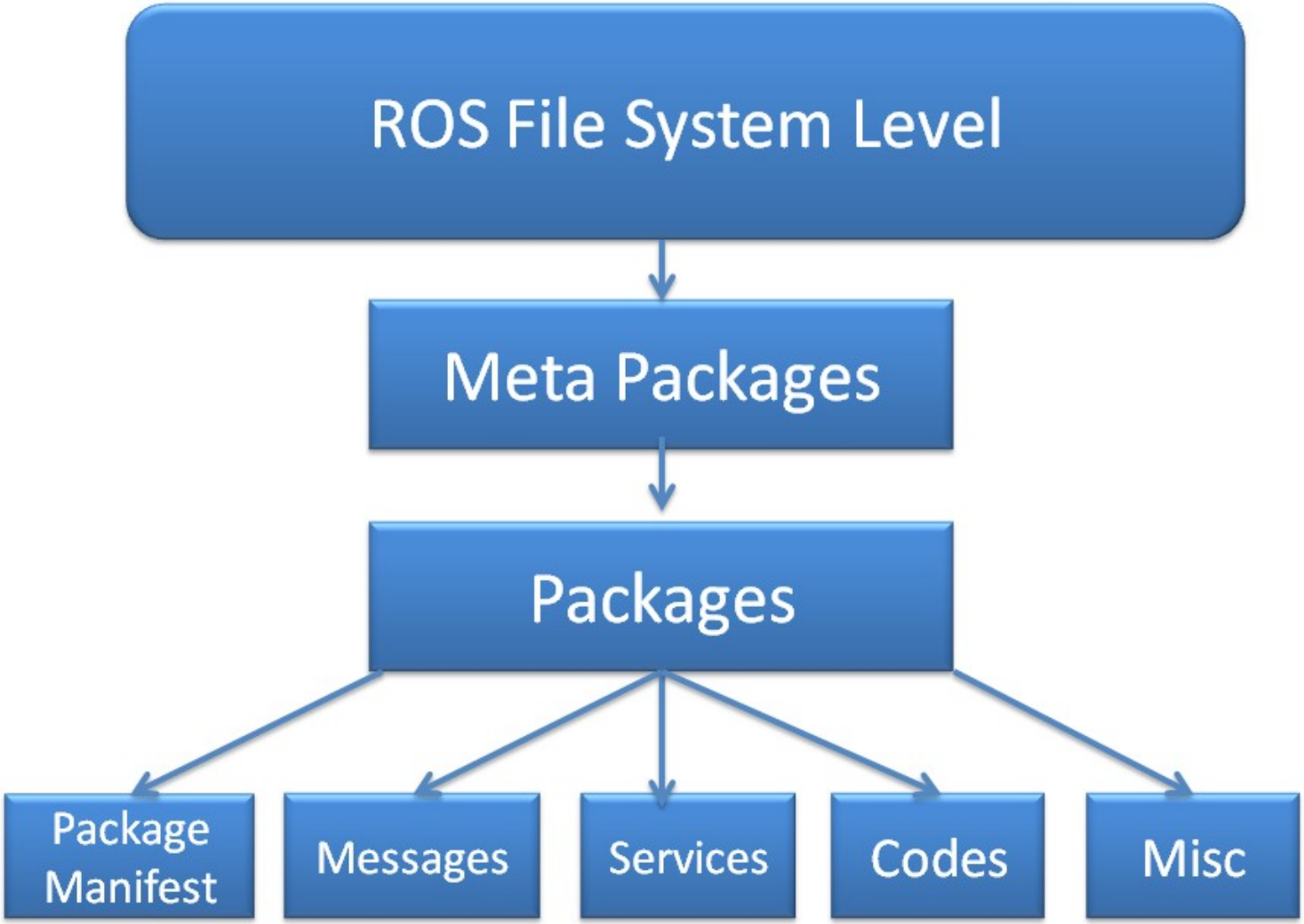
Robot Operating System

Chapter 1

ROS Basic

Pyae Soan Aung
RHCSA
Rom Robitics Co.ltd

ROS File System



Creating Workspace

~$ mkdir -p your_folder/catkin_ws/src

~$ cd your_folder/catkin_ws/src

~$ catkin_init_workspace

~$ cd ..
~$ catkin_make

~$ echo “source /home/username/your_folder/catkin_ws/devel/setup.bash”>> ~/.bashrc

or

~$ subl ~/.bashrc

```
if ! shopt -oq posix; then
  if [ -f /usr/share/bash-completion/bash_completion ]; then
    . /usr/share/bash-completion/bash_completion
  elif [ -f /etc/bash_completion ]; then
    . /etc/bash_completion
  fi
fi

source /opt/ros/noetic/setup.bash

source /home/ghostman/ros_course/catkin_ws/devel/setup.bash
source /home/ghostman/ros_driver/catkin_ws/devel/setup.bash
source /home/ghostman/aco/catkin_ws/devel/setup.bash
```

Package

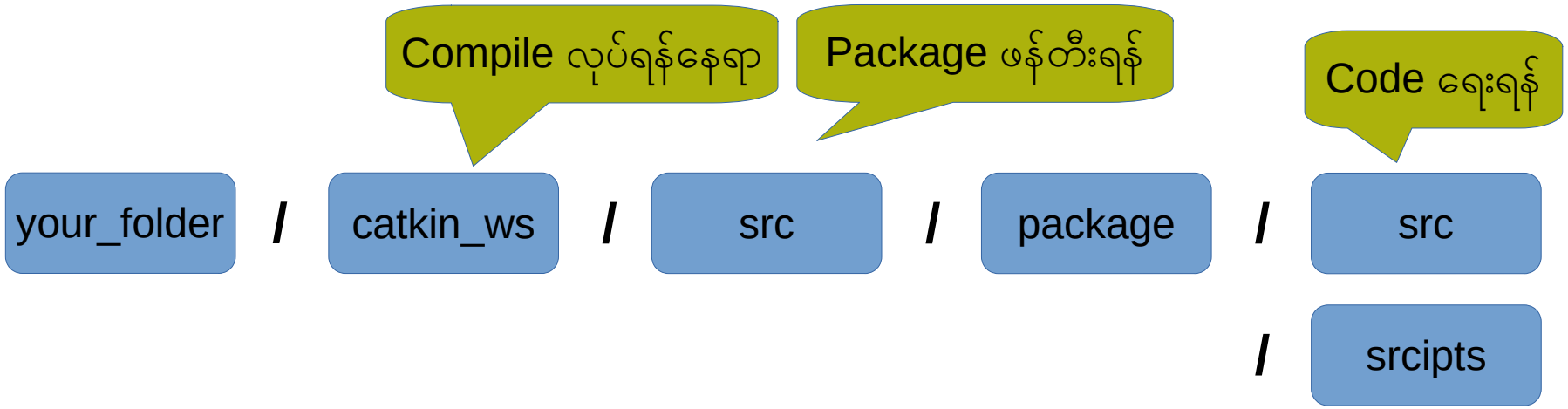
Creating package

```
~$ cd your_folder/catkin_ws/src

~$ catkin_create_pkg test roscpp rospy std_msgs

~$ cd ..

~$ catkin_make
```



Package.xml

```
<buildtool_depend>catkin</buildtool_depend>
<build_depend>roscpp</build_depend>
<build_depend>rospy</build_depend>
<build_depend>std_msgs</build_depend>
<build_export_depend>roscpp</build_export_depend>
<build_export_depend>rospy</build_export_depend>
<build_export_depend>std_msgs</build_export_depend>

<exec_depend>roscpp</exec_depend>
<exec_depend>rospy</exec_depend>
<exec_depend>std_msgs</exec_depend>
```

Command for ros package

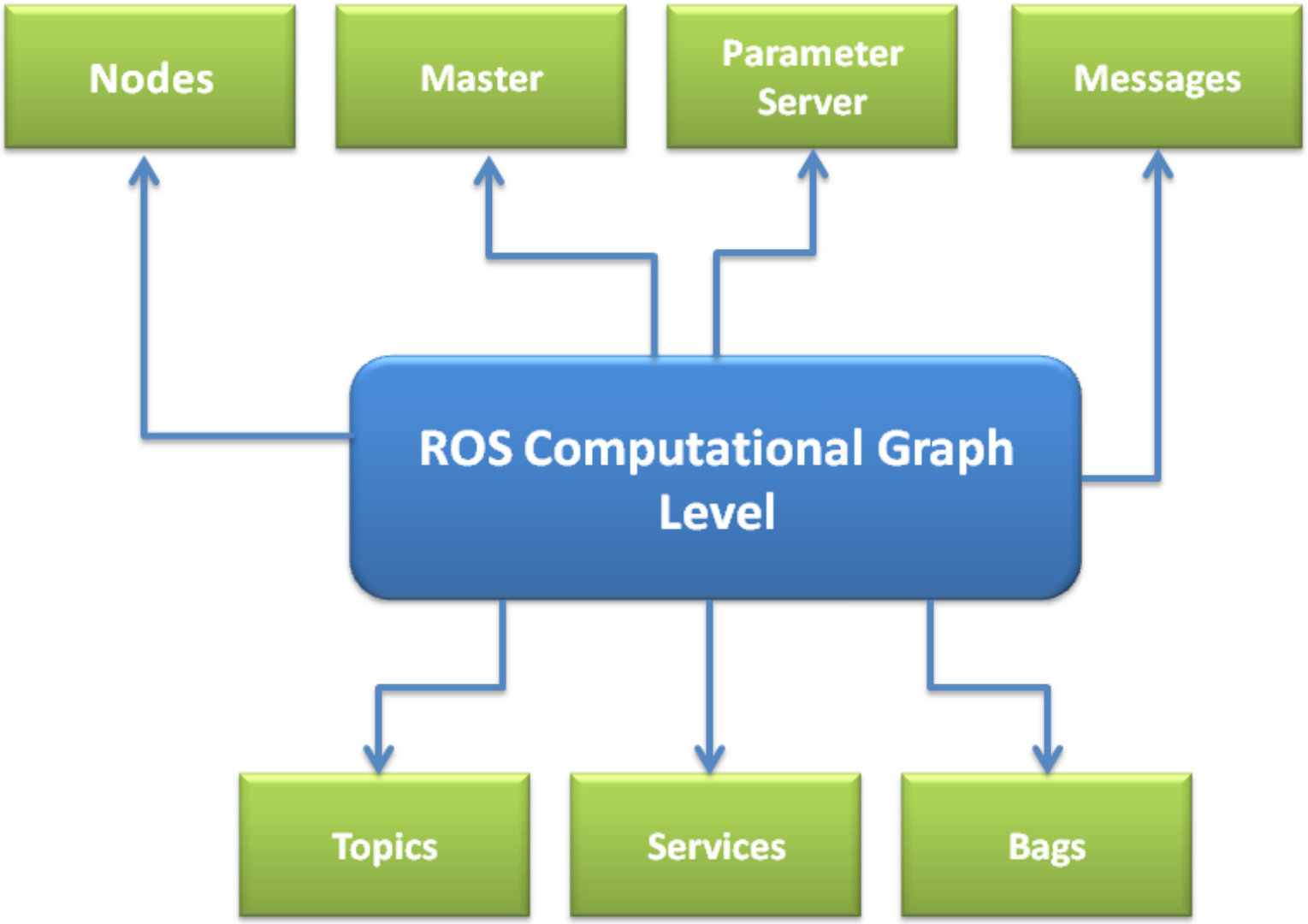
~$ rospack

~$ rospack list | grep turtlesim

~$ roscd package_name

~$ rospack find package_name

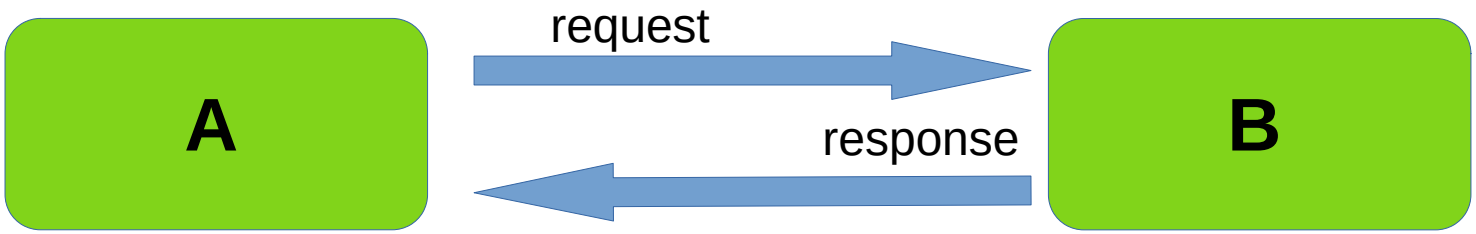
How ROS works?



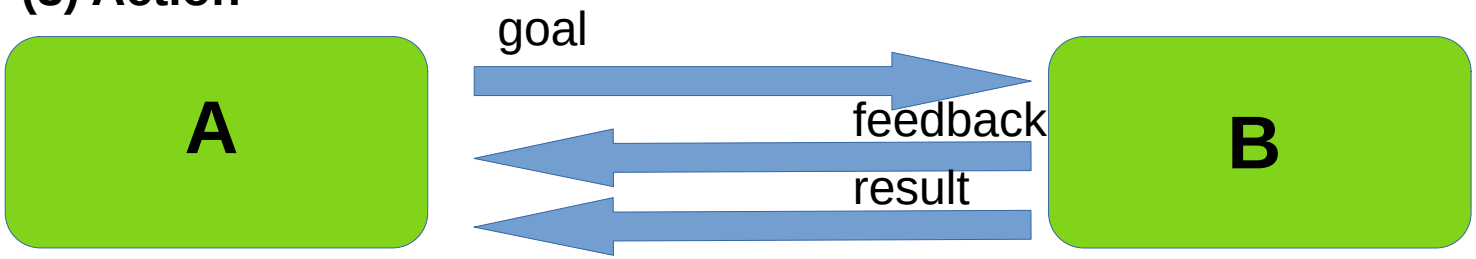
3 Communication methods

(1) topic

(2) Service

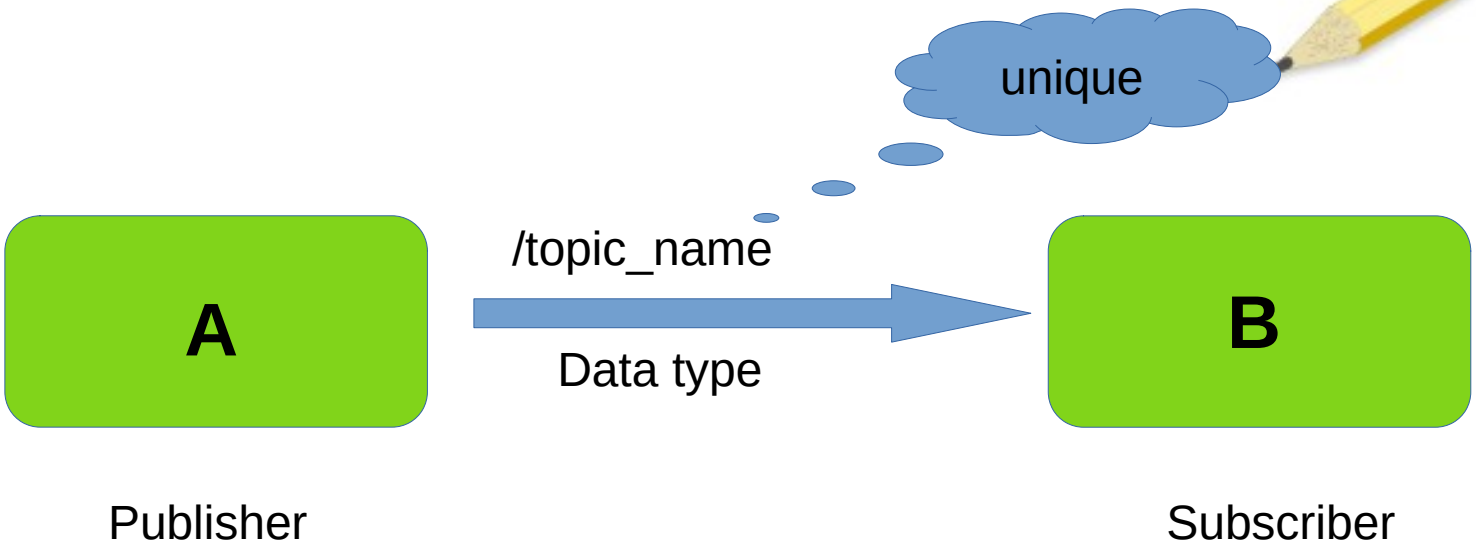


(3) Action



= node

Topic



Pubisher C++

```
1   #include "ros/ros.h"
2   #include "std_msgs/Int32.h"
3
4   int main(int argc, char ** argv)
5   {
6       ros::init(argc,argv, "publisher_node");
7       ros::NodeHandle n;
8       ros::Publisher pub = n.advertise<std_msgs::Int32>("message",1000);
9
10      ros::Rate r(10);
11
12      std_msgs::Int32 msg;
13
14      int x=0;
15
16      while(ros::ok())
17      {
18
19          msg.data = x;
20
21          ROS_INFO("i sent %d", msg.data);
22
23          pub.publish(msg);
24          x +=1;
25
26          ros::spinOnce();
27          r.sleep();
28
29      }
30      return 0;
31  }
```

Subscriber C++

```
#include "ros/ros.h"
#include "std_msgs/Int32.h"

void callback(const std_msgs::Int32 &msg)
{
    ROS_INFO("I Heard [%d]", msg.data);
}

int main(int argc, char ** argv)
{
    ros::init(argc, argv, "subscriber_node");
    ros::NodeHandle n;
    ros::Subscriber sub = n.subscribe("message",1000,callback);

    ros::spin();


}
```

CMakeLists.txt

```
add_executable(pub_node src/publisher.cpp)
add_dependencies(pub_node node_tuto_generate_messages_cpp)
target_link_libraries(pub_node ${catkin_LIBRARIES})

add_executable(sub_node src/subscriber.cpp)
add_dependencies(sub_node node_tuto_generate_messages_cpp)
target_link_libraries(sub_node ${catkin_LIBRARIES})
```

publisher.py

```python
#!/usr/bin/env python

import rospy
from std_msgs.msg import String

def talker():
    pub = rospy.Publisher('chatter', String,queue_size=10)
    rospy.init_node('talker', anonymous=True)

    rate = rospy.Rate(10)
    msg = String()
    while not rospy.is_shutdown():
        hello_str = "hello world %s" %rospy.get_time()
        rospy.loginfo(hello_str)
        msg.data = hello_str
        pub.publish(msg)
        rate.sleep()

if __name__ == '__main__':
    try:
        talker()
    except rospy.ROSInterruptException :
        pass
```

~$ sudo chmod a+x publisher.py

subscriber.py

```python
#!/usr/bin/env python

import rospy
from std_msgs.msg import String

def callback(msg):
    rospy.loginfo("I Heard %s", msg.data)

def listener():

    rospy.init_node('listener', anonymous=True)

    rospy.Subscriber('chatter',String, callback)

    rospy.spin()

if __name__ == '__main__':
    listener()
```

~$ sudo chmod a+x subscriber.py

How to run ros node?

```
~$ roscore                                  # running Master
~$ rosrun    packageName    executalbeName
```

#Checking Node

```
~$ rosnode   list
~$ rosnode   info   /node_name
```

#Checking topic

```
~$ rostopic   list
~$ rostopic   info    /topic_name
~$ rostopic   echo   /topic_name
~$ rostopic   pub    /topic_name    dataType    value
```

#Checking msg

```
~$ rosmsg
~$ rosmsg   list
~$ rosmsg   show   dataType
```

Message အဖြစ်အသုံးပြုတဲ့
Header ဖိုင်တွေ
ဘယ်ကလာသလဲ?

Launch

~$ roslaunch package_name filename.launch

```
<launch>
    <node pkg="node_tuto" name="y" type="pub_node"/>
    <node pkg="node_tuto" name="x" type="sub_node"/>
</launch>
```

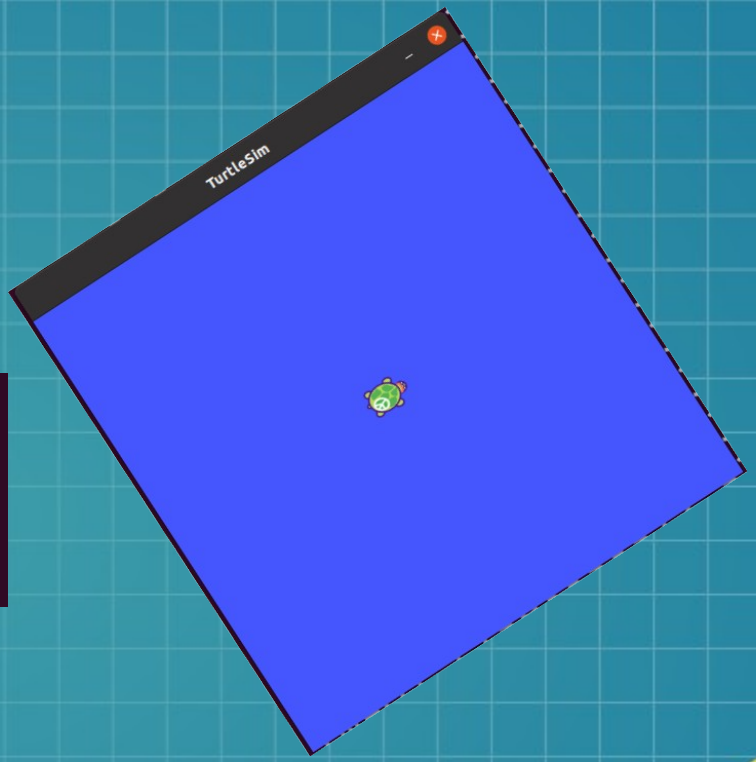
For more tags, see roswiki

```
<launch>
    <node pkg=”blabal”  type=”blabla” name=”blabla” >
        <remap from=”current_topic_name to=”new_topic_name>
    </node>


    <args name=”use_gui”  default=”false”   />
    <param name=”number_of_camera”   value=”3”/>
    <rosparam .../>
<launch>
```

Turtle Simulation

```
phyo@phyo:~$ rosrun turtlesim turtle_teleop_key
Reading from keyboard
---------------------------
Use arrow keys to move the turtle. 'q' to quit.
```



~$ roscore

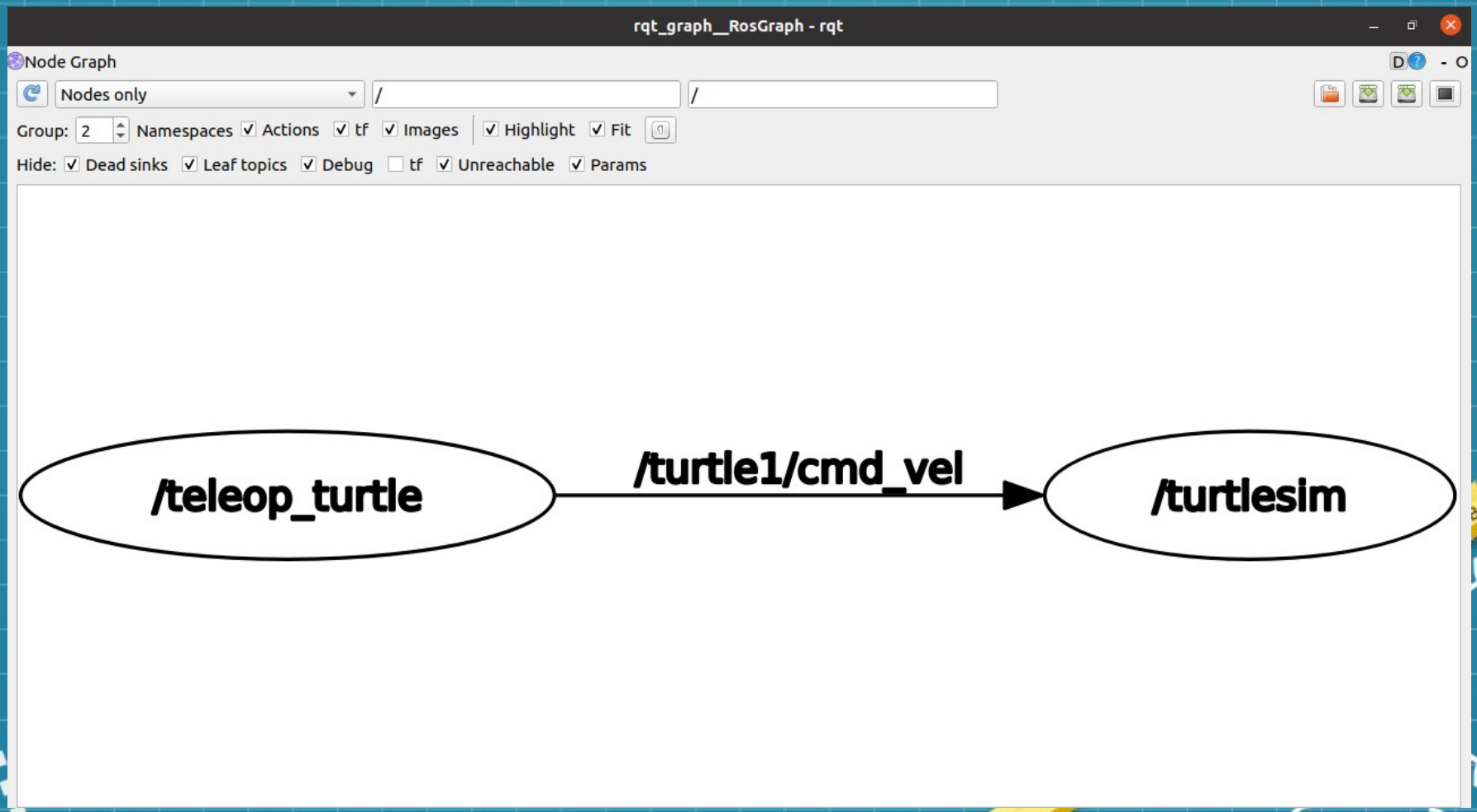
~$ rosrun turtlesim turtlesim_node

~$ rosrun turtlesim turtle_teleop_key

1 meter square?

Node Graph

~$ rosrun rqt_graph rqt_graph



What is namespace?

- publisher.cpp မှာ topic ကို message ဆိုပြီးကြေညာခဲ့တယ်။
- ros::Publisher pub = n.advertise<std_msgs::Int32>("message",1000);
- အဲ့မှာ message က သေချာ ကြေညာခြင်းမဟုတ်။ default ပါ။ relative ဖြစ်နေပါတယ်။ Relative ဆိုတာ သက်ဆိုင်ရာ node မှာရှိနေတာပါ။
- global namespace အဖြစ် “/message” လို့ရေးလိုက်ရင် ဘယ် subscriber ကနေမဆို အလွယ်တကူ ယူသုံးလို့ရမှာဖြစ်ပါတယ်။
- ros::Publisher pub = n.advertise<std_msgs::Int32>("/message",1000);
- နောက်ထပ် “message” ဆိုတဲ့ နာမည်နဲ့ထုတ်ချင်တဲ့အခါ “nodeA/message, nodeB/message” ဆိုပြီး node name နဲ့ သက်ဆိုင်ရာ message topic ပေါင်းခြင်းဖြင့် နာမည်တူ topic တွေကို သုံးလို့ရပါမယ်။
- Local namespace ဖြစ်ချင်ရင် “/” ကိုဖြုတ်ပါ။ ros:NodeHandle n(“~”); လို့ NodeHandle ကိုပြင်ပါ။
- Topic, node, param, service, action အားလုံးကိုသက်ရောက်ပါတယ်။

```
phyo@phyo:~$ rostopic list
/msg
/rosout
/rosout_agg
```

```
phyo@phyo:~$ rostopic list
/aNode/msg
/rosout
/rosout_agg
```

Namespace ခွဲသုံးခြင်းအားဖြင့် system မှာ node တွေ message တွေ topicတွေ အများကြီး ခွဲသုံးနိုင်မှာဖြစ်ပါတယ်။

ROS PARAMETER SERVER

This comes with ROS MASTER!

~$ rosparam list
~$ rosparam set /param_name /value
~$ rosparam get /param_name

~$ rosparam dump myparam.yaml
~$ rosparam load myparam.yaml

```
std::string global_name, relative_name, default_param;
if (ros::param::get("/global_name", global_name))
{
  ...
}

if (ros::param::get("relative_name", relative_name))
{
...
}
```

```
ros::param::set("/global_param", 5);
ros::param::set("relative_param", "my_string");
ros::param::set("bool_param", false);
```

```
ros::NodeHandle nh;
nh.setParam("/global_param", 5);
nh.setParam("relative_param", "my_string");
nh.setParam("bool_param", false);
```

```
# Using rospy and raw python objects
rospy.set_param('a_string', 'baz')
rospy.set_param('~private_int', 2)
rospy.set_param('list_of_floats', [1., 2., 3., 4.])
rospy.set_param('bool_True', True)
rospy.set_param('gains', {'p': 1, 'i': 2, 'd': 3})

# Using rosparam and yaml strings
rosparam.set_param('a_string', 'baz')
rosparam.set_param('~private_int', '2')
rosparam.set_param('list_of_floats', "[1., 2., 3., 4.]")
rosparam.set_param('bool_True', "true")
rosparam.set_param('gains', "{'p': 1, 'i': 2, 'd': 3}")

rospy.get_param('gains/p') #should return 1
```

Thank you!!!

rom robotics