Lab

Contents

1. **About ROS**
2. Purpose of ROS
3. Configuration of ROS
4. Features of ROS
5. ROS Installation & Development Environment
6. ROS terminology
7. Message communication
8. ROS commands
9. ROS tools
10. Communication between heterogeneous devices

History of personal gadgets

	PC (Personal Computer)	PP (Personal Phone)	PR (Personal Robot)
User			
Service			
Operating System			
Hardware			

What is ROS

ROS (Robot Operating System) is an meta-operating system for your robot.

It provides libraries and tools to **help software developers create robot applications.** It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more.

ROS is licensed under an **open source**,BSD license.

1. About ROS
2. **Purpose of ROS**
3. Configuration of ROS
4. Features of ROS
5. ROS Installation & Development Environment
6. ROS terminology
7. Message communication
8. ROS commands
9. ROS tools
10. Communication between heterogeneous devices

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 tool, visualization tool Rviz, GUI toolbar rqt, 3D simulator Gazebo
  - Supports modeling, sensing, recognition, navigation, and manipulation functions
  - commonly used in robotics
  - **Create Robotics Ecosystem!**

Meta-Operating system

1. About ROS
2. Purpose of ROS
3. **Configuration of ROS**
4. Features of ROS
5. ROS Installation & Development Environment
6. ROS terminology
7. Message communication
8. ROS commands
9. ROS tools
10. Communication between heterogeneous devices

ROS Configuration

Client Layer	roscpp	rospy	roslisp	rosjava	roslibjs		
Robotics Application	MoveIt!	navigatioin	executive smach	descartes	rospeex		
	teleop pkgs	rocon	mapviz	people	ar track		
Robotics Application Framework	dynamic reconfigure	robot localization	robot pose ekf	Industrial core	robot web tools	ros realtime	mavros
	tf	robot state publisher	robot model	ros control	calibration	octomap mapping	
	vision opencv	image pipeline	laser pipeline	perception pcl	laser filters	ecto	
Communication Layer	common msgs	rosbag	actionlib	pluginlib	rostopic	rosservice	
	rosnode	roslaunch	rosparam	rosmaster	rosout	ros console	
Hardware Interface Layer	camera drivers	GPS/IMU drivers	joystick drivers	range finder drivers	3d sensor drivers	diagnostics	
	audio common	force/torque sensor drivers	power supply drivers	rosserial	ethercat drivers	ros canopen	
Software Development Tools	RViz	rqt	wstool	rospack	catkin	rosdep	
Simulation	gazebo ros pkgs	stage ros					

http://wiki.ros.org/APIs

1. About ROS
2. Purpose of ROS
3. Configuration of ROS
4. **Features of ROS**
5. ROS Installation & Development Environment
6. ROS terminology
7. Message communication
8. ROS commands
9. ROS tools
10. Communication between heterogeneous devices

Feature 1. Communication infrastructure

- Provides data communication between nodes
- Support for message transfer interface, which is commonly referred as the middleware

- **Message parsing function**
  - Provides communication system frequently used in robot development
  - Message transfer interface between nodes facilitating encapsulation and code reuse

- **Message Record and Play**
  - Messages that are transmitted/received between nodes can be stored and reused as needed
  - It is possible to repeat an experiment based on stored messages, and it is easy to develop algorithm

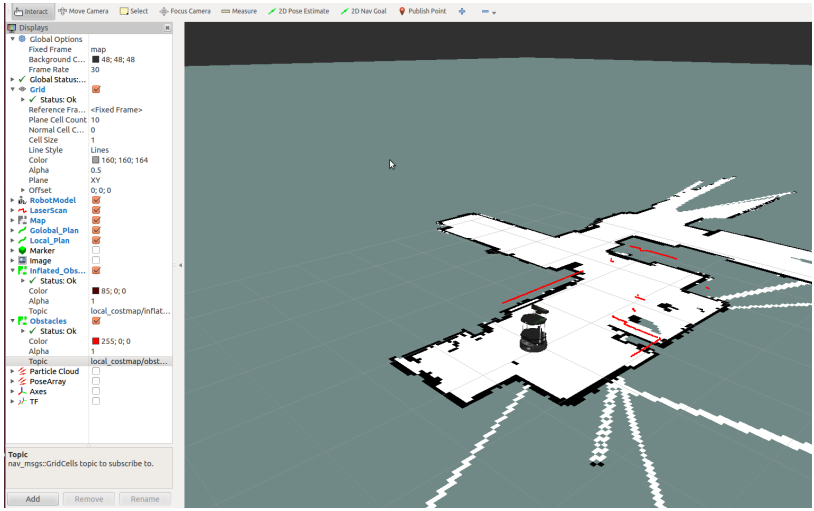
- **Use of various programming languages due to the use of messages**
    - Since data exchange between nodes use messages, each node can be written in different languages
    - Client libraries: roscpp, rospy, roslisp, rosjava, roslua, roscs, roseus, PhaROS, rosR

- **Distributed parameter system**
  - Variables used in the system are created as global key values so they can be shared, modified and applied in real-time

Feature 2. Various functions related to robots

- **Define a standard message for a robot**
  - Modularization by defining standard message such as camera, IMU, laser sensor / odometry, navigation data such as route and map, inducing collaborative work, and improving efficiency

- **Robot geometry library**
  - Provides TF to calculate the relative coordinates of robot, sensor, etc.



- **Robot description language**
  - XML document describing physical characteristics of the robot

- **Diagnostic system**
  - Provides diagnostic system to grasp the state of the robot

- **Sensing / recognition**
  - Sensor drivers, libraries for sensing / recognition

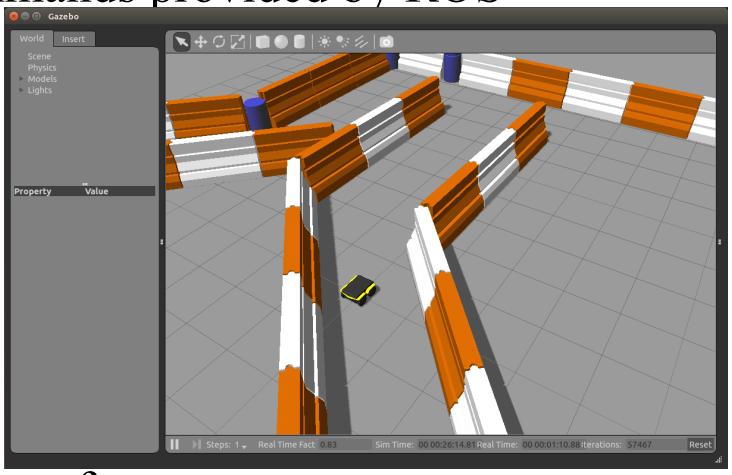
- **Navigation**
  - Estimation of poses (position / posture) of robots commonly used in robots, provision of self position estimation in the map
  - SLAM required for map creation, and Navigation library for navigating to destinations within the created map

- **Manipulation**
  - Provides various Manipulation libraries to support IK and FK used in robot arm as well as pick and place of application
  - Provides GUI manipulation tools (MoveIt!)

Feature 3. Various development tools

- Provides various development tools needed for robot development
- Improving the efficiency of robot development

- **Command-Line Tools**
  - Access to the robot and use ROS functions only with commands provided by ROS without GUI



- **RViz**
  - Provide powerful 3D visualization tool
  - Visualize sensor data such as laser, camera, etc.
  - Represent robot outline and planned motion

- **RQT**
  - Provides Qt-based framework for developing graphic interface
  - Displays connection information among nodes (rqt_graph)
  - Values such as encoder, voltage, numbers that change over time (rqt_plot)
  - Records and plays data in the form of message (rqt_bag)

- **Gazebo**
  - 3D simulator with physics engine. Supports robot, sensor, environmental models
  - Highly Compatible with ROS

1. About ROS
2. Purpose of ROS
3. Configuration of ROS
4. Features of ROS
5. **ROS Installation & Development Environment**
6. ROS terminology
7. Message communication
8. ROS commands
9. ROS tools
10. Communication between heterogeneous devices

ROS 1 Line Installation

- Wget https://raw.githubusercontent.com/ROBOTIS-GIT/robotis_tools/master/install_ros_kinetic.sh && chmod 755 ./install_ros_kinetic.sh && bash ./install_ros_kinetic.sh

ROS Manual Installation

- **ROS Installation**

  http://wiki.ros.org/kinetic/Installation/Ubuntu

- **ROS Environment Setting**

  http://wiki.ros.org/ROS/Tutorials/InstallingandConfiguringROSEnvironment

ROS Environment Setting

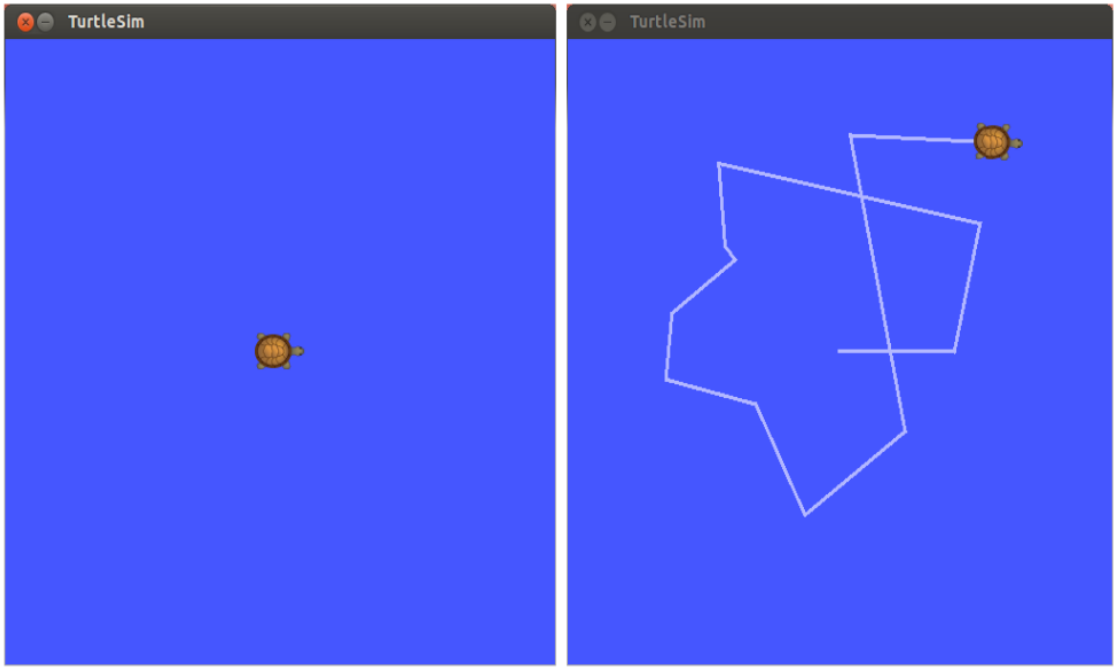
- nano ~/.bashrc

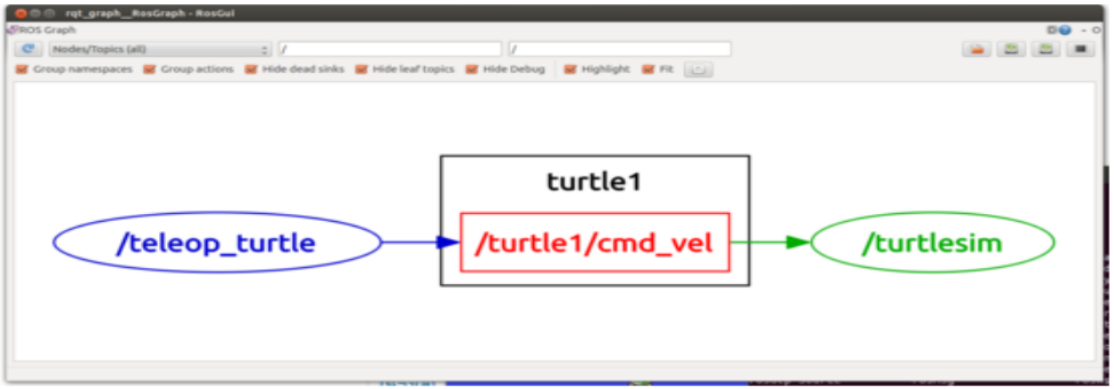
- source /opt/ros/kinetic/setup.bash
- source ~/catkin_ws/devel/setup.bash
- export ROS_MASTER_URI=http://localhost:11311
- export ROS_HOSTNAME=localhost
- #export ROS_MASTER_URI=http://192.168.1.100:11311
- #export ROS_HOSTNAME=192.168.1.100

ROS Operation Test

turtlesim package

- roscore
- rosrun turtlesim turtlesim_node
- rosrun turtlesim turtle_teleop_key
- rosrun rqt_graph rqt_grap





Integrated Development Environment(IDE) available on ROS

- Recommendation : Sublime Text
- Install: https://www.sublimetext.com/3
- Advantage: A light text editor oriented, Fast

Installation

wget -qO - https://download.sublimetext.com/sublimehq-pub.gpg | sudo apt-key add -

echo "deb https://download.sublimetext.com/ apt/stable/" | sudo tee /etc/apt/sources.list.d/sublime-text.list

sudo apt-get update

sudo apt-get install sublime-text

1. About ROS
2. Purpose of ROS
3. Configuration of ROS
4. Features of ROS
5. ROS Installation & Development Environment
6. **ROS terminology**
7. Message communication
8. ROS commands
9. ROS tools
10. Communication between heterogeneous devices

ROS terms

- **Node**

  The smallest unit of executable processors. 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**

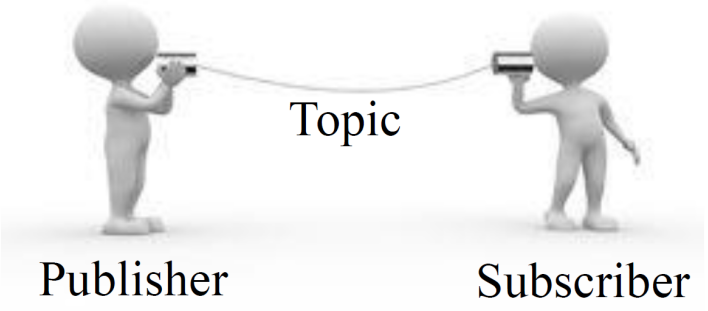
  One or more nodes, information for node execution, etc. Also, bundles of packages are called as metapackages.

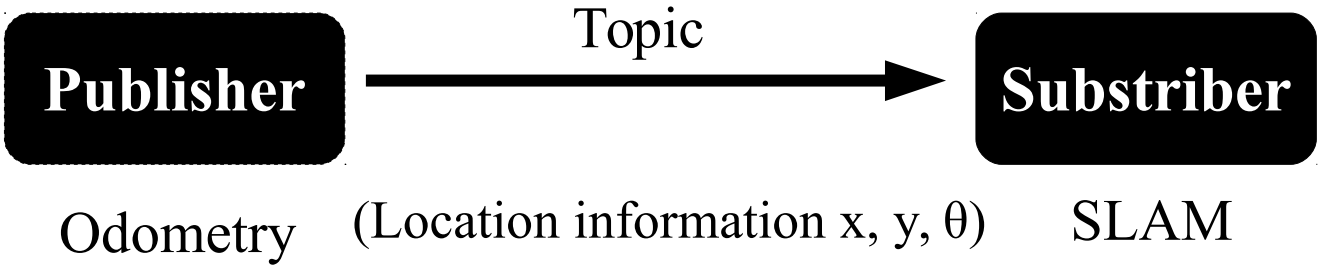
- **Message**

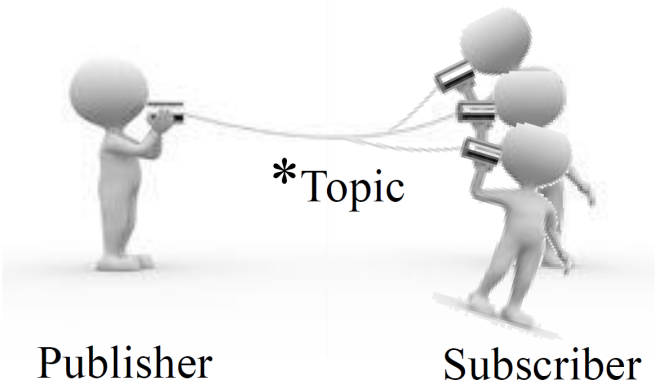
  Data is transmitted and received through message between nodes. Messages can have various types such as integer, floating point, and boolean. You can also use structures such as a simple data structure and an array of messages that hold messages in the message.

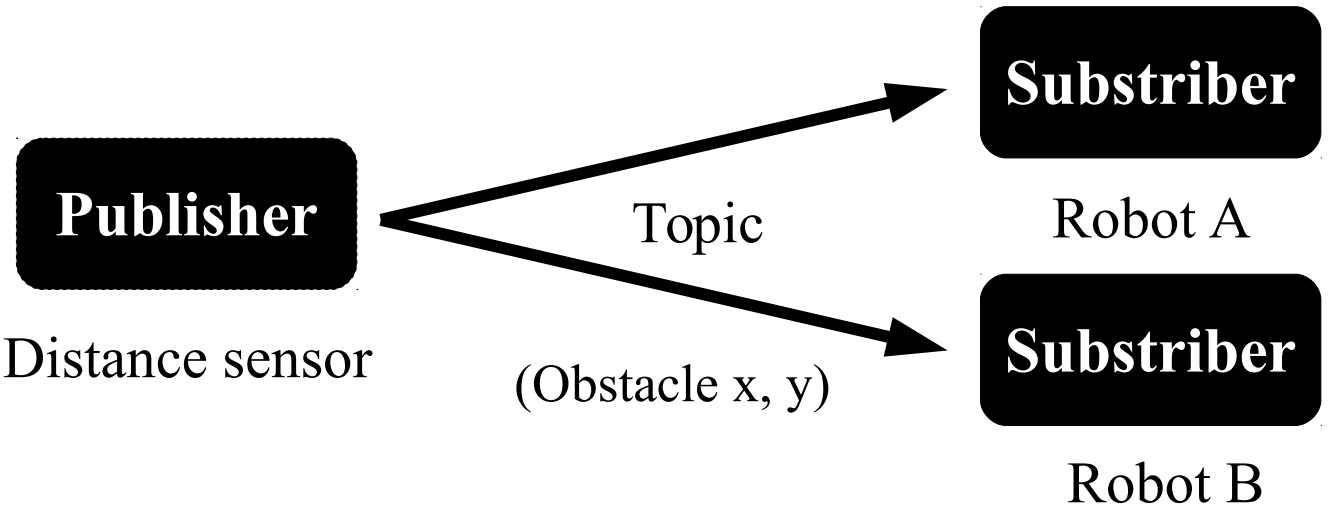
ROS terms

Topic, Publisher, Subscriber







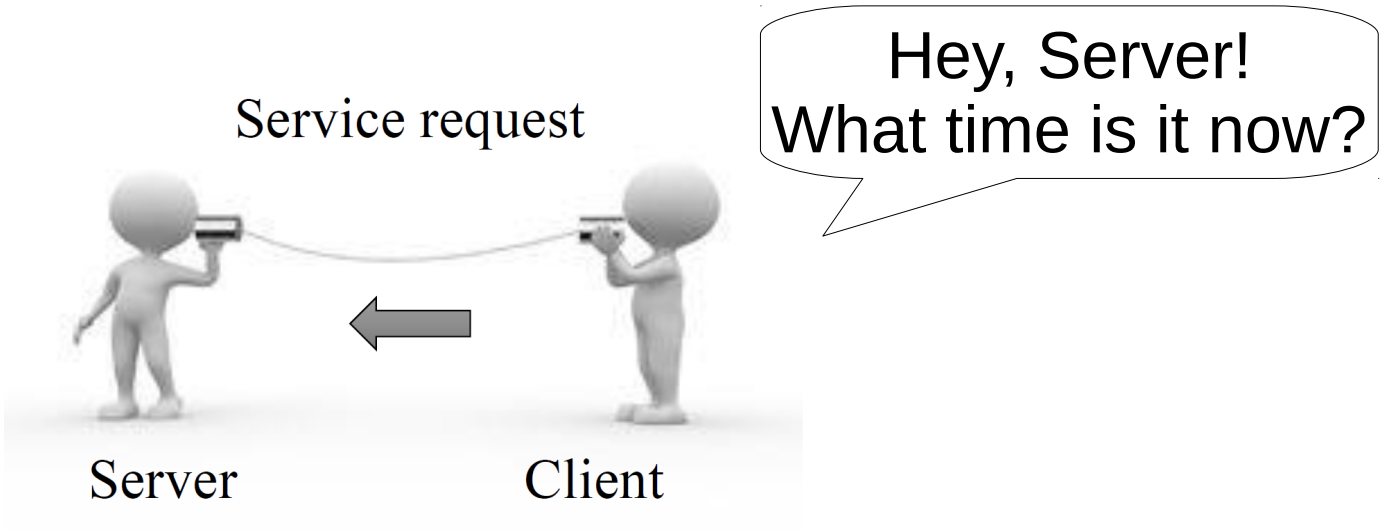


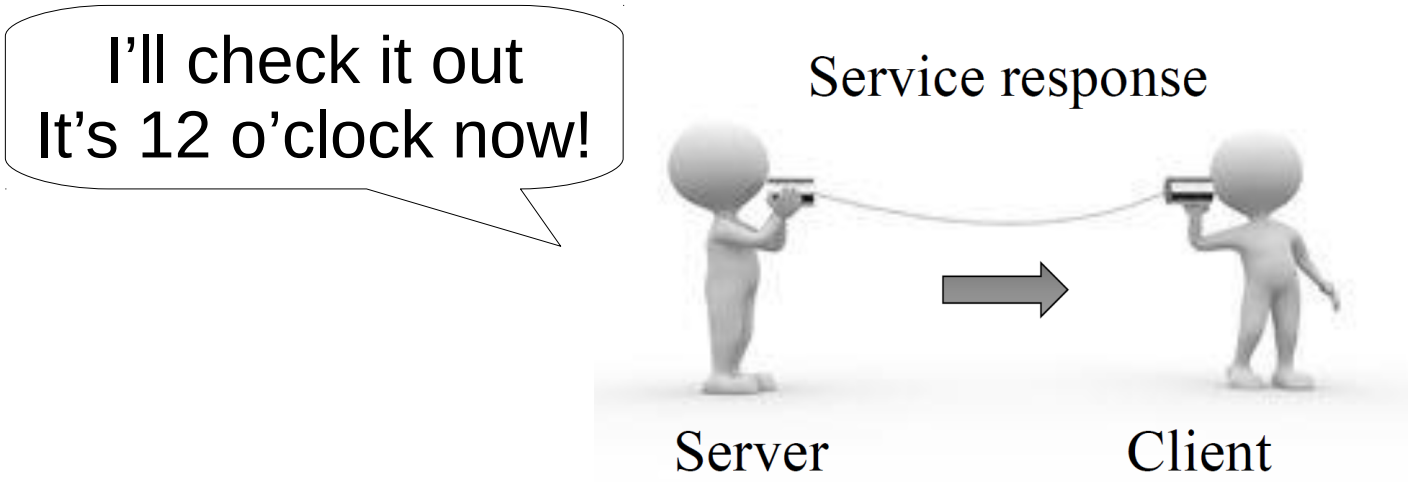
- 1: 1 Publisher and Subscriber communication is also possible for Topic,and 1: N, N: 1, N: N communication is also possible depending on the purpose.

http://www.dreamstime.com/illustration/people-talk-listen-tin-can-phone-communication.html

ROS terms

Service, Service server, Service client

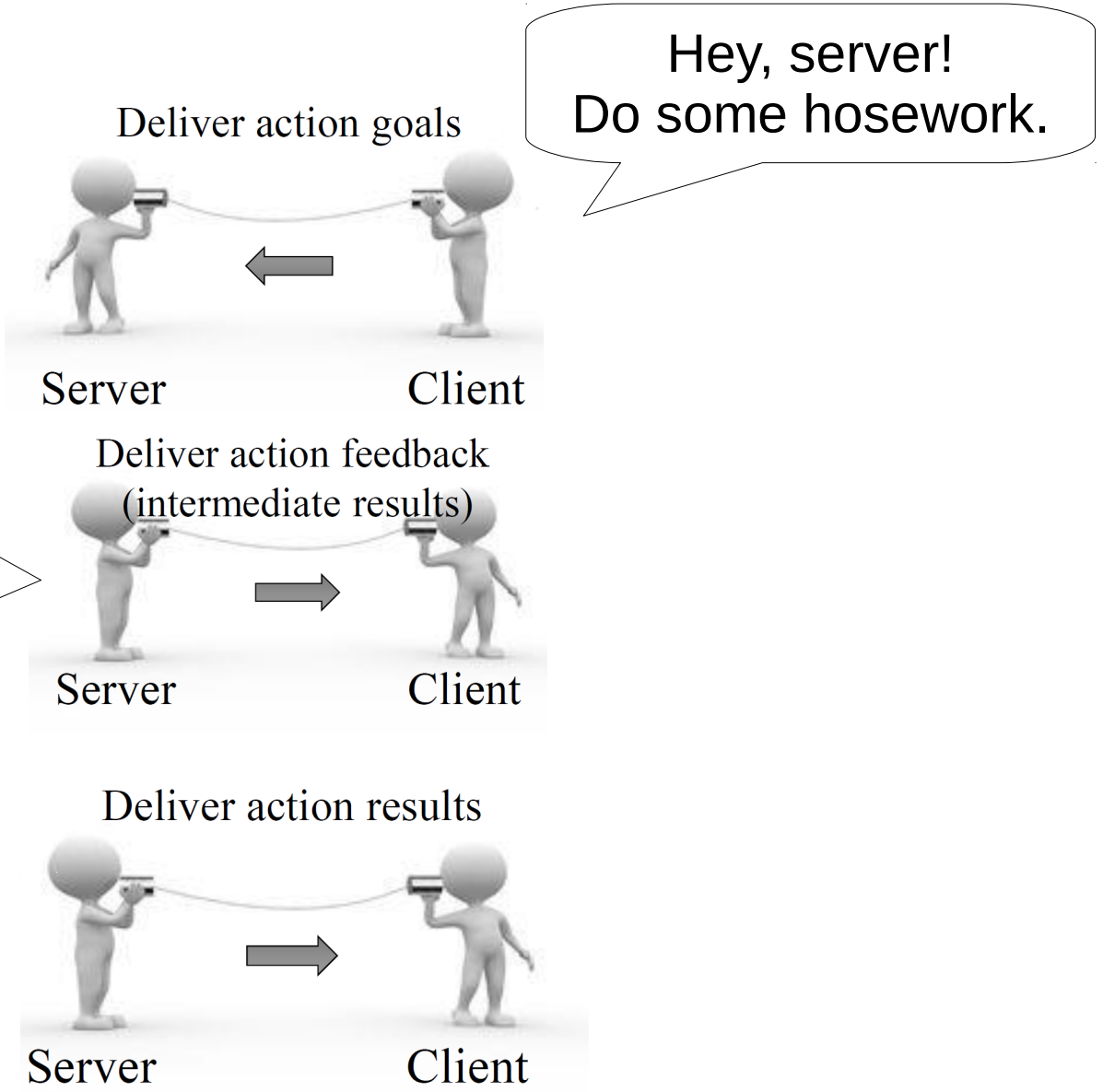




http://www.dreamstime.com/illustration/people-talk-listen-tin-can-phone-communication.html

ROS terms

Action, Action server, Action client

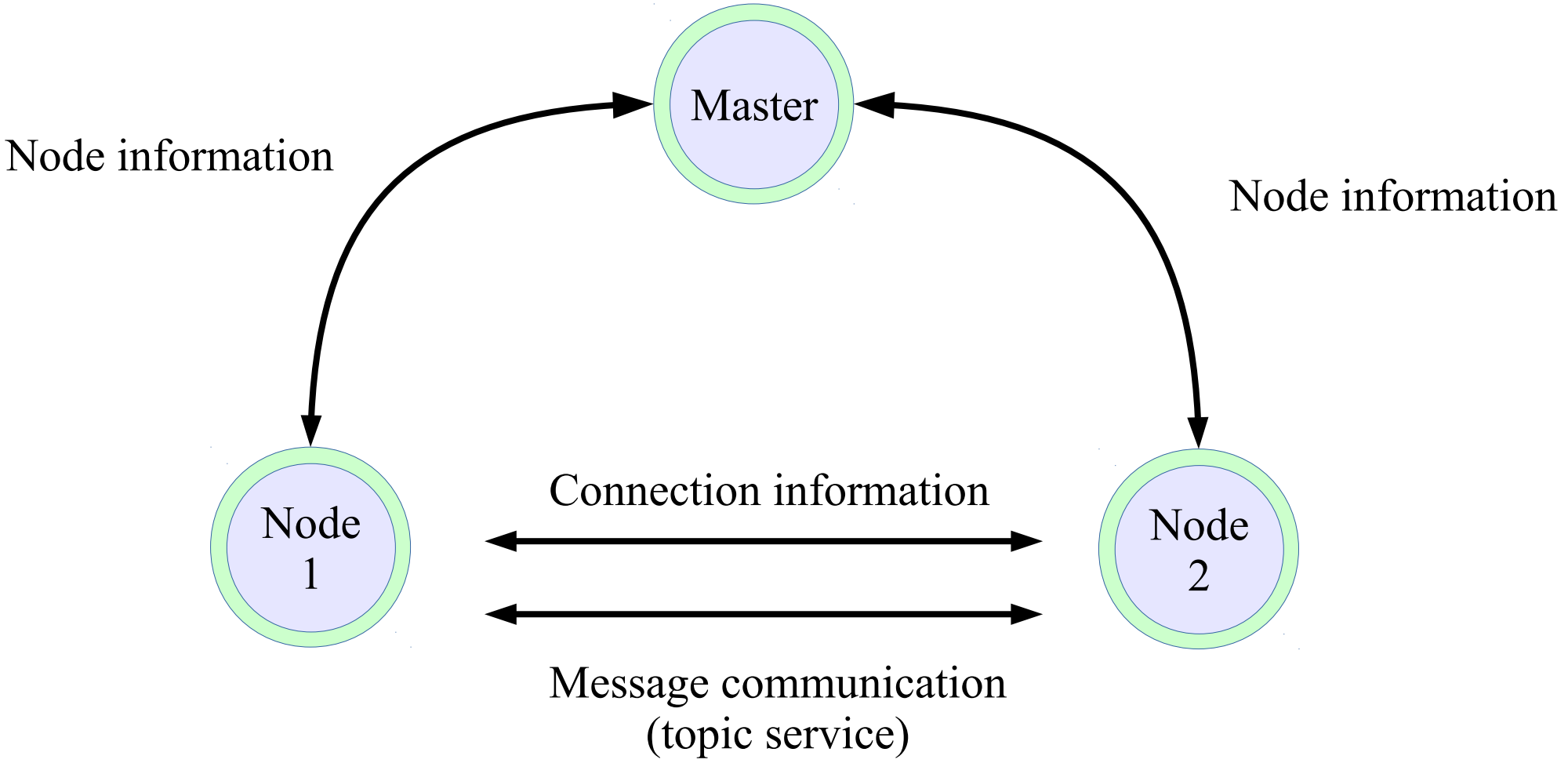


http://www.dreamstime.com/illustration/people-talk-listen-tin-can-phone-communication.html

1. About ROS
2. Purpose of ROS
3. Configuration of ROS
4. Features of ROS
5. ROS Installation & Development Environment
6. ROS terminology
7. **Message communication**
8. ROS commands
9. ROS tools
10. Communication between heterogeneous devices

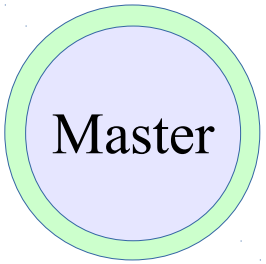
Understanding message communication

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



Understanding message communication

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

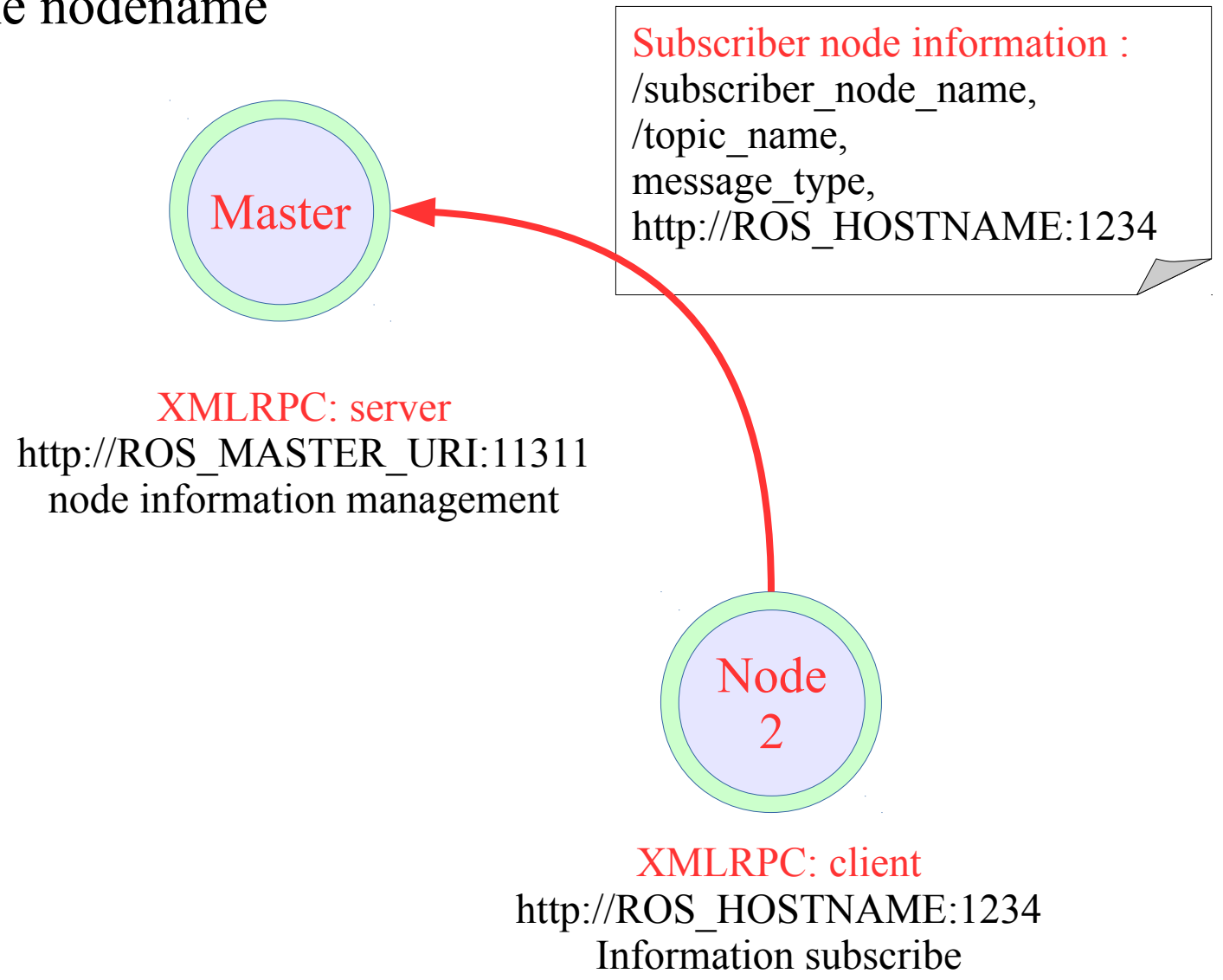


XMLRPC: server
http://ROS_MASTER_URI:11311
Node information management

Understanding message communication

2. Run Subscriber node

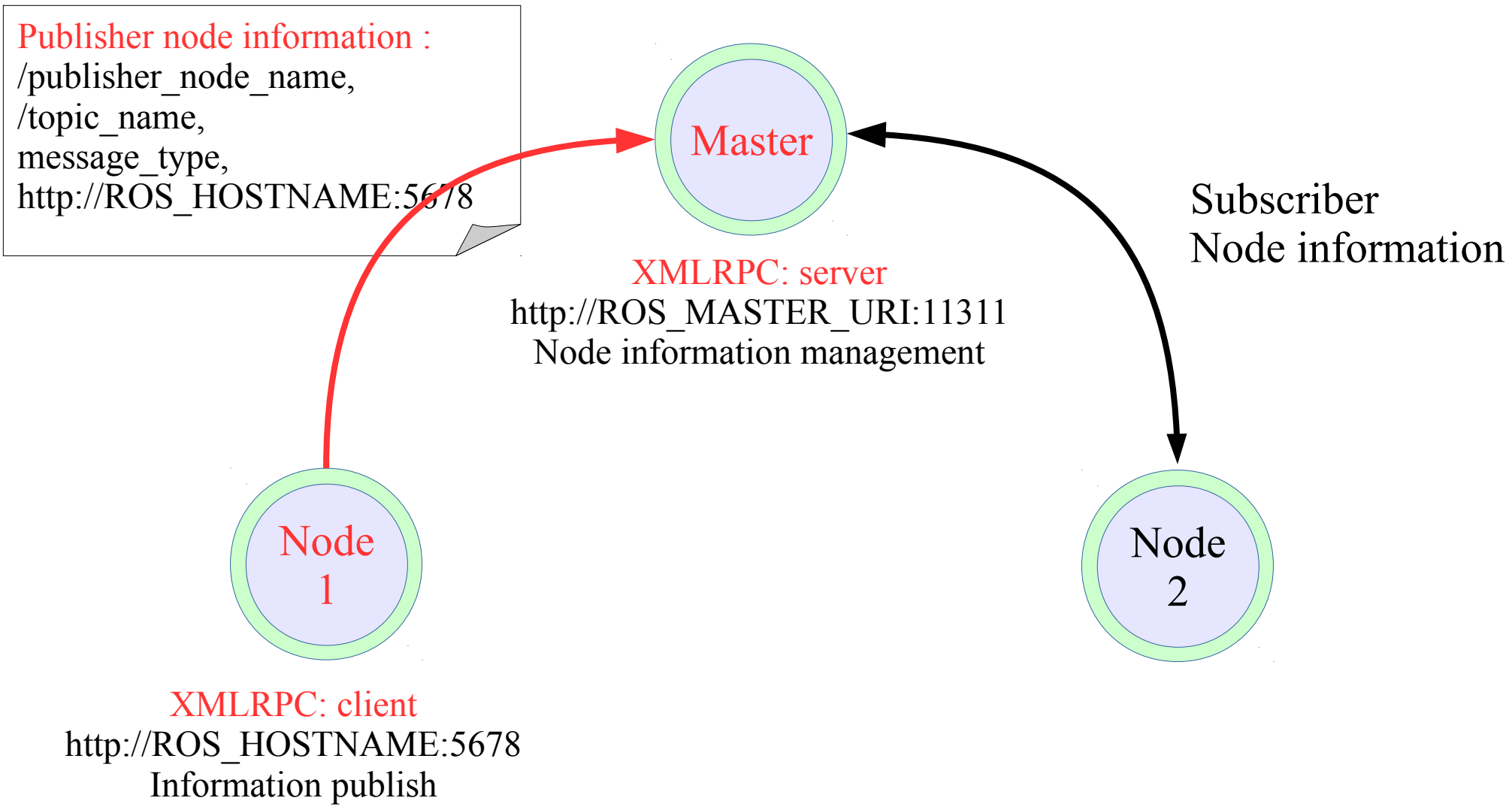
$ rosrun packagename nodename



Understanding message communication

3. Run Publisher node

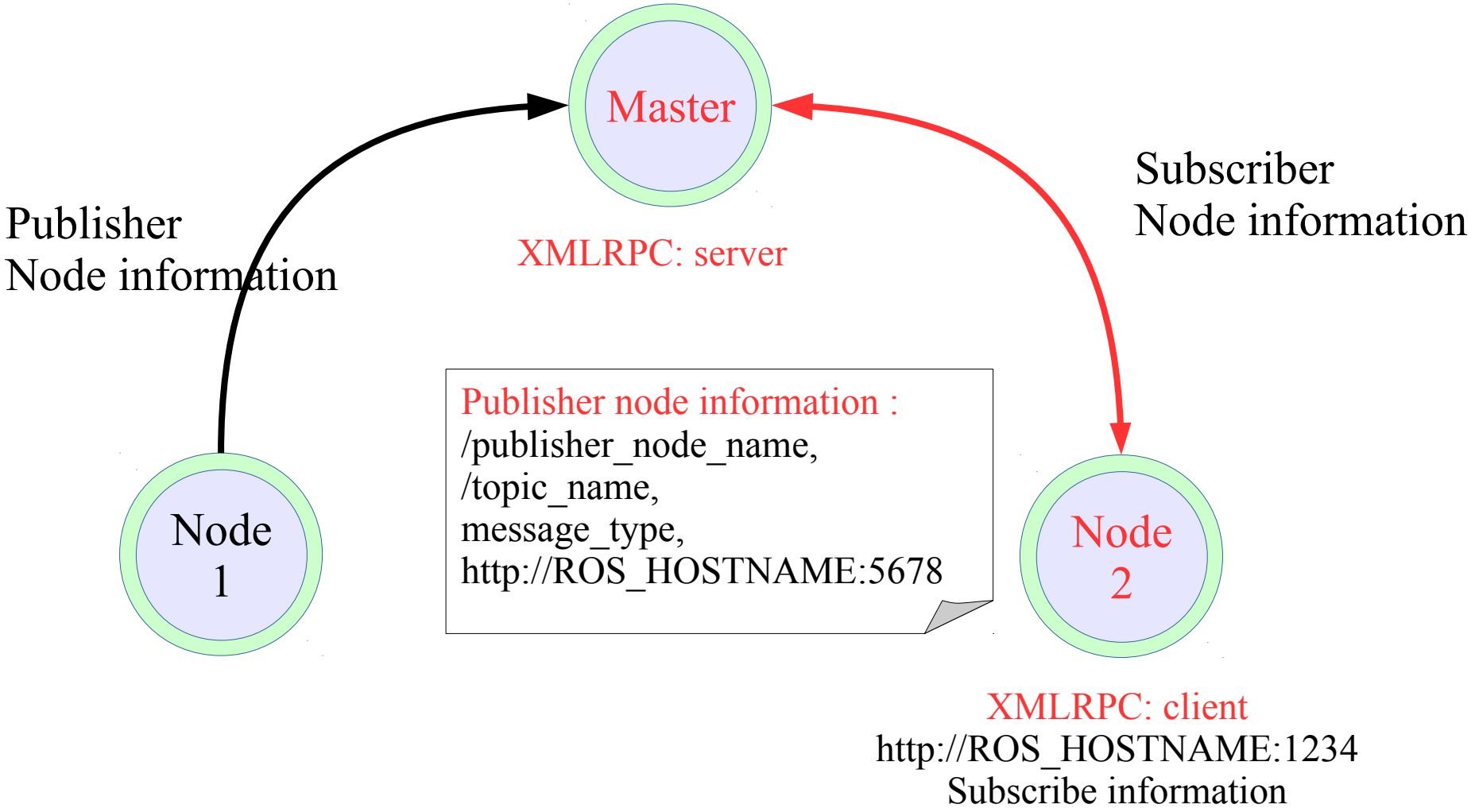
$ rosrun packagename nodename



Understanding message communication

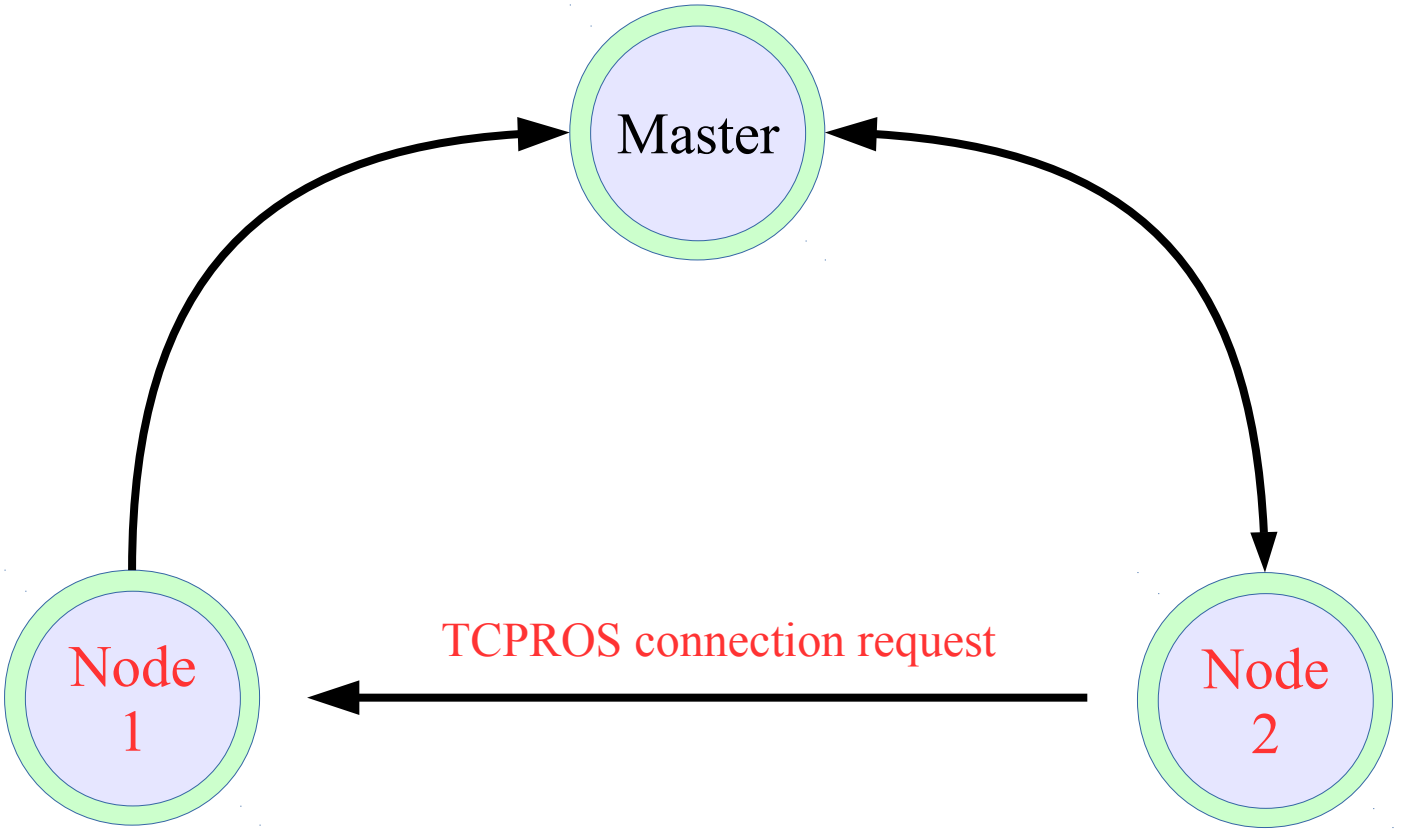
4. Publisher Information

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



Understanding message communication

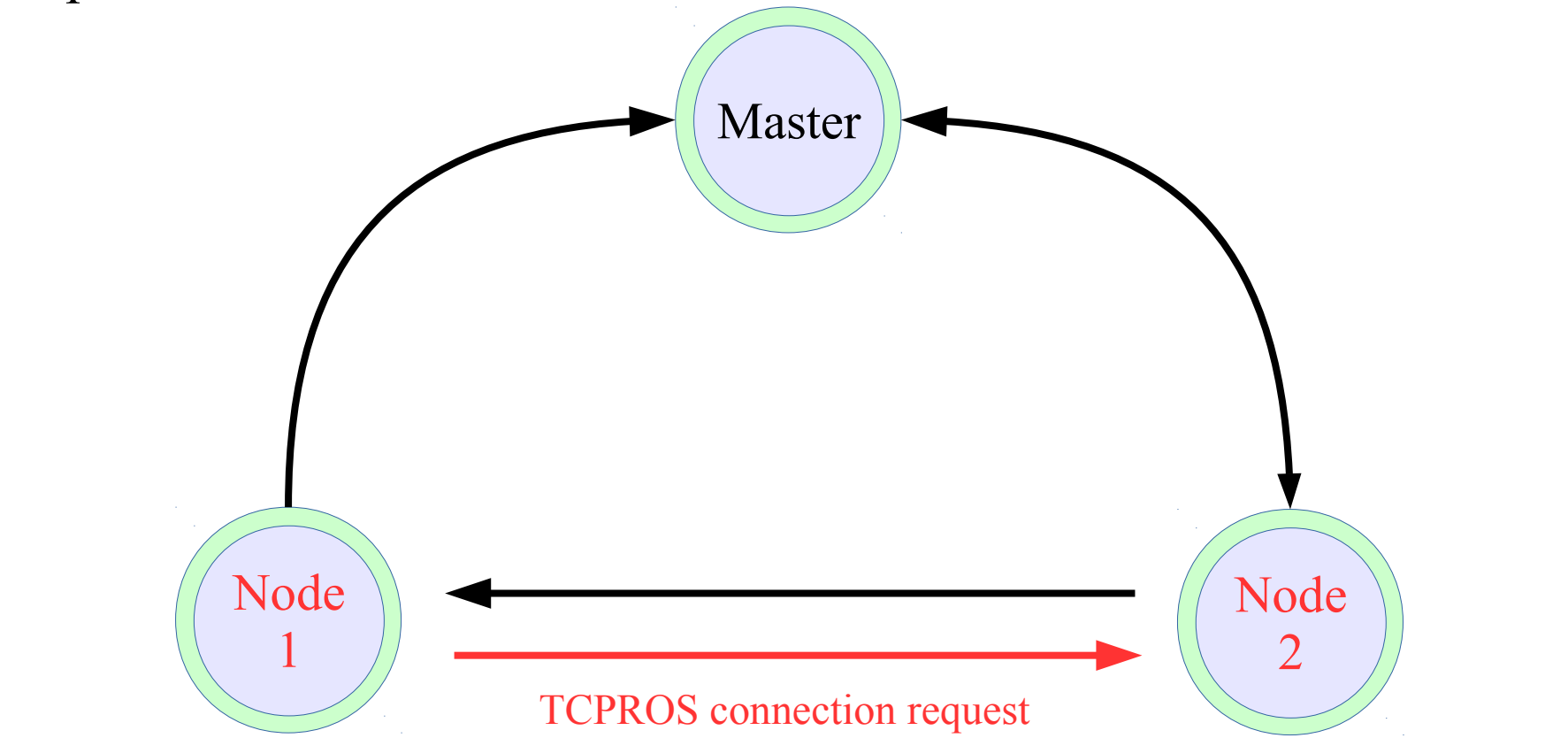
- **5. Request access to the publisher node**
  Request TCPROS connection using the publisher information from the master



Understanding message communication

6. Connection response to subscriber node

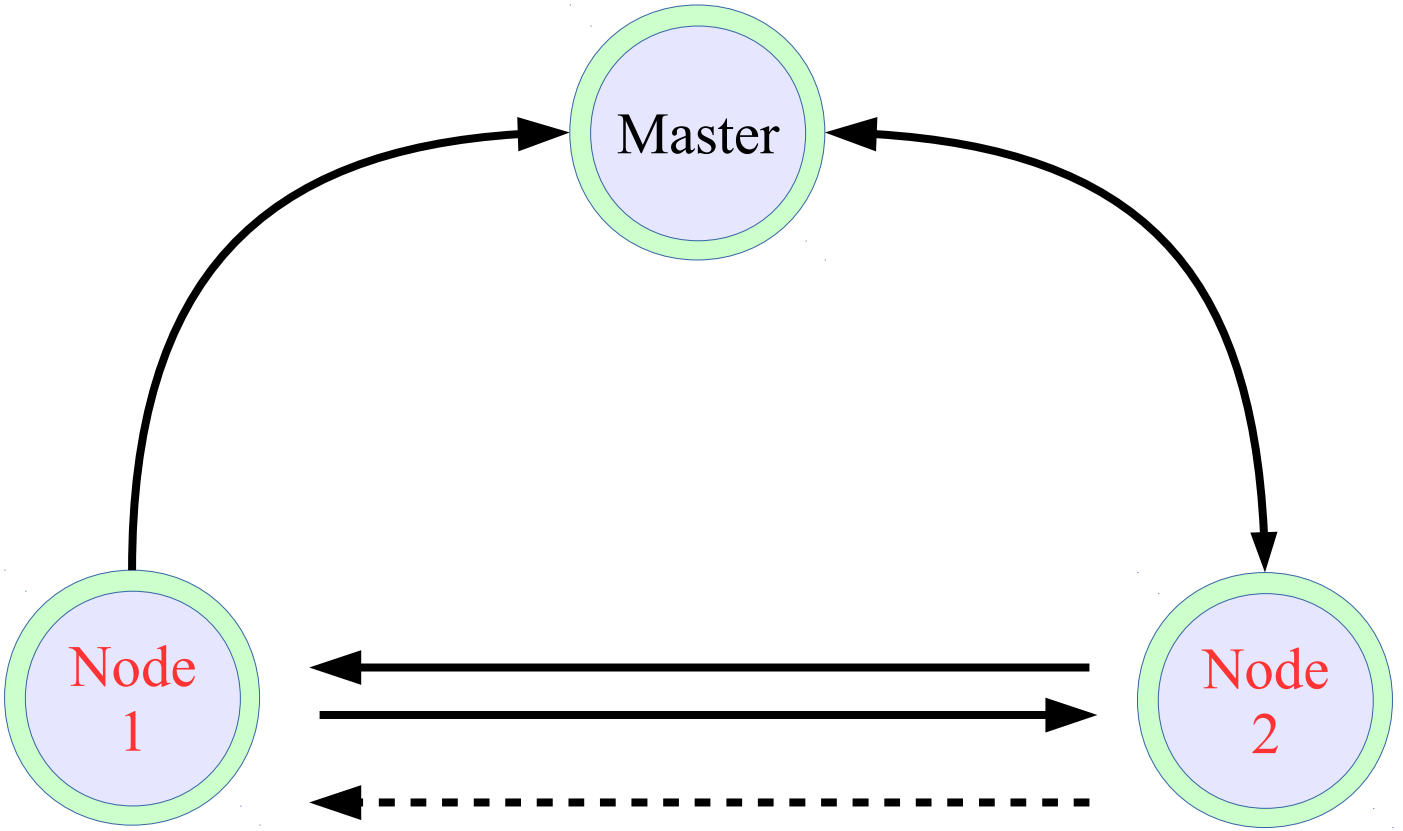
Return TCP URI address and port number corresponding to the connection response



Understanding message communication

7. TCP Connection

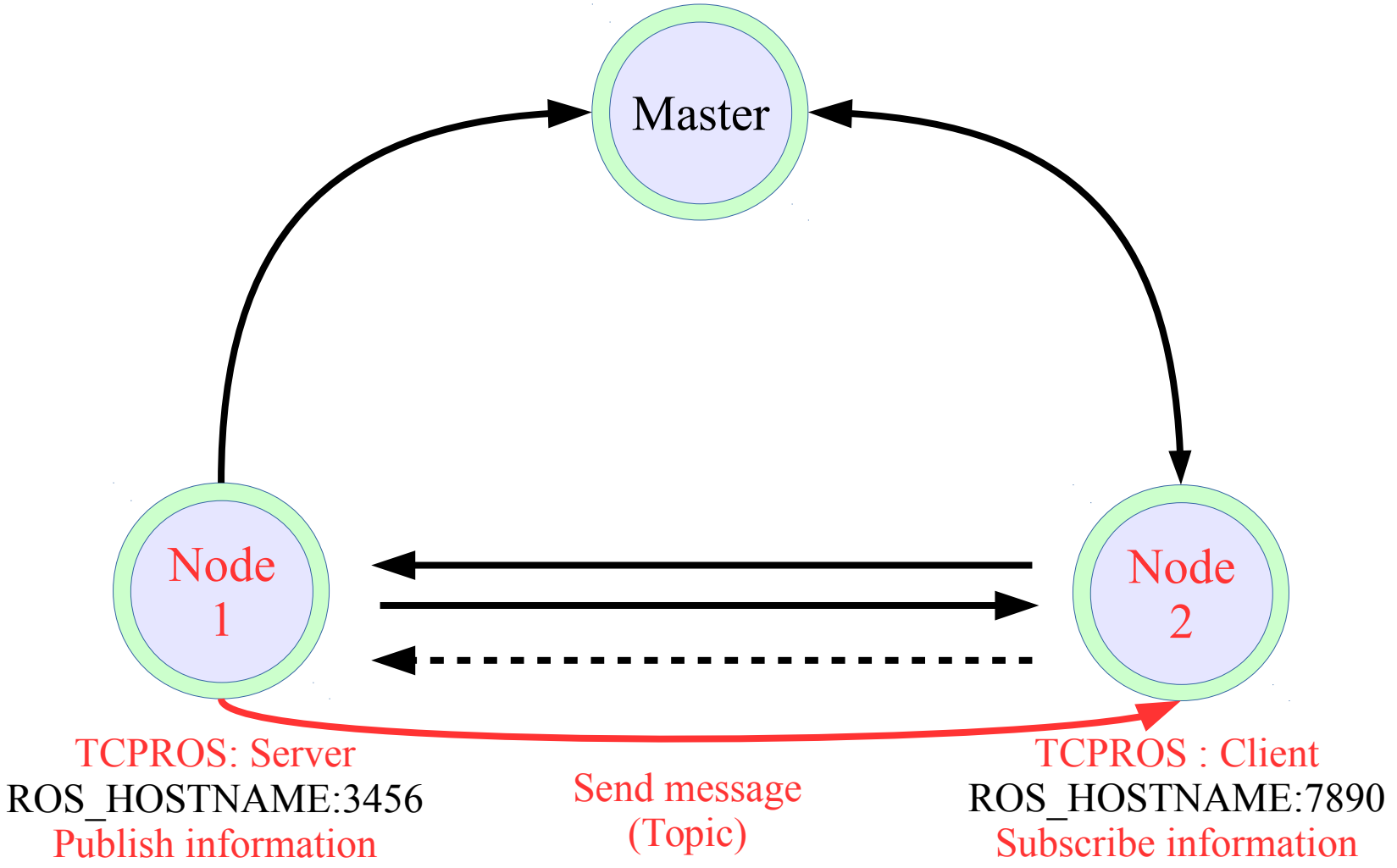
Establish connection with the publisher node using TCPROS.



Understanding message communication

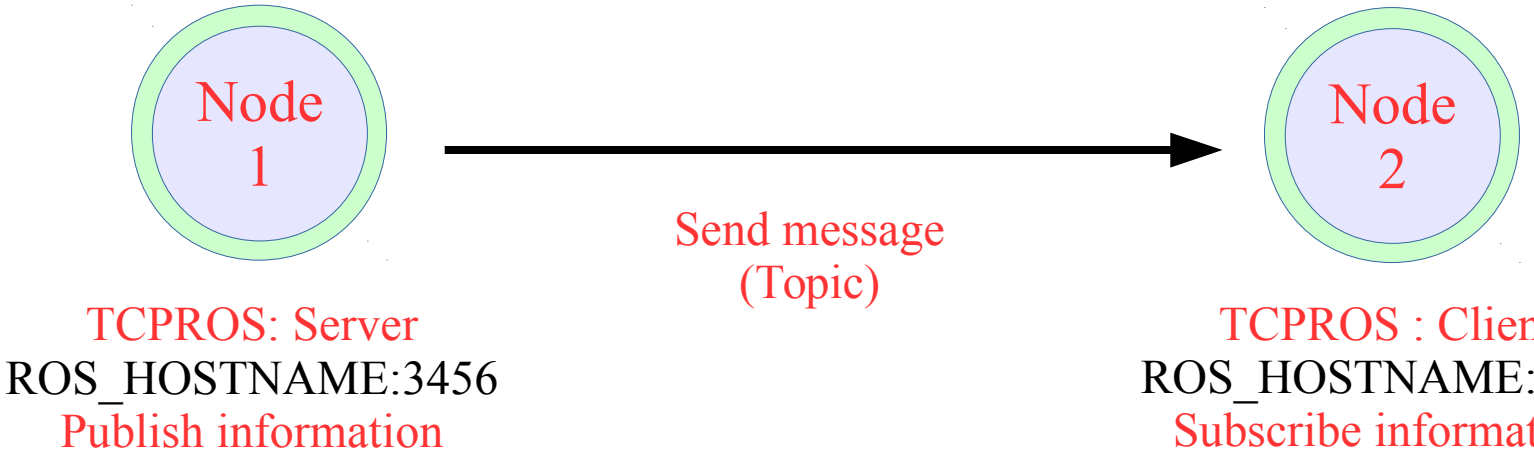
8. Send message

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



Understanding message communication

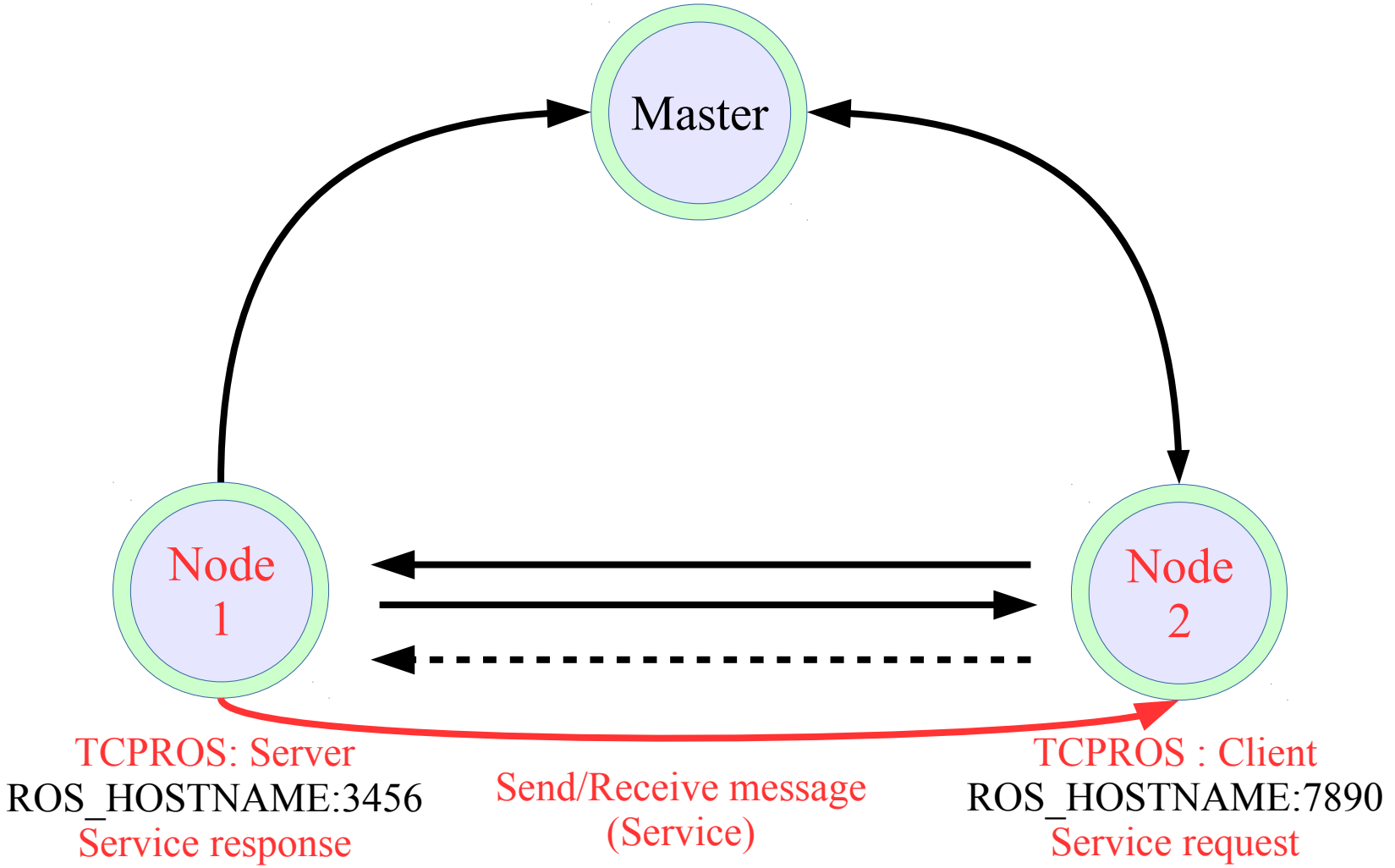
- In Topic mode, messages are continuously transmitted unless the connection is terminated. That is, continuity.



Understanding message communication

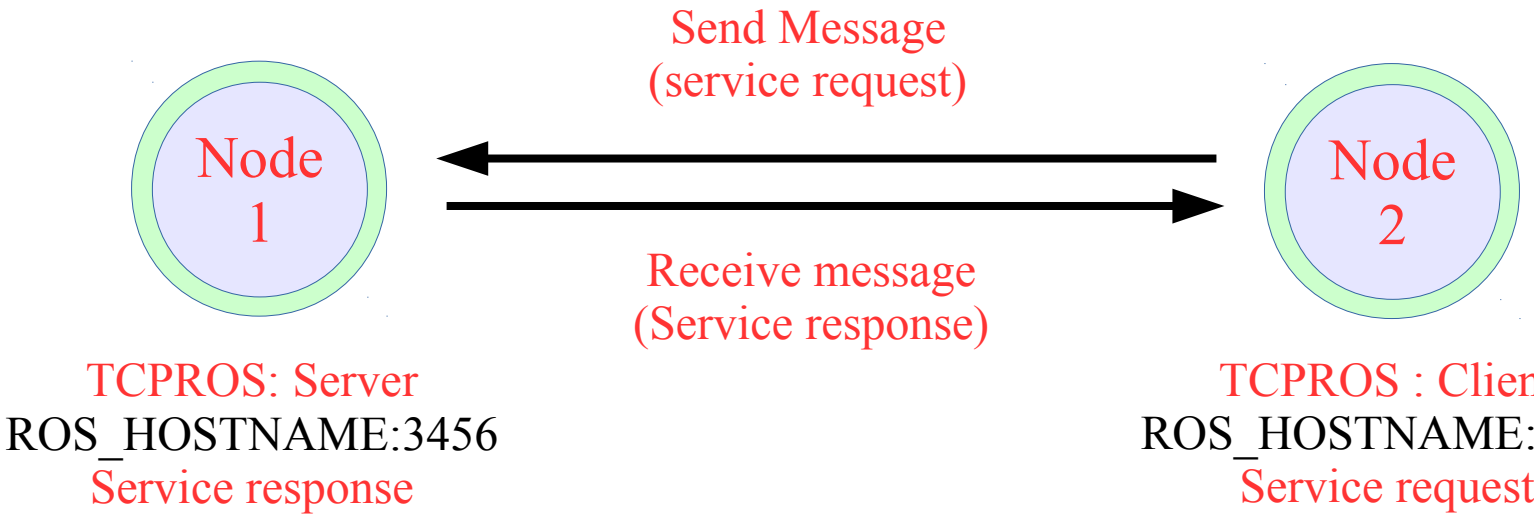
9. Service Request and Response

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



Understanding message communication

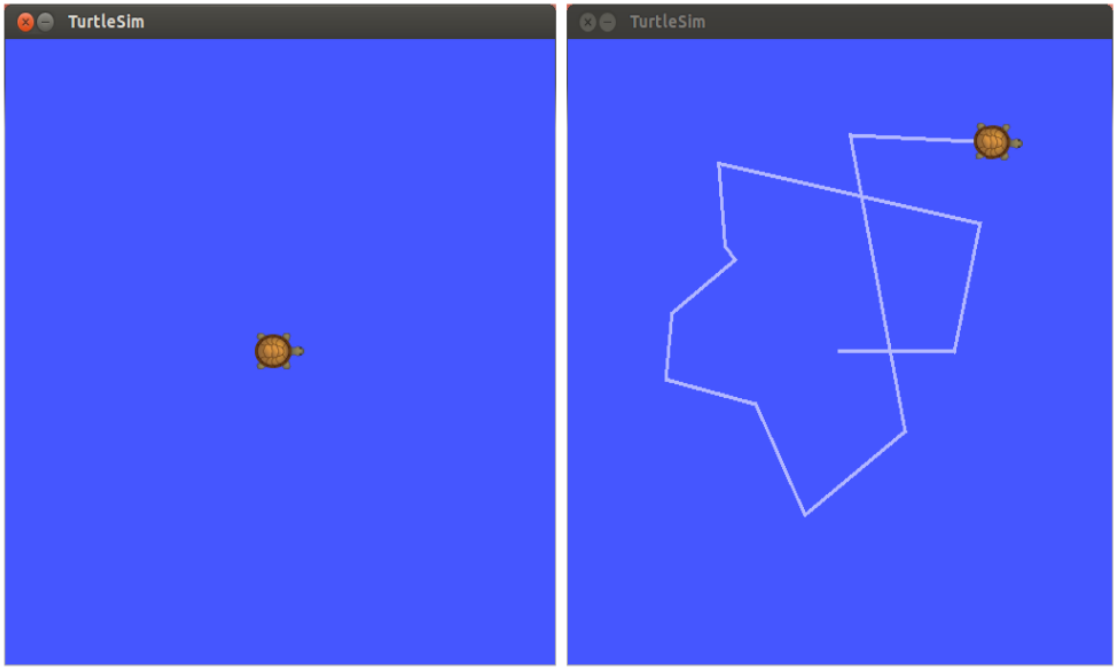
- Unlike the topic, the service connects only once and disconnected after a service request and a service response are performed. That is, it is one-time.

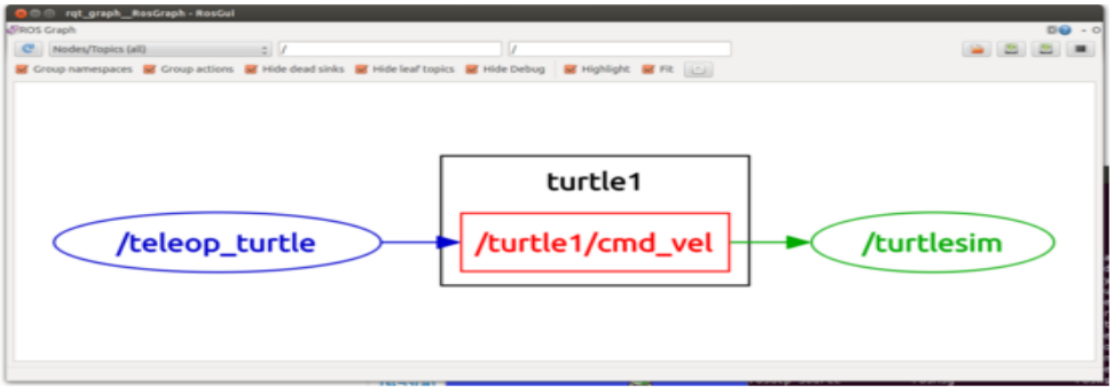


Understanding the message communication concept!

turtlesim package

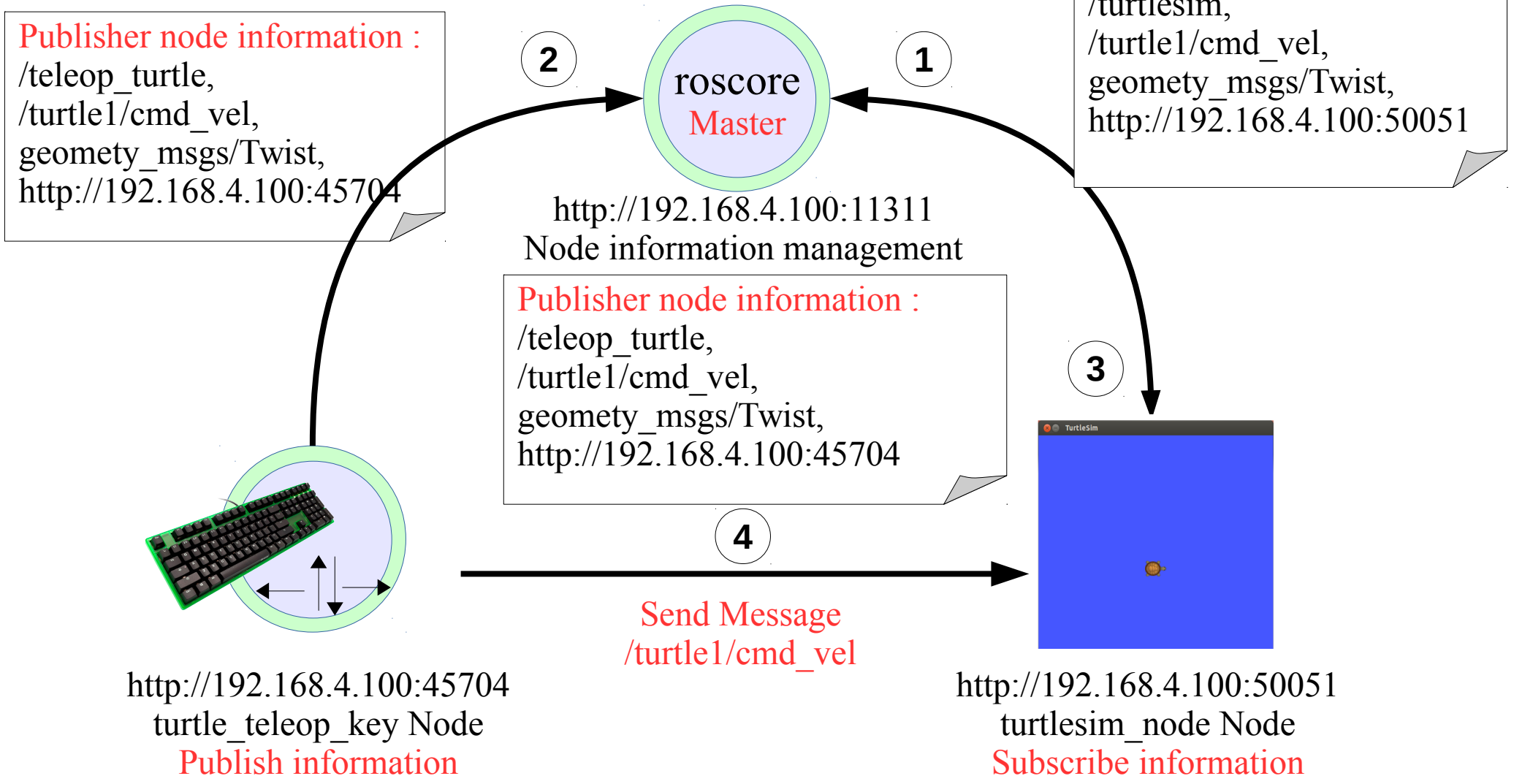
- roscore
- rosrun turtlesim turtlesim_node
- rosrun turtlesim turtle_teleop_key
- rosrun rqt_graph rqt_grap





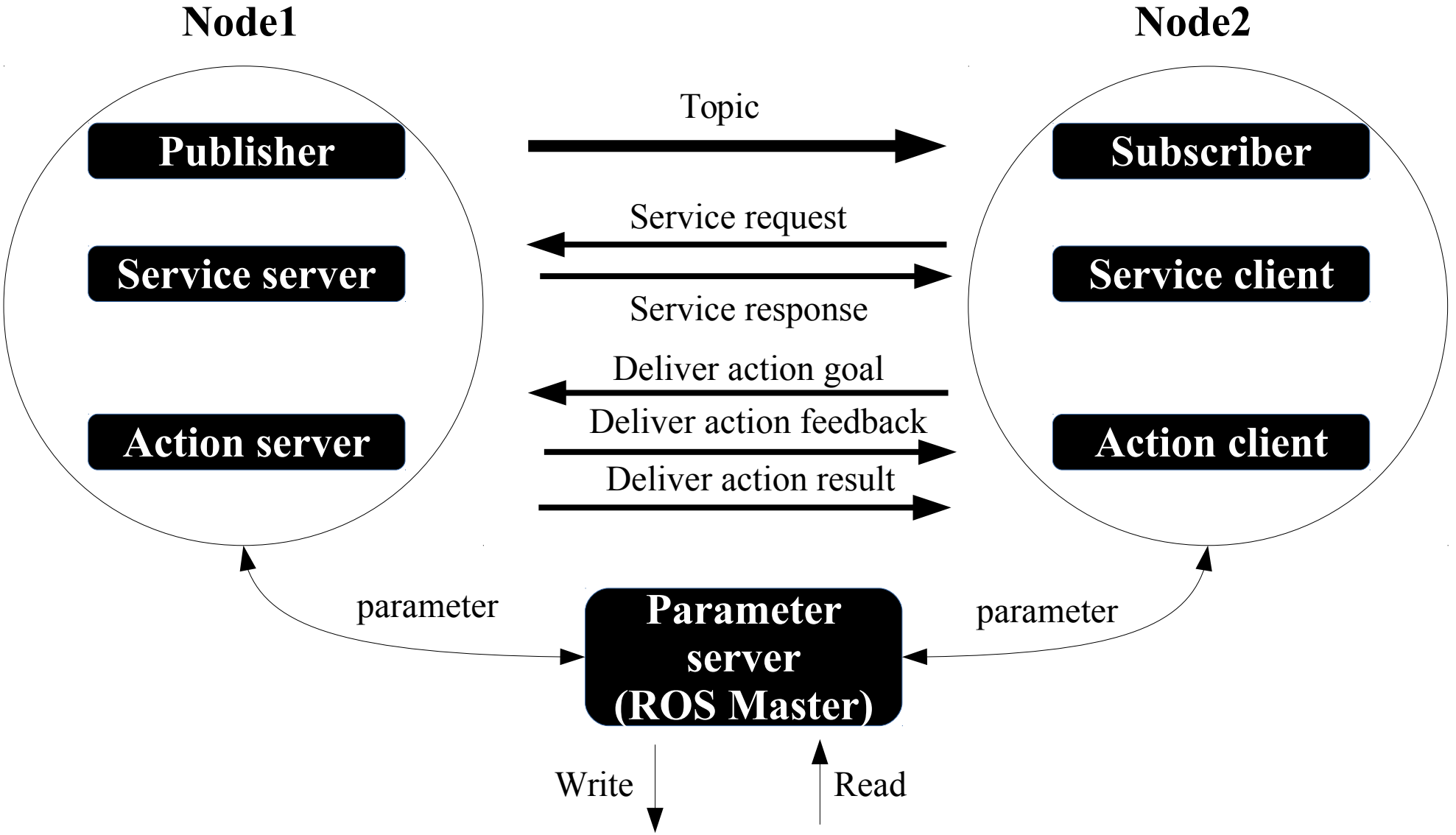
Catching the message communication concept!

- 10. Example! turtlesim



ROS Message

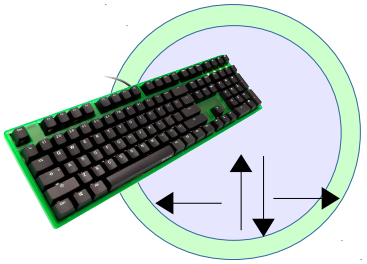
Message communication
(Topic, Service, Action, Parameter)



ROS Message

- Message is a type of data travel around nodes
  - Topics, services, and actions all use messages
    - http://wiki.ros.org/msg
    - http://wiki.ros.org/common_msgs
  - **Simple type**
    - ex) integer, floating point, boolean
    - http://wiki.ros.org/std_msgs
  - **A simple data structure containing messages in a message**
    - ex) geometry_msgs/PoseStamped
    - http://docs.ros.org/api/geometry_msgs/html/msg/PoseStamped.html
  - **An array data structure in which messages are listed**
    - ex) float32[ ] ranges
    - ex) sensor_msgs/LaserScan
    - http://docs.ros.org/api/sensor_msgs/html/msg/LaserScan.html

ROS Message (ex: geometry_msgs/Twist)



Send message
/turtle1/cmd_vel
(geometry_msgs/Twist)

http://192.168.4.100:45704
turtle_teleop_key Node
Publish information

http://192.168.4.100:50051
turtlesim_node Node
Subscribe information

[geometry_msgs/Twist]

Vector3 linear
Vector3 angular

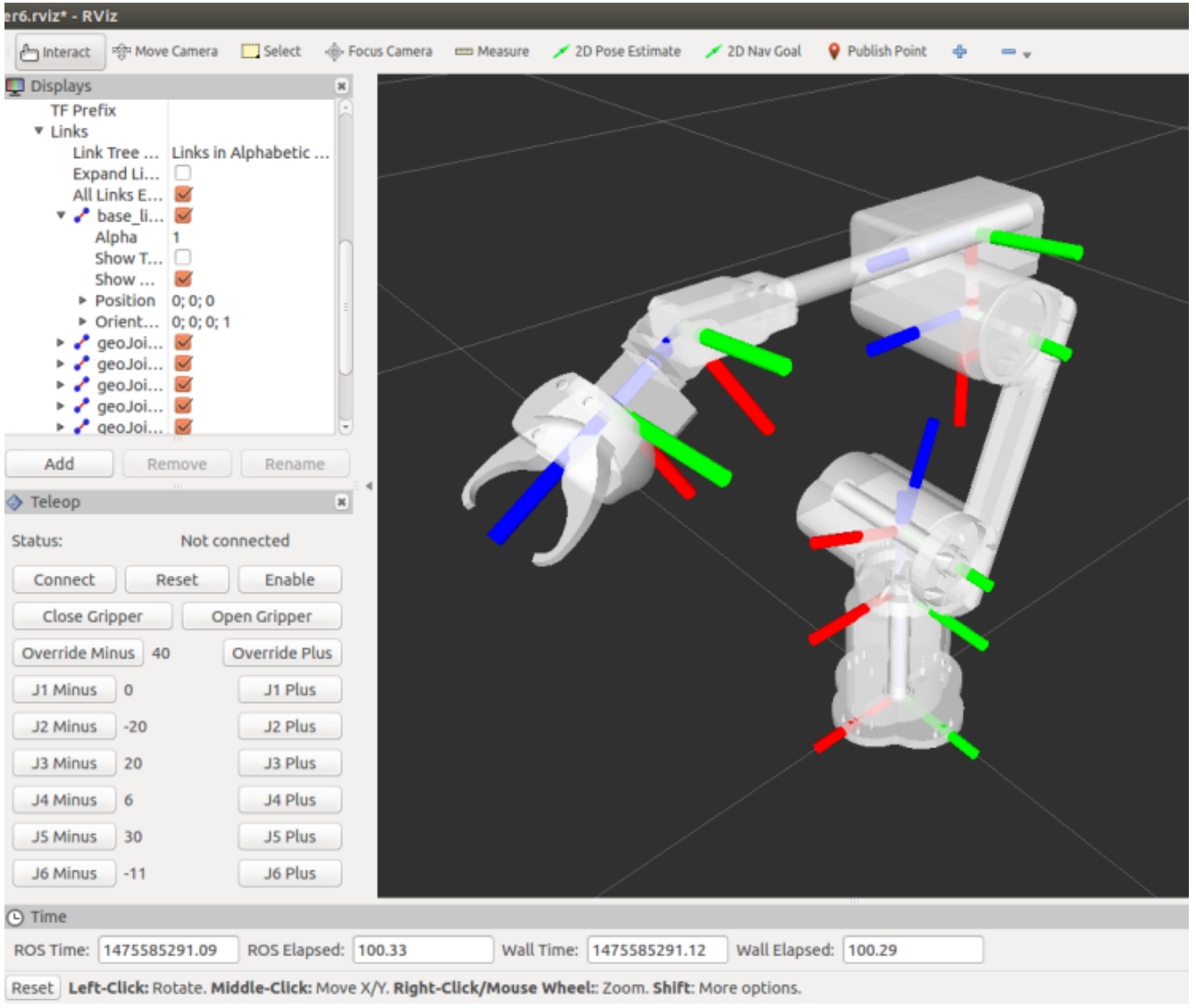
[geometry_msgs/Vector3]
float64 x
float64 y
float64 z

[geometry_msgs/Vector3]
float64 x
float64 y
float64 z

http://docs.ros.org/api/geometry_msgs/html/msg/Twist.html

Coordinate transformation(TF, transform)

- Relative coordinate transformation of each joint
  - Indicates the relationship between joints in the form of tree structure



ROS execution commands

Command	Importance	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+launch	Run multiple nodes with options
rosclean	★★☆	ros+clean	Check or delete ROS log files

1. About ROS
2. Purpose of ROS
3. Configuration of ROS
4. Features of ROS
5. ROS Installation & Development Environment
6. ROS terminology
7. Message communication
8. **ROS commands**
9. ROS tools
10. Communication between heterogeneous devices

ROS shell commands

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

ROS execution commands

Command	Importance	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+launch	Run multiple nodes with options
rosclean	★★☆	ros+clean	Check or delete ROS log files

ROS information commands

Command	Importance	Command description	Detailed description
rostopic	★★★	ros+topic	Check ROS topic information
rosservice	★★★	ros+service	Check ROS service information
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
rosversion	★☆☆	ros+version	Check ROS Package, Release Version Information
roswtf	☆☆☆	ros+wtf	ROS System Inspection

ROS catkin commands

Command	Importance	Detailed description
catkin_create_pkg	★★★	Check ROS topic information
catkin_make	★★★	Check ROS service information
catkin_eclipse	★★★	Check ROS node information
catkin_prepare_release	★★★	Check and modify ROS parameter information
catkin_generate_changelog	★★★	ROS message recording, playback
catkin_init_workspace	★★☆	Check ROS message information
catkin_find	★★☆	Check ROS service information

ROS package commands

Command	Importance	Command description	Detailed description
rospack	★★★	ros+pack(age)	Displays information related to the ROS package
rosinstall	★★☆	ros+install	Installs additional package for ROS
rosdep	★★☆	ros+dep(endencies)	Installs dependency files for the package
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)

1. About ROS
2. Purpose of ROS
3. Configuration of ROS
4. Features of ROS
5. ROS Installation & Development Environment
6. ROS terminology
7. Message communication
8. ROS commands
9. **ROS tools**
10. Communication between heterogeneous devices

ROS Tools (3D Visualization Tool (Rviz))

- **3D visualization tool for ROS**
  - Visualization of sensor data
  - Distance data of LASER distance sensor(LDS)
  - Point cloud data from depth camera such as ‘RealSense’, ‘Kinect’, ‘Xtion’, etc.
  - Image data of camera
  - Inertia data of IMU sensor ...
- **Represents robot configuration and planned motion**
  - URDF(Unified Robot Description Format)
- **Navigation**
- **Manipulation**
- **Tele-operation**

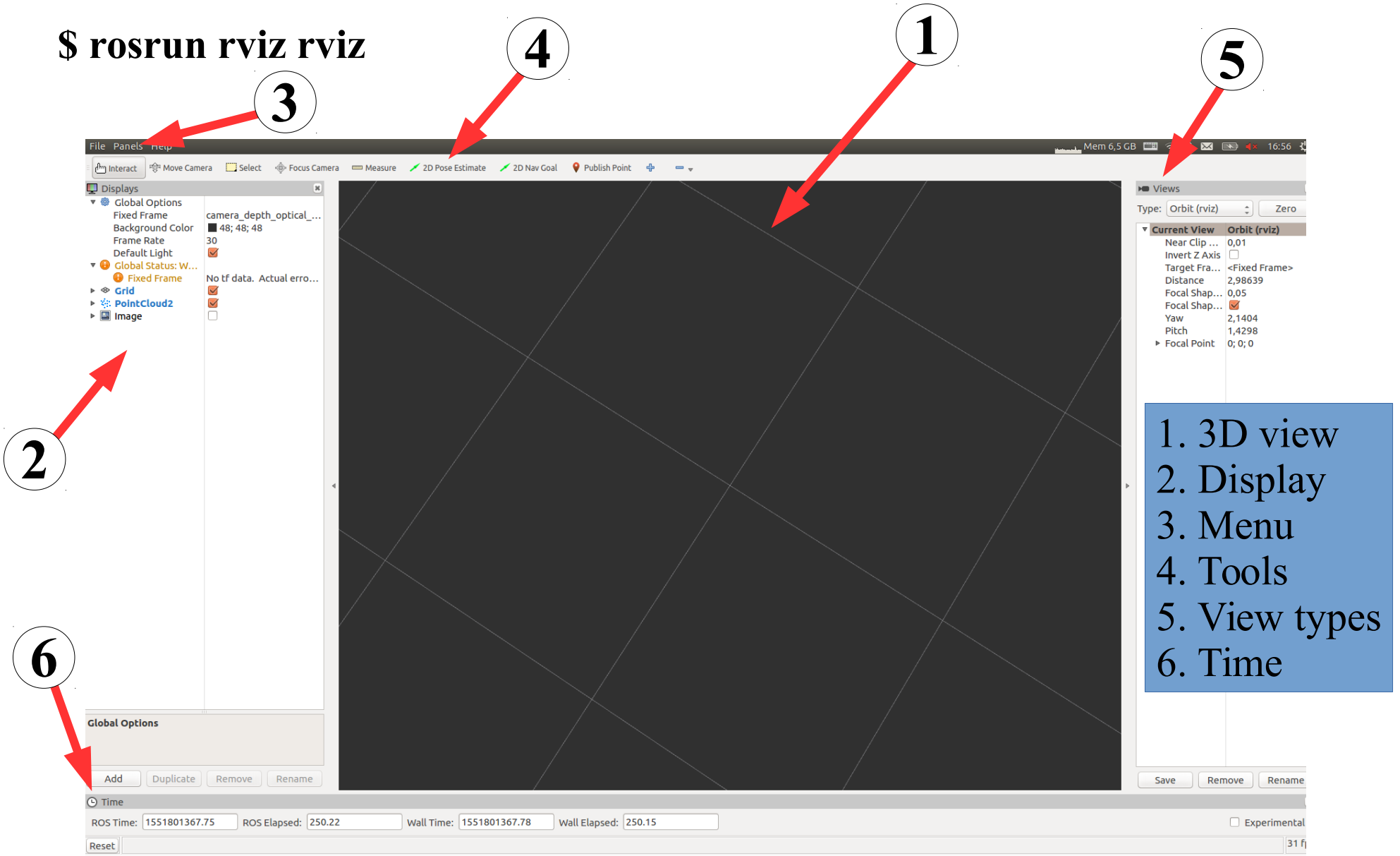
https://www.youtube.com/watch?v=ath8uNv9c_Q&feature=youtu.be

https://www.youtube.com/watch?v=9lbuLAD1c_4&feature=youtu.be

Rviz Test

1

$ rosrun rviz rviz



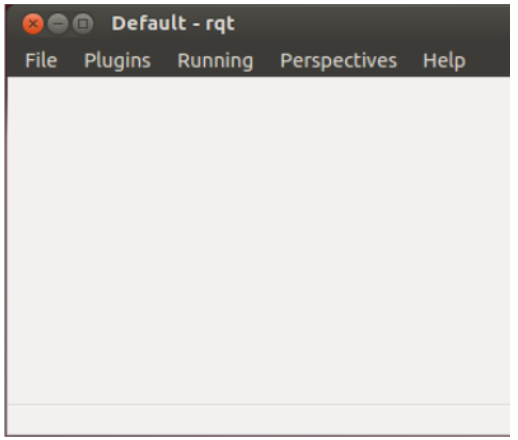
1. 3D view
2. Display
3. Menu
4. Tools
5. View types
6. Time

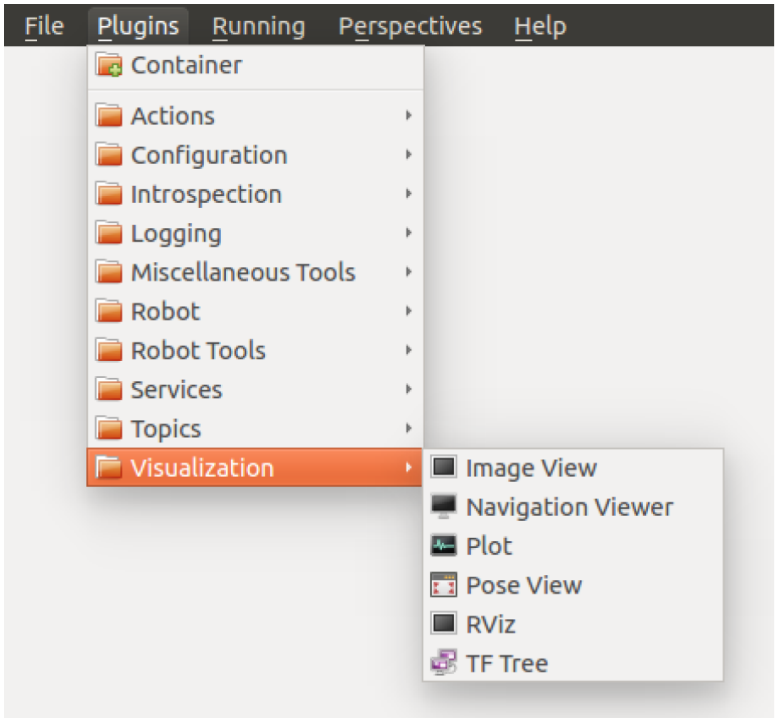
ROS Tools (GUI Tool Box: **RQT**)

- **RQT: Plug-in Type Comprehensive GUI Tool for ROS**

1. **Action**
2. **Configuration**
3. **Introspection**
4. **Logging**
5. **Miscellaneous Tools**
6. **Robot**
7. **Robot Tools**
8. **Services**
9. **Topics**
10. **Visualization**

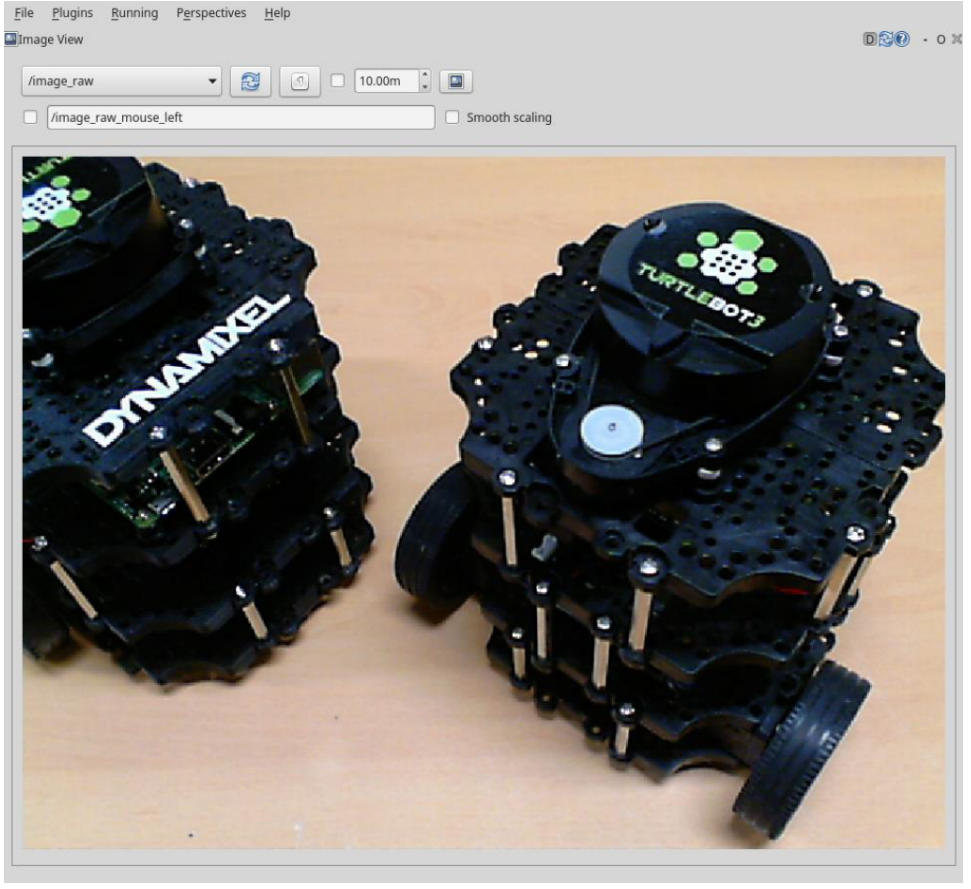
$ rqt





RQT Practice #1: rqt_image_view

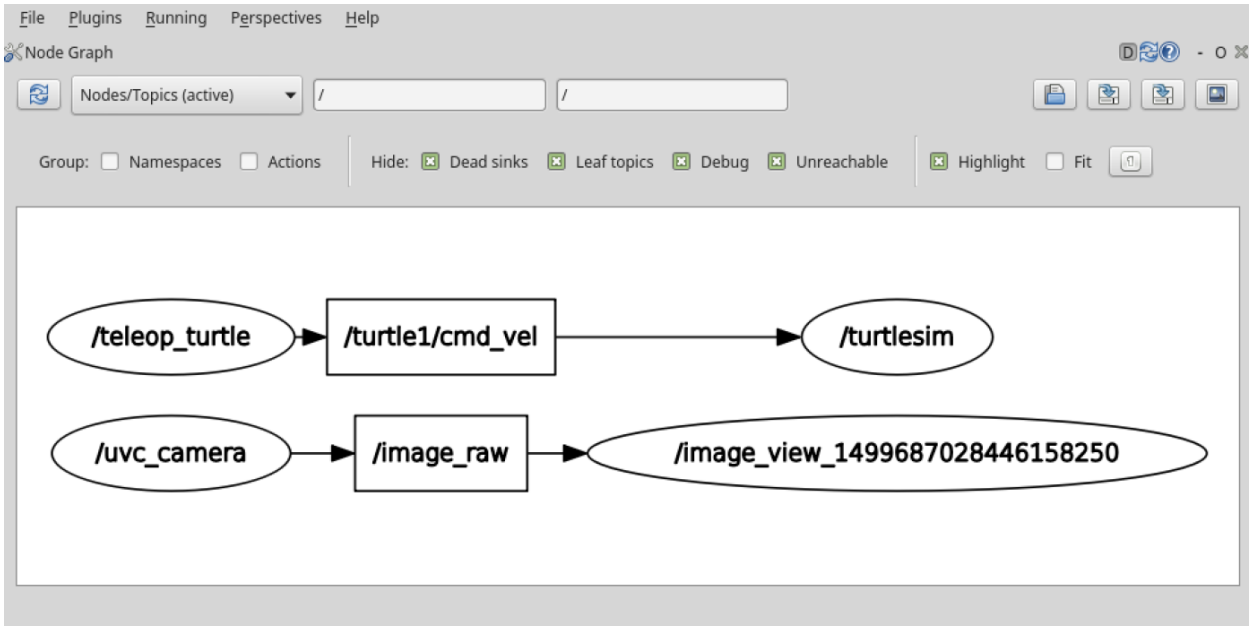
$ **rosrun uvc_camera uvc_camera_node**
$ **rqt**
(Select [Plugins] → [Visualization] → [Image View] in menu)
or
$ **rqt_image_view**



RQT Practice #2: rqt_graph

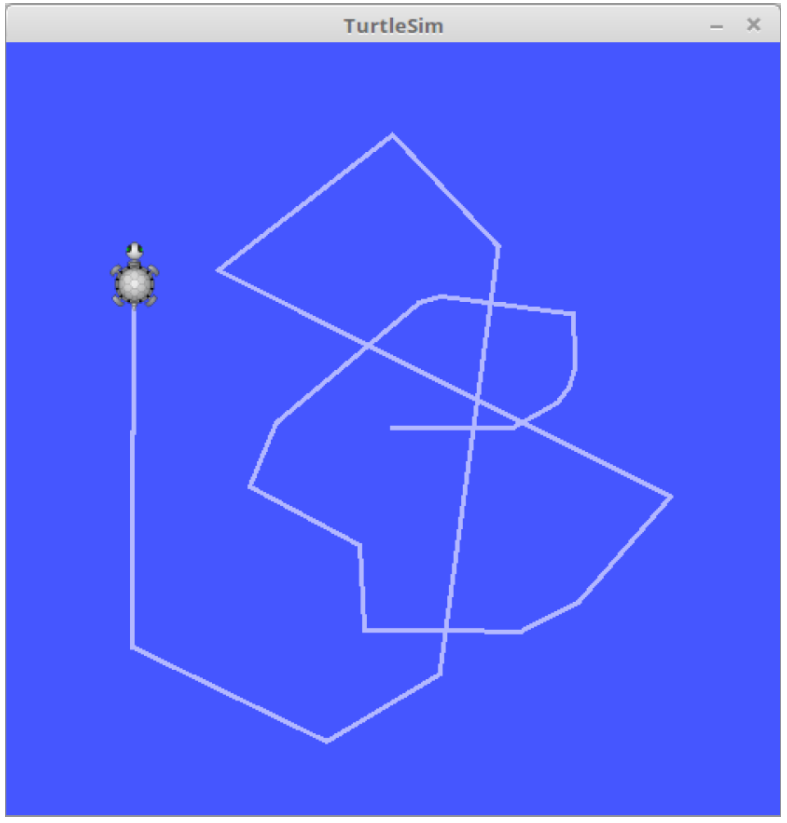
$ **rosrun turtlesim turtlesim_node**
$ **rosrun turtlesim turtle_teleop_key**
$ **rosrun uvc_camera uvc_camera_node**
$ **rosrun image_viewimage_viewimage:=image_raw**

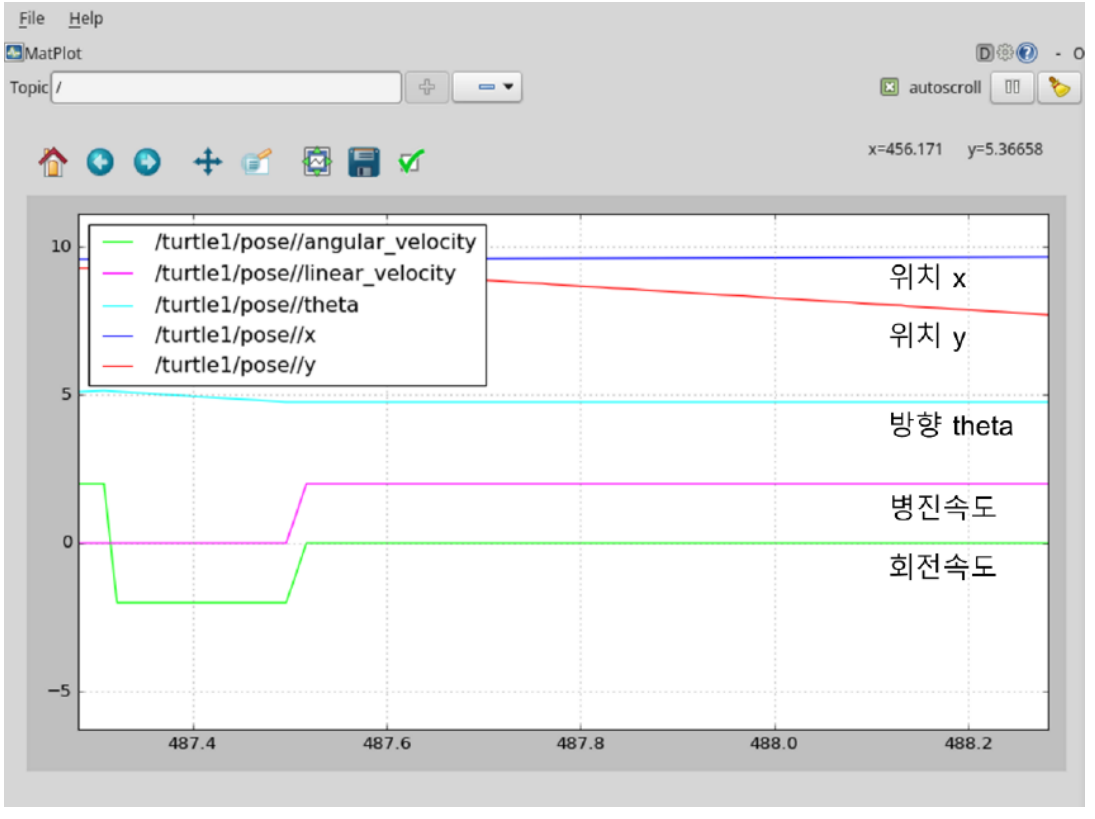
$ **rqt** (Select [Plugins] → [Introspection] → [Node_Graph]in menu)
Or
$ **rqt_graph**



RQT Practice #3: rqt_plot

$ **rosrun turtlesim turtlesim_node**
$ **rosrun turtlesim turtle_teleop_key**
$ **rqt** (Select [Plugins] → [Visualization] → [Plot]in menu)
or
$ **rqt_plot/turtle1/pose/**





RQT Practice #4: rqt_bag

$ **rosrun uvc_camerauvc_camera_node**
$ **rosbag record/image_raw**
$ **rqt** (Select [Plugins] → [Logging] → [Bag]in menu)
or
$ **rqt_bag**

Communication between heterogeneous devices

- Example 1: Transferring images remotely

- Example 2: Checking the acceleration value of your Android smartphone on your PC
https://play.google.com/store/apps/details?id=com.robotca.ControlApp&hl=ko&rdid=com.robotca.ControlApp

- Example 3: Controlling TurtleBot with Android Smartphone
https://play.google.com/store/apps/details?id=org.ros.android.sensors_driver&hl=en_US

1. About ROS
2. Purpose of ROS
3. Configuration of ROS
4. Features of ROS
5. ROS Installation & Development Environment
6. ROS terminology
7. Message communication
8. ROS commands
9. ROS tools
10. **Communication between heterogeneous devices**

Communication between heterogeneous devices

- Example 1: Transferring images remotely

- Example 2: Checking the acceleration value of your Android smartphone on your PC
https://play.google.com/store/apps/details?id=com.robotca.ControlApp&hl=ko&rdid=com.robotca.ControlApp

- Example 3: Controlling TurtleBot with Android Smartphone
https://play.google.com/store/apps/details?id=org.ros.android.sensors_driver&hl=en_US

Reference

- **Book**

  - ROS_Robot_Programming_EN
- http://wiki.ros.org/Documentation
- http://wiki.ros.org/ROS/Tutorials

Question Time!