ROS-Industrial Basic Developer's Training Class

Southwest Research Institute

Session 1:

ROS Basics

Southwest Research Institute

Outline

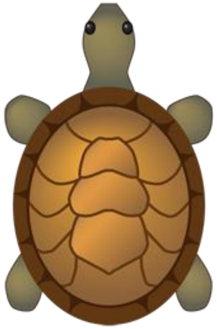
- Intro to ROS
- Catkin (Create workspace)
- Installing packages (existing)
- Packages (create)
- Nodes
- Messages / Topics

An Introduction to ROS

(Image taken from Willow Garage's "What is ROS?" presentation)

ROS Versions

Annual releases ("distribution")

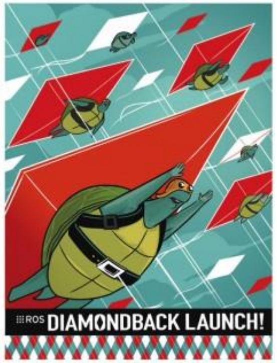


Box Turtle

Mar 2010

C Turtle

Aug 2010

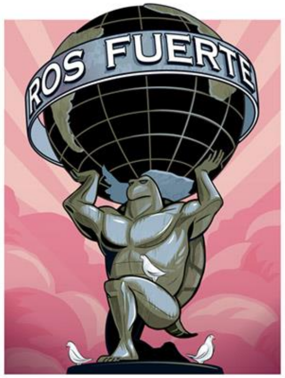


Diamondback

Mar 2011

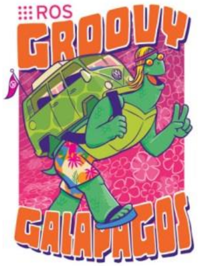
Electric

Aug 2011



Fuerte

April 2012



Groovy

2012 - 2014

Hydro

2013 - 2015

Indigo

2014 - 2019

Jade

2015 - 2017

Kinetic

2016 - 2021

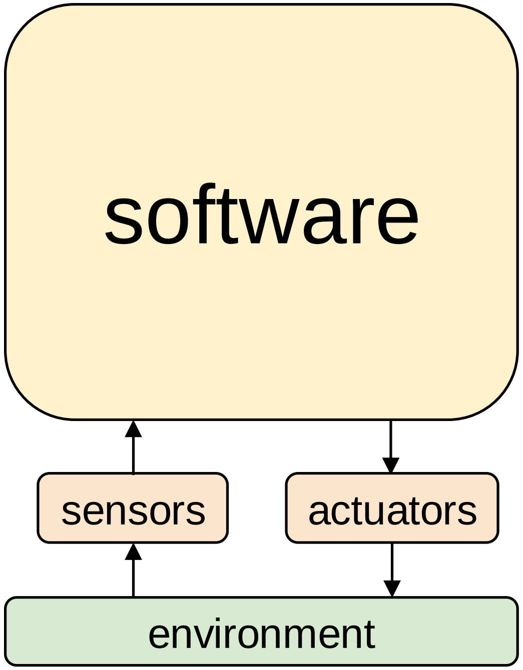
Lunar

2017 - 2019

Melodic

2018 - 2023

ROS : The Big Picture

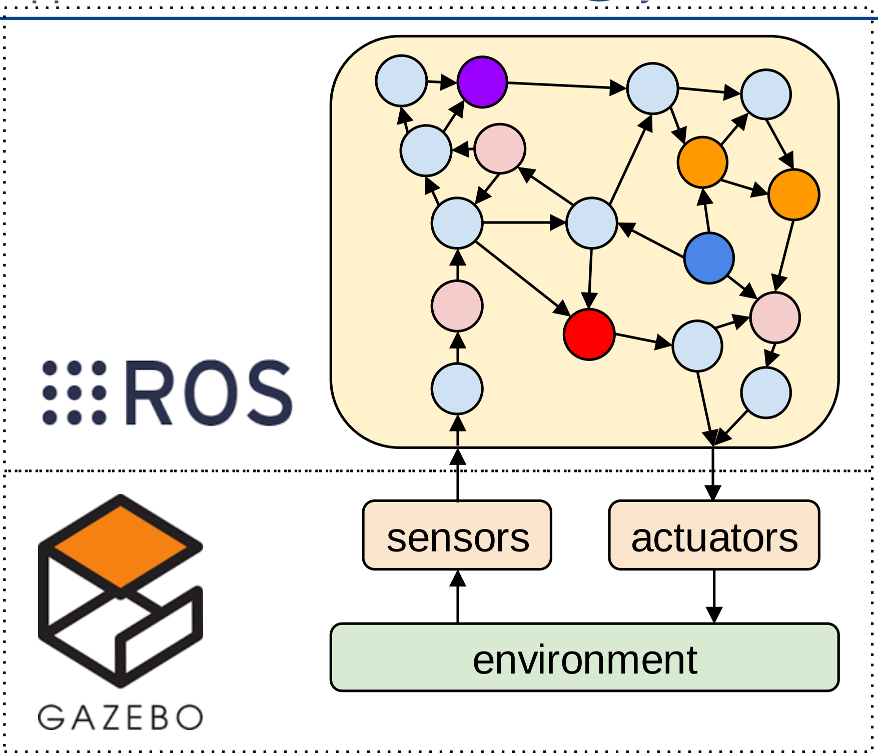


All robots are:

Software connecting Sensors to Actuators to interact with the Environment

(Adapted from Morgan Quigley's "ROS: An Open-Source Framework for Modern Robotics" presentation)

ROS : The Big Picture

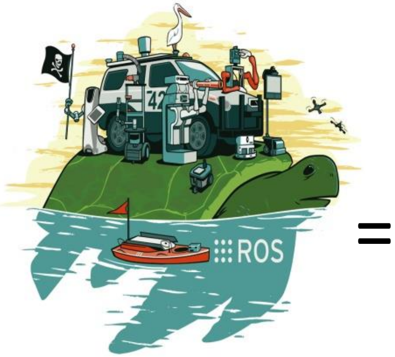


- Break Complex Software into Smaller Pieces
- Provide a framework, tools, and interfaces for distributed development
- Encourage re-use of software pieces
- Easy transition between simulation and hardware

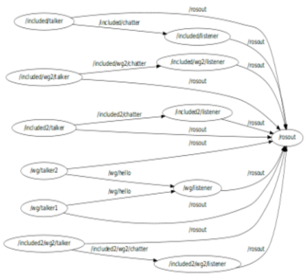
(Adapted from Morgan Quigley's "ROS: An Open-Source Framework for Modern Robotics" presentation)

What is ROS?

ROS is…

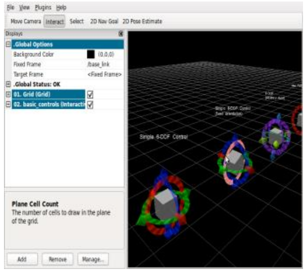


=



Plumbing

+



.Global Options
Background Color (0,0,0)
Fixed Frame /base_link
Target Frame <Fixed Frame>
.Global Status: OK
01. Grid (Grid) ☑
02. basic_controls (Interacti ☑

Plane Cell Count
The number of cells to draw in the plane of the grid.

Tools

+

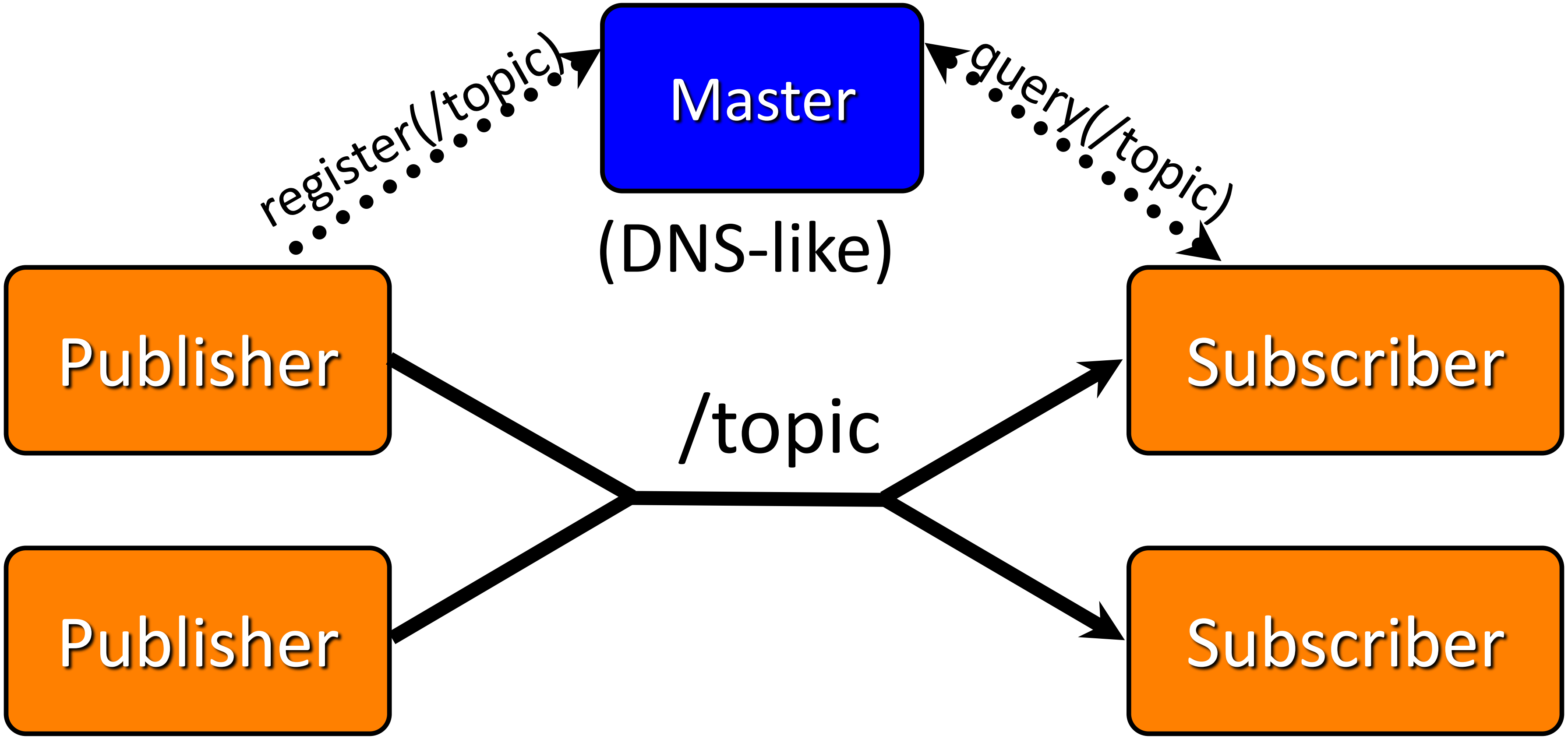
Capabilities

+

Ecosystem

(Adapted from Willow Garage's "What is ROS?" Presentation)

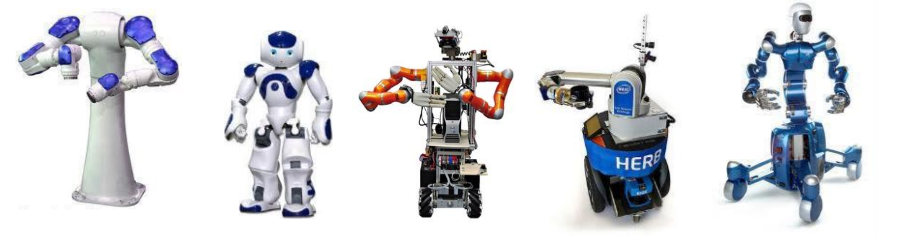
ROS is… plumbing



(Adapted from Willow Garage's "What is ROS?" Presentation)

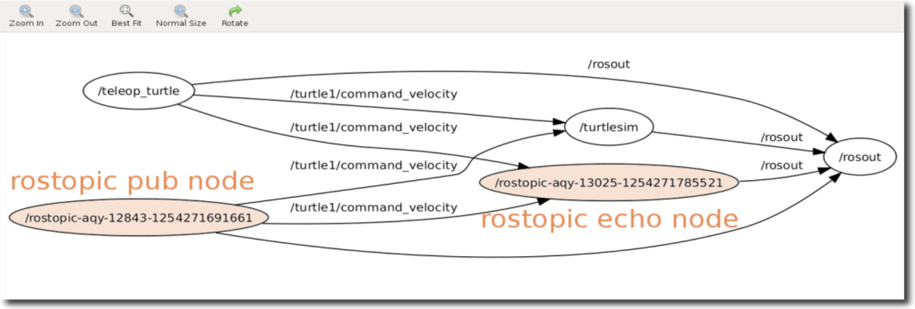
ROS Plumbing : Drivers

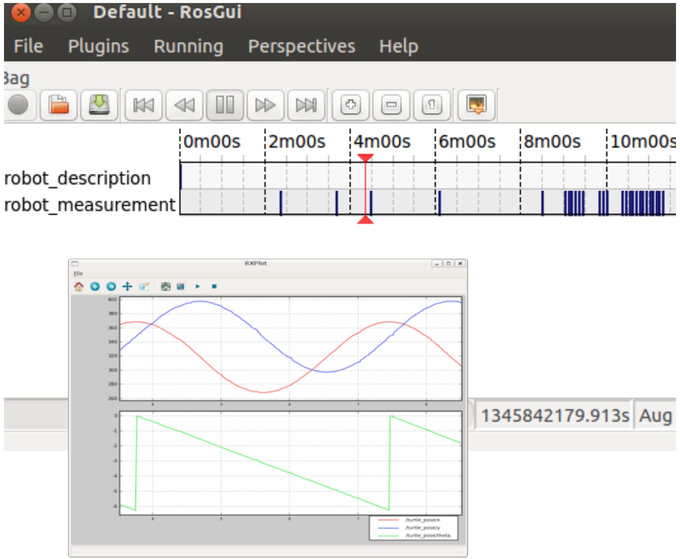
- 2d/3d cameras
- laser scanners
- robot actuators
- inertial units
- audio
- GPS
- joysticks
- etc.



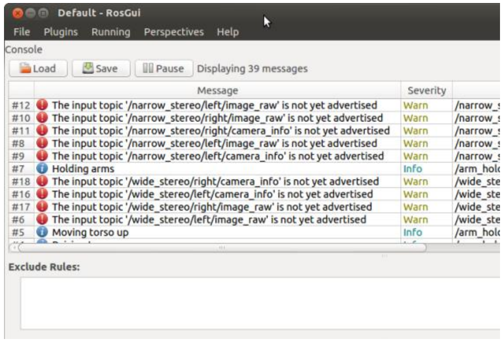
(Adapted from Morgan Quigley's "ROS: An Open-Source Framework for Modern Robotics" presentation)

ROS is …Tools


Zoom In
Zoom Out
Best Fit
Normal Size
Rotate
/teleop_turtle
/rosout
/turtle1/command_velocity
/turtle1/command_velocity
/turtlesim
/rosout
/turtle1/command_velocity
/rosout
/rosout
rostopic pub node
/rostopic-aqy-13025-1254271785521
/rostopic-aqy-12843-1254271691661
/turtle1/command_velocity
rostopic echo node



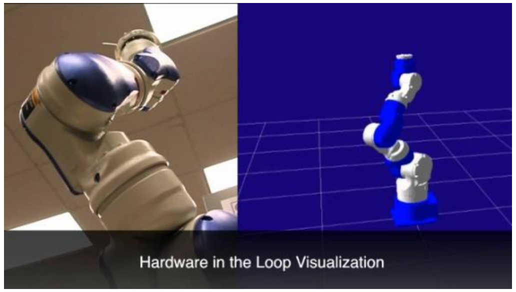
- logging/plotting
- graph visualization
- diagnostics
- visualization



Displaying 39 messages

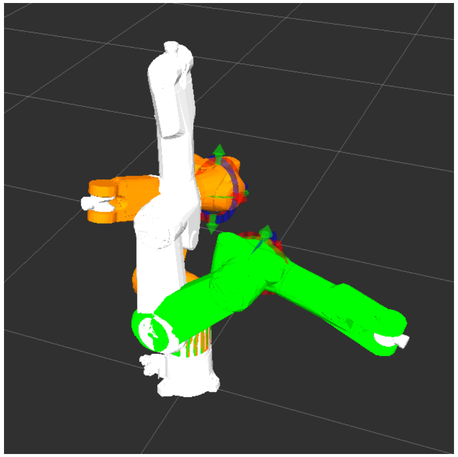
	Message	Severity	
#12	The input topic '/narrow_stereo/left/image_raw' is not yet advertised	Warn	/narrow_s
#10	The input topic '/narrow_stereo/right/image_raw' is not yet advertised	Warn	/narrow_s
#11	The input topic '/narrow_stereo/right/camera_info' is not yet advertised	Warn	/narrow_s
#8	The input topic '/narrow_stereo/left/image_raw' is not yet advertised	Warn	/narrow_s
#9	The input topic '/narrow_stereo/left/camera_info' is not yet advertised	Warn	/narrow_s
#7	Holding arms	Info	/arm_holc
#18	The input topic '/wide_stereo/right/camera_info' is not yet advertised	Warn	/wide_ste
#16	The input topic '/wide_stereo/left/camera_info' is not yet advertised	Warn	/wide_ste
#17	The input topic '/wide_stereo/right/image_raw' is not yet advertised	Warn	/wide_ste
#6	The input topic '/wide_stereo/left/image_raw' is not yet advertised	Warn	/wide_ste
#5	Moving torso up	Info	/arm_holc

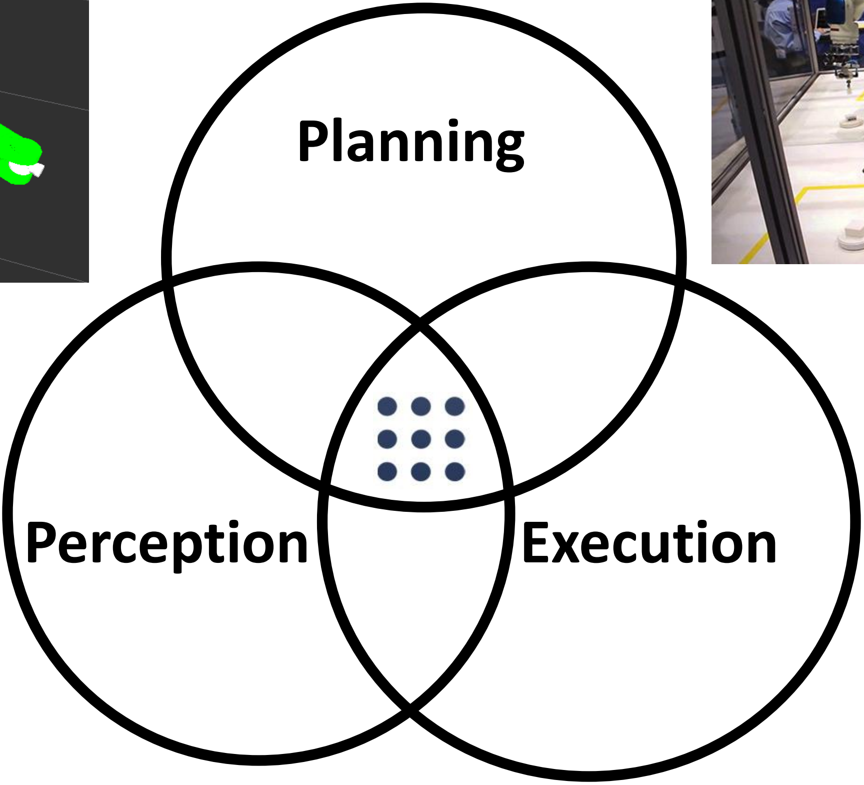
Exclude Rules:



(Adapted from Willow Garage's "What is ROS?" Presentation)

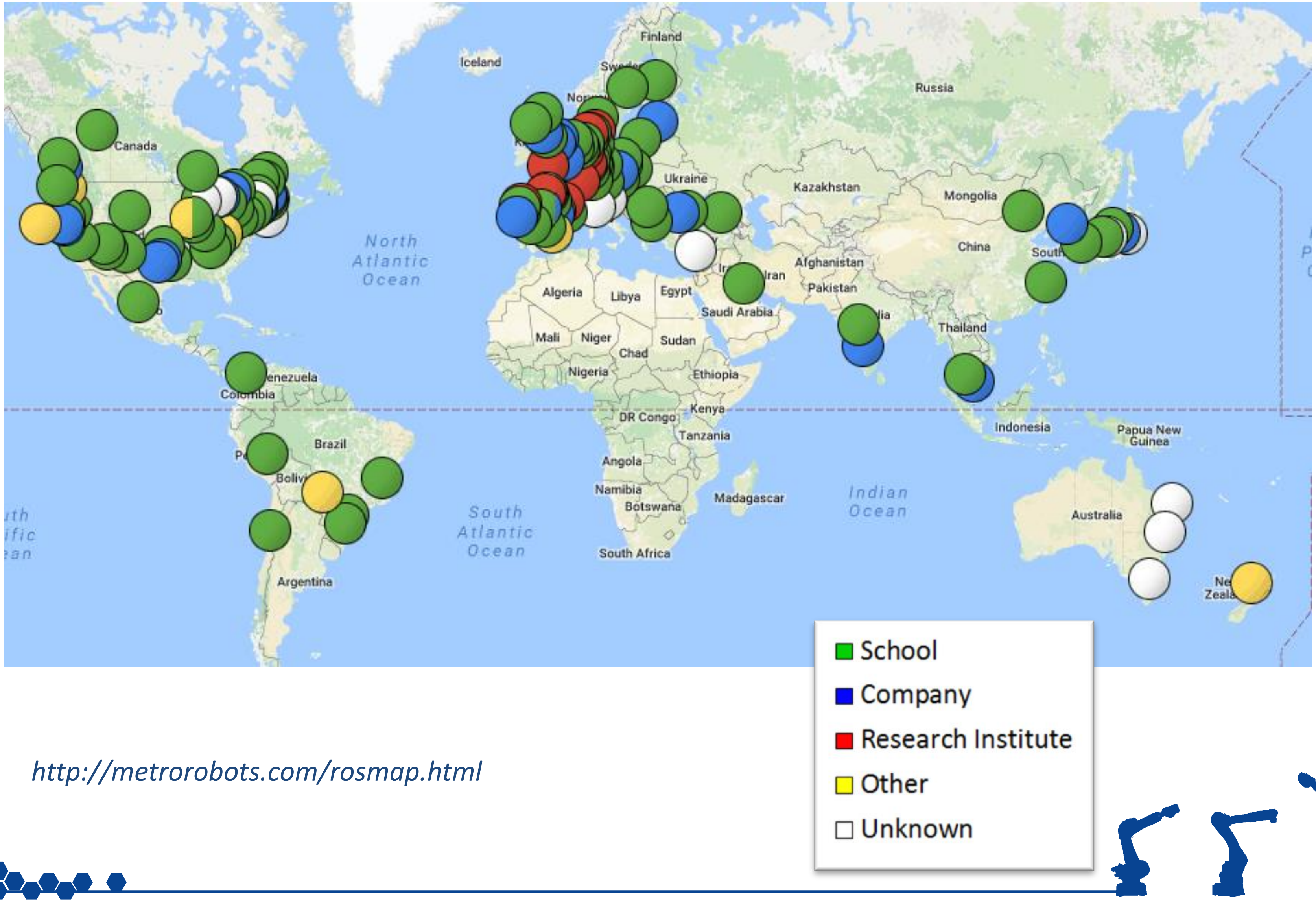
ROS is…Capabilities





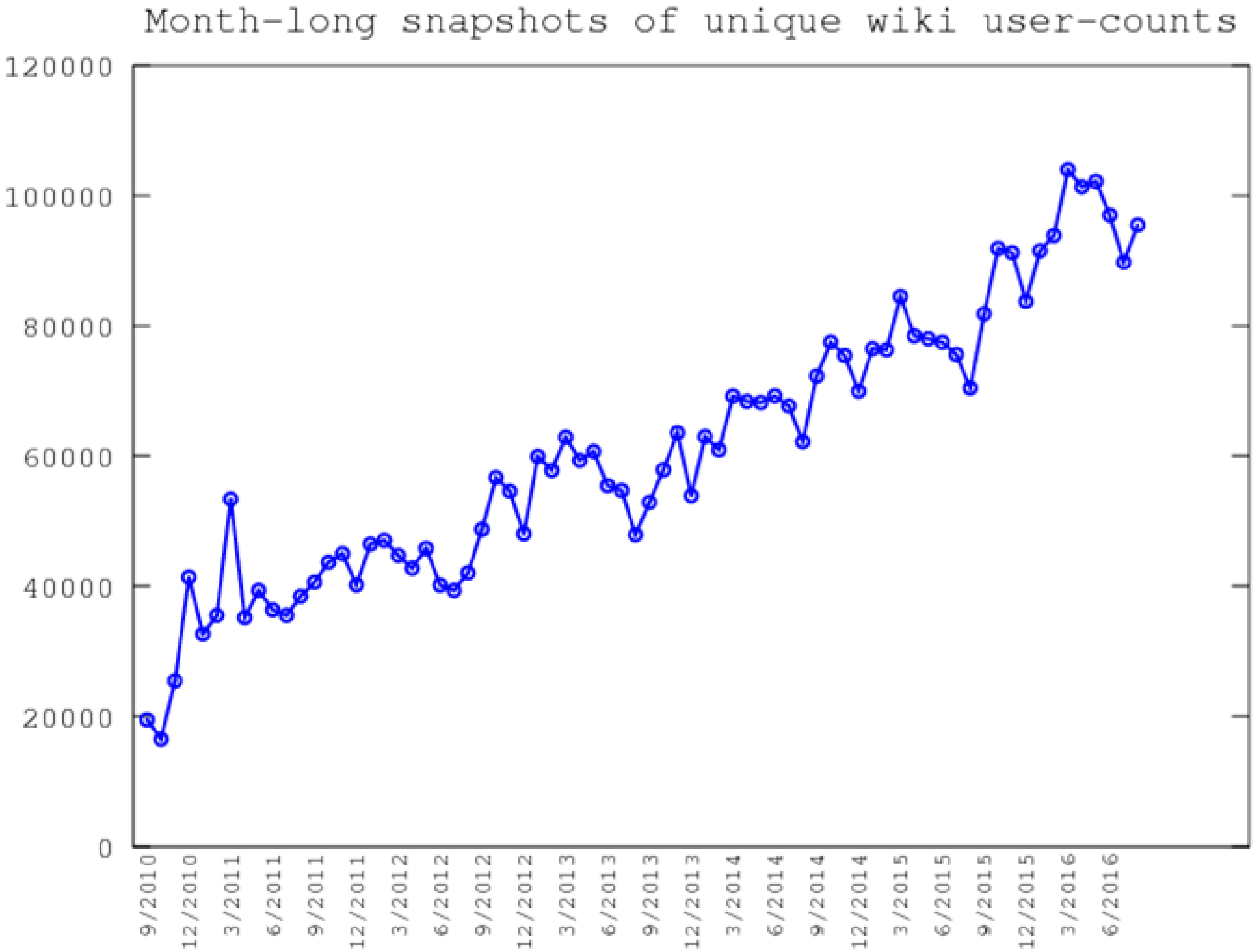
(Adapted from Willow Garage's "What is ROS?" Presentation)

ROS is… an Ecosystem



http://metrorobots.com/rosmap.html

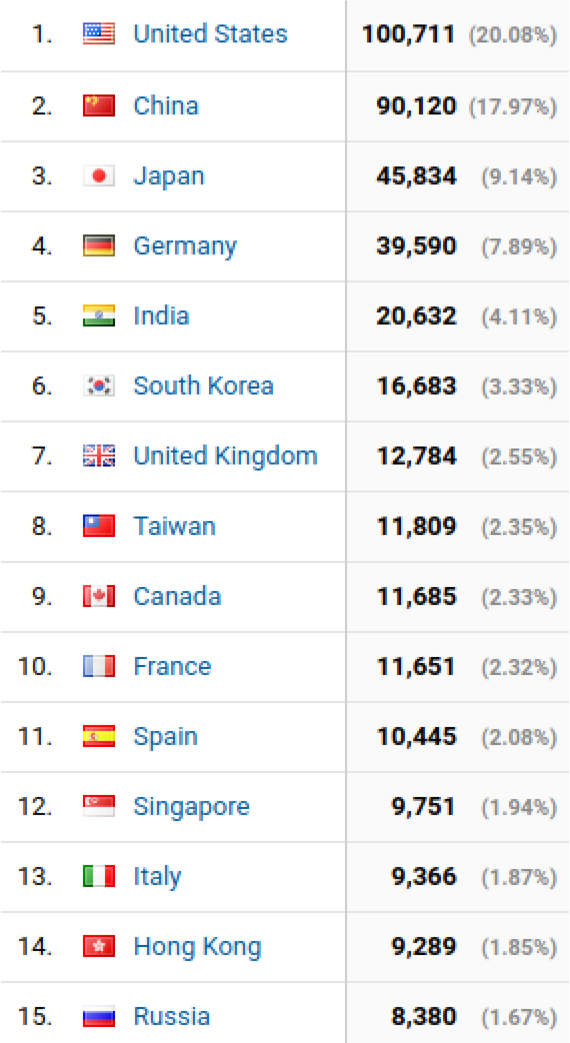
ROS is a growing Ecosystem



(From Morgan Quigley's "ROS: An Open-Source Framework for Modern Robotics" presentation)

ROS is International

unique wiki visitors July 2017



1.	United States	100,711	(20.08%)
2.	China	90,120	(17.97%)
3.	Japan	45,834	(9.14%)
4.	Germany	39,590	(7.89%)
5.	India	20,632	(4.11%)
6.	South Korea	16,683	(3.33%)
7.	United Kingdom	12,784	(2.55%)
8.	Taiwan	11,809	(2.35%)
9.	Canada	11,685	(2.33%)
10.	France	11,651	(2.32%)
11.	Spain	10,445	(2.08%)
12.	Singapore	9,751	(1.94%)
13.	Italy	9,366	(1.87%)
14.	Hong Kong	9,289	(1.85%)
15.	Russia	8,380	(1.67%)

http://wiki.ros.org/Metrics

visitors per million people

1. Singapore: 1711
2. Hong Kong: 1255
3. Switzerland: 526
4. Taiwan: 500
5. Germany: 482

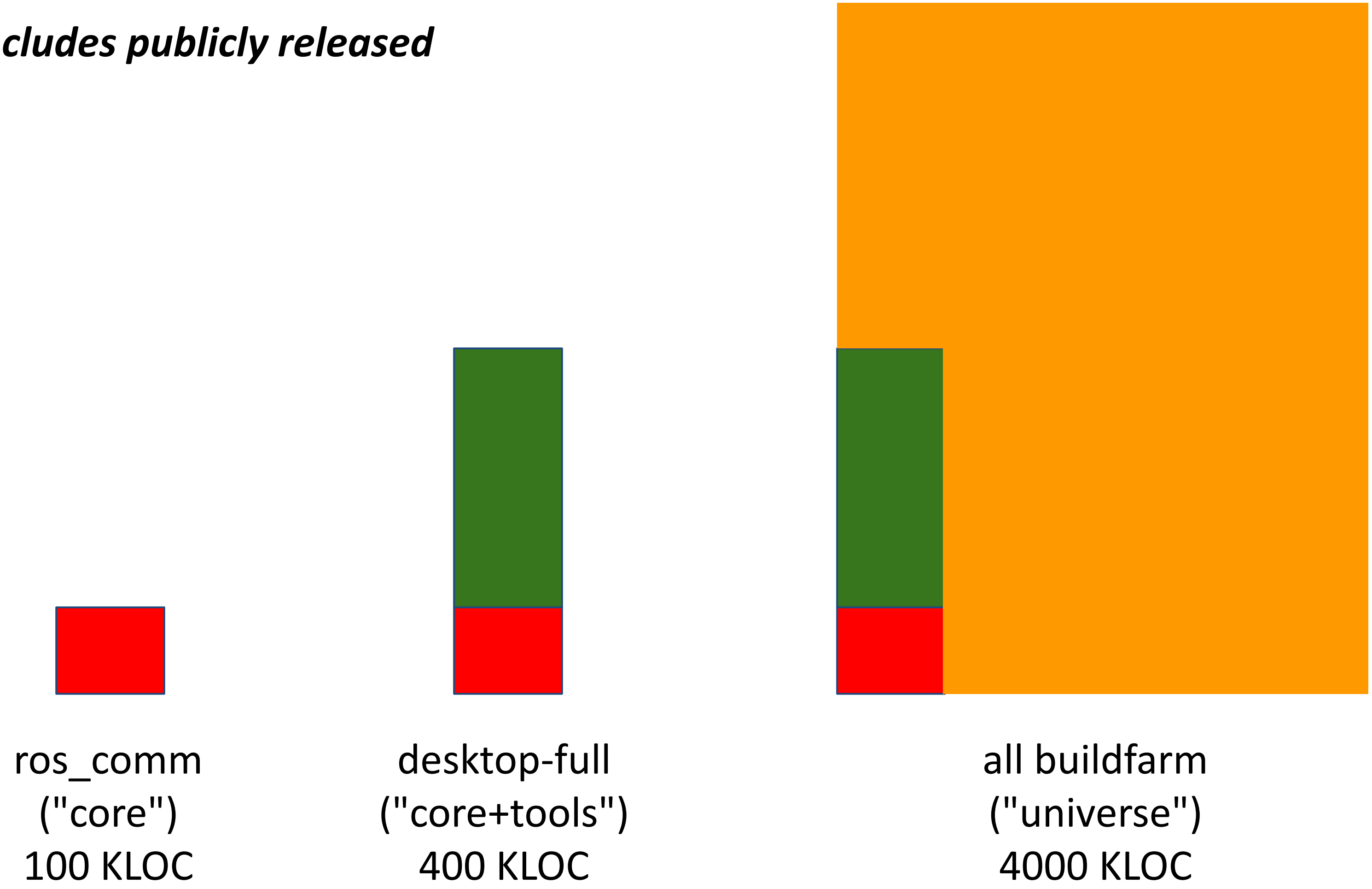
…

10. USA: 310

Does not include visitors to wiki mirrors (Singapore, China, ...)

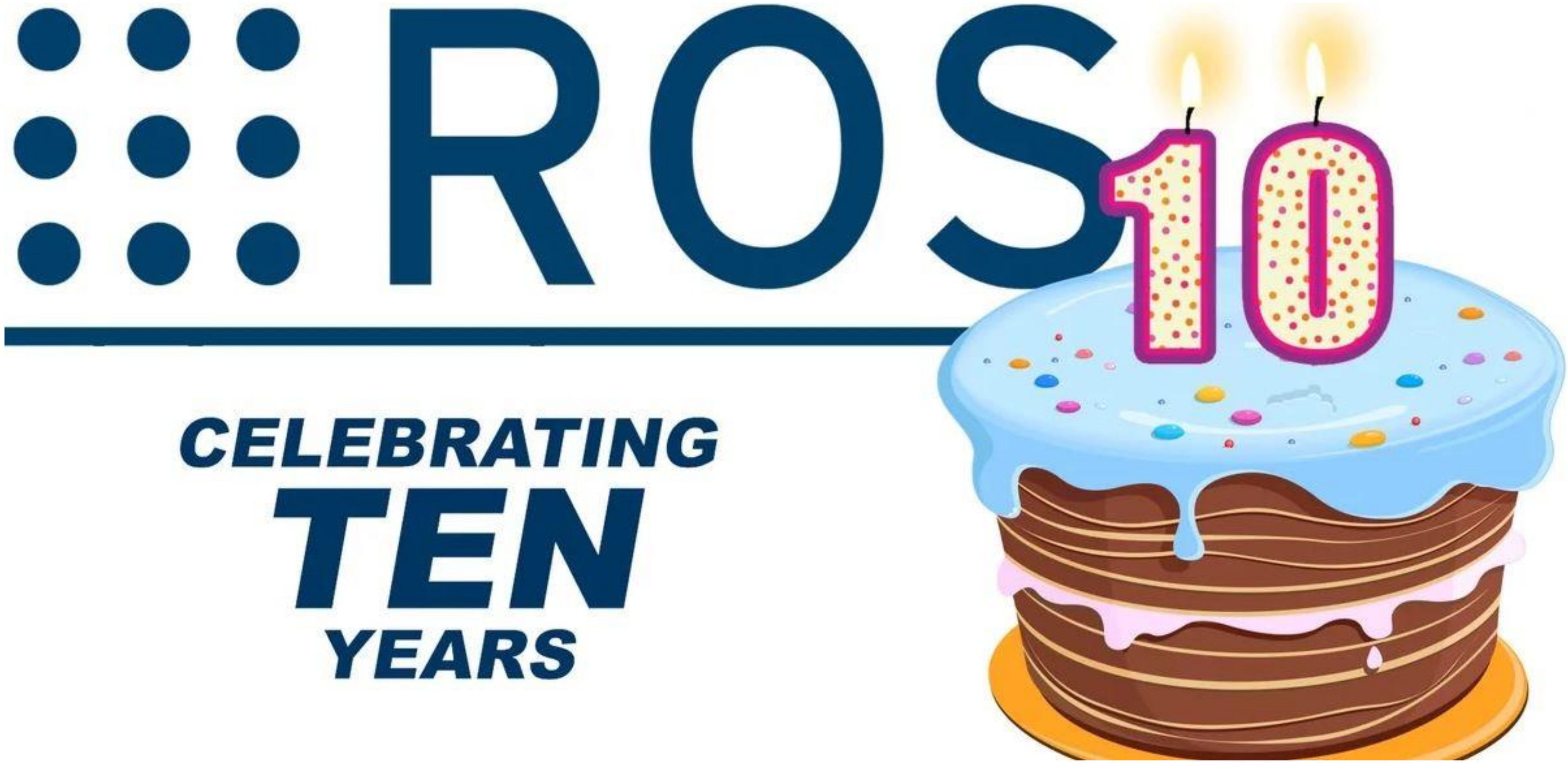
ROS is a Repository

only includes publicly released code!



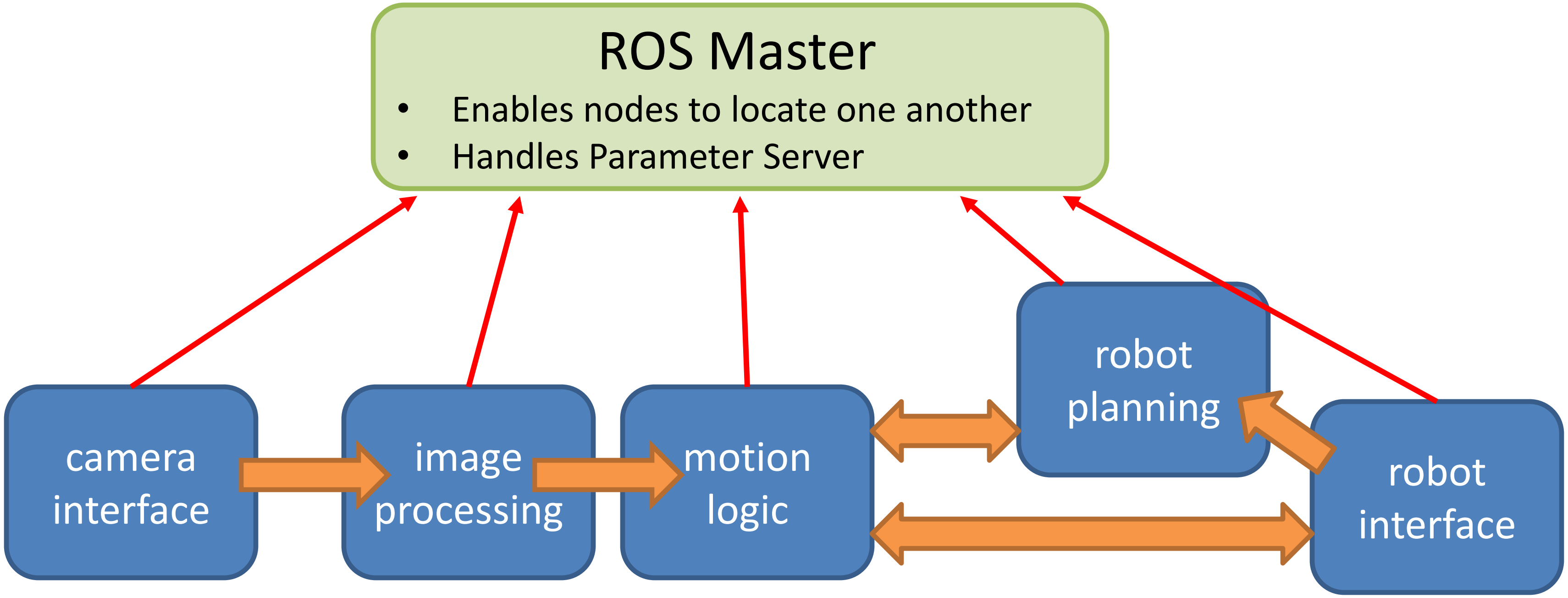
(From Morgan Quigley's "ROS: An Open-Source Framework for Modern Robotics")

ROS is …



https://vimeo.com/245826128

ROS Architecture: Nodes

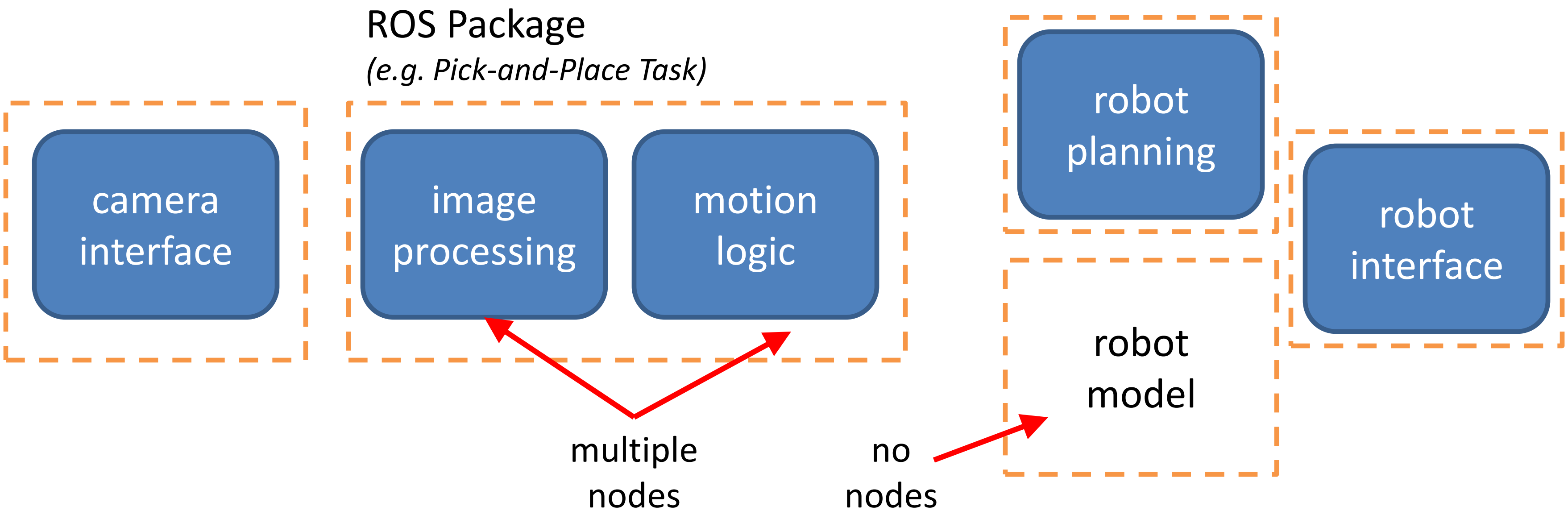


ROS Master

- Enables nodes to locate one another
- Handles Parameter Server

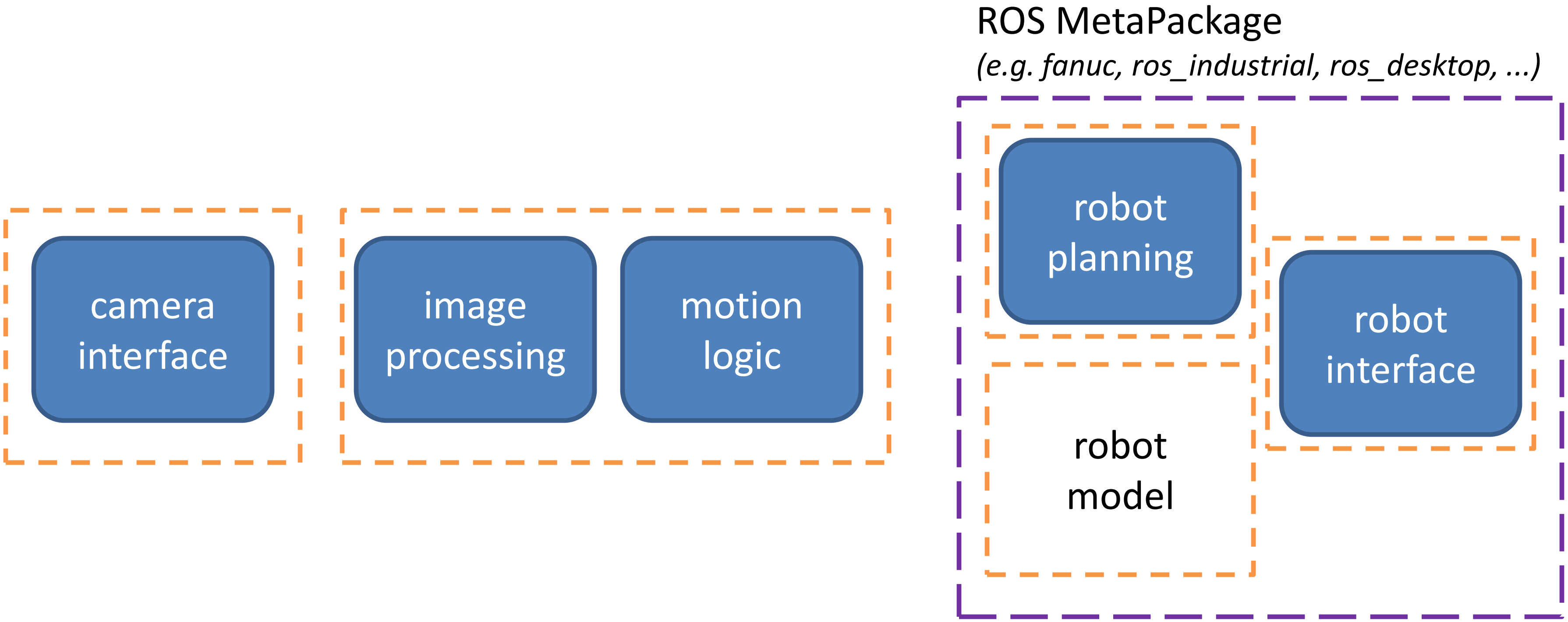
- A **Node** is a single ROS-enabled program
  - Most communication happens **between** nodes
  - Nodes can run on many different **devices**
- One **Master** per system

ROS Architecture: Packages



- ROS **Packages** are groups of related nodes/data
  - Many ROS commands are **package-oriented**

ROS Architecture: MetaPkg



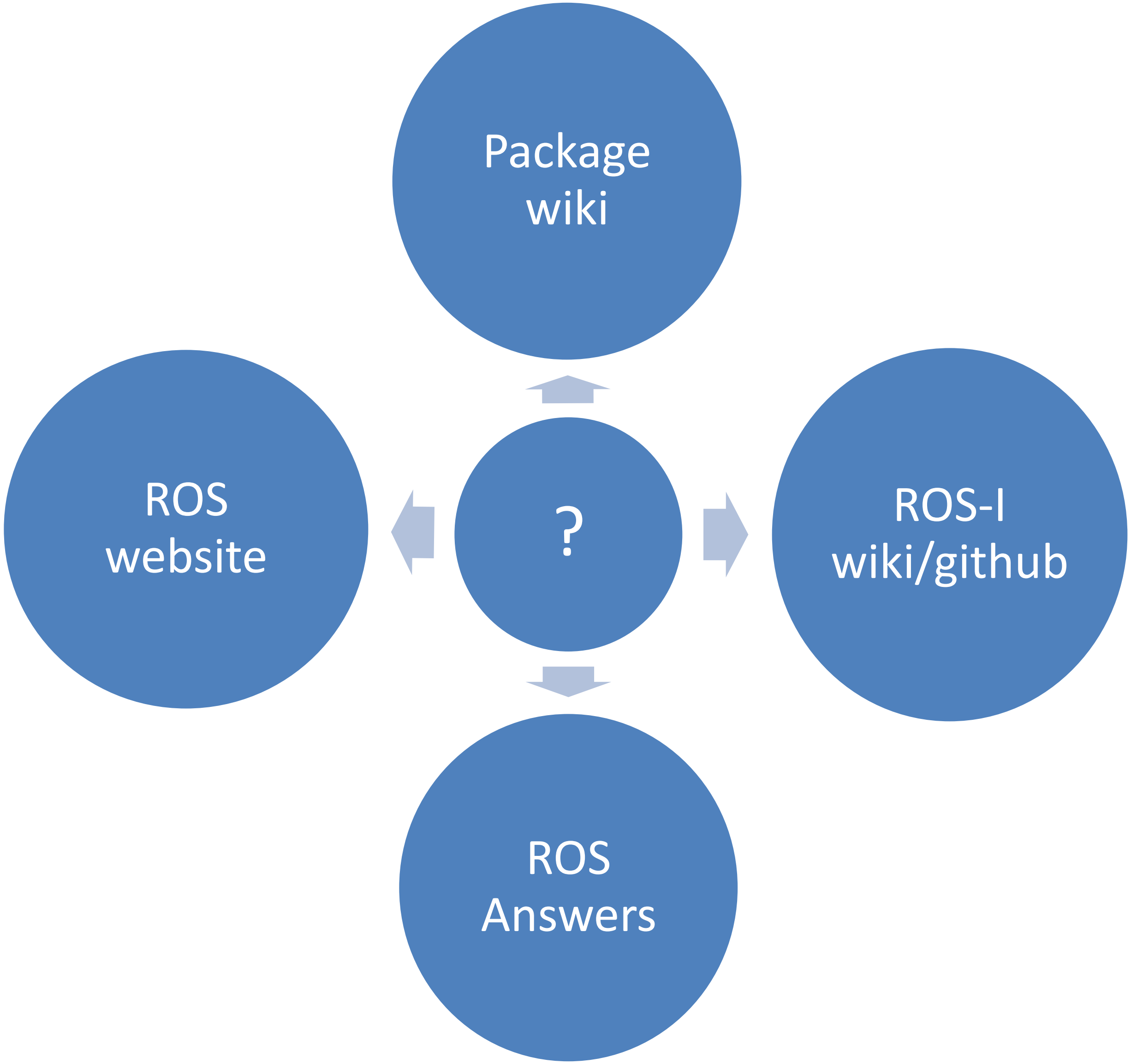
- **MetaPackages** are groups of related packages
  - Mostly for convenient install/deployment

ROS Programming

- ROS uses **platform-agnostic** methods for most communication
  - TCP/IP Sockets, XML, etc.

- Can intermix programming languages
  - primary: C++, Python, Lisp
  - also: C#, Java, Matlab, etc.
  - We will be using C++ for our exercises

ROS Resources



?

Package wiki

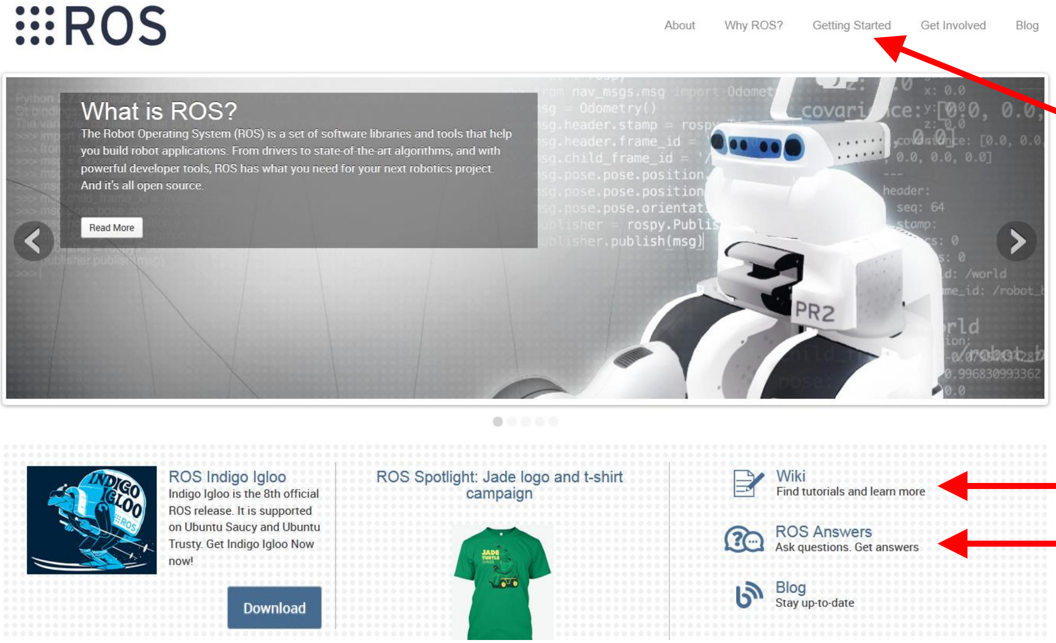
ROS website

ROS-I wiki/github

ROS Answers

ROS.org Website

http://ros.org



ROS

About | Why ROS? | Getting Started | Get Involved | Blog

What is ROS?

The Robot Operating System (ROS) is a set of software libraries and tools that help you build robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools, ROS has what you need for your next robotics project. And it's all open source.

Read More

ROS Indigo Igloo

Indigo Igloo is the 8th official ROS release. It is supported on Ubuntu Saucy and Ubuntu Trusty. Get Indigo Igloo Now now!

Download

ROS Spotlight: Jade logo and t-shirt campaign

Wiki
Find tutorials and learn more

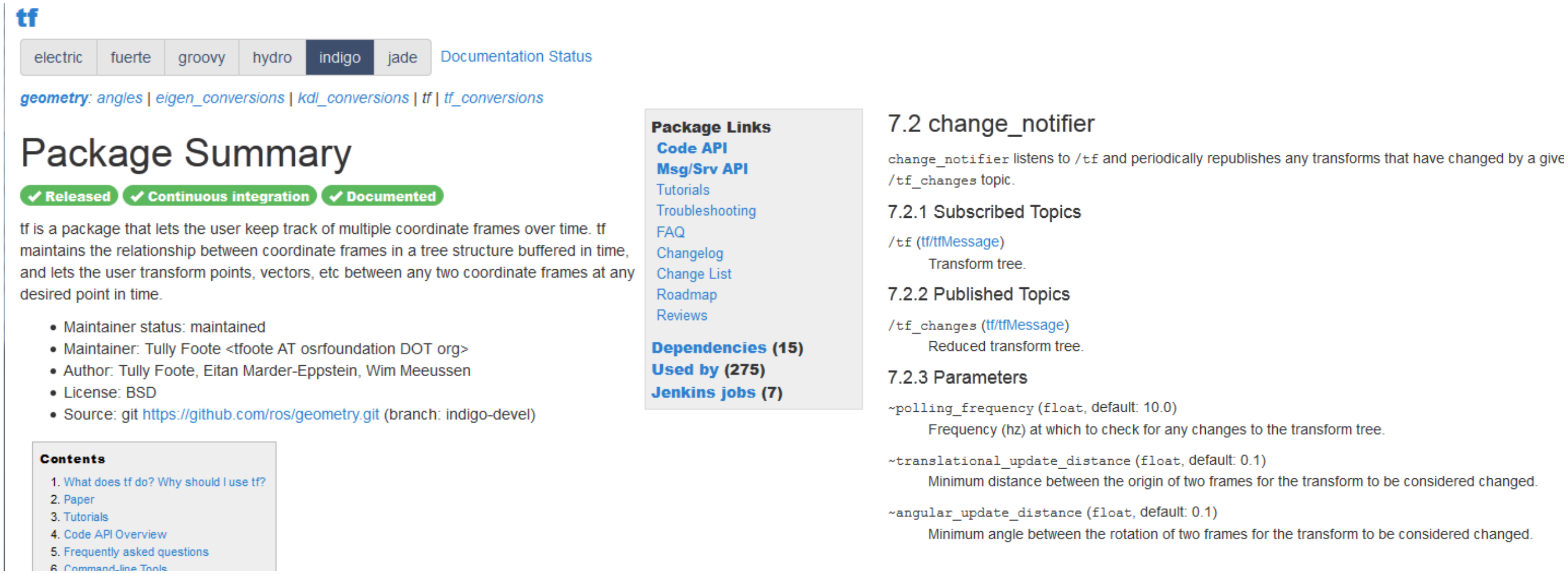
ROS Answers
Ask questions. Get answers

Blog
Stay up-to-date

- Install Instructions
- Tutorials
- Links
  - Packages, ROS Answers, etc.

Package Wiki

http://wiki.ros.org/<packageName>



tf

electric | fuerte | groovy | hydro | indigo | jade | Documentation Status

geometry: *angles* | *eigen_conversions* | *kdl_conversions* | *tf* | *tf_conversions*

Package Summary

✓ Released ✓ Continuous integration ✓ Documented

tf is a package that lets the user keep track of multiple coordinate frames over time. tf maintains the relationship between coordinate frames in a tree structure buffered in time, and lets the user transform points, vectors, etc between any two coordinate frames at any desired point in time.

- Maintainer status: maintained
- Maintainer: Tully Foote <tfoote AT osrfoundation DOT org>
- Author: Tully Foote, Eitan Marder-Eppstein, Wim Meeussen
- License: BSD
- Source: git https://github.com/ros/geometry.git (branch: indigo-devel)

Contents

Package Links

- **Code API**
- **Msg/Srv API**
- Tutorials
- Troubleshooting
- FAQ
- Changelog
- Change List
- Roadmap
- Reviews

Dependencies (15)
Used by (275)
Jenkins jobs (7)

7.2 change_notifier

`change_notifier` listens to `/tf` and periodically republishes any transforms that have changed by a give `/tf_changes` topic.

7.2.1 Subscribed Topics

`/tf` (tf/tfMessage)
Transform tree.

7.2.2 Published Topics

`/tf_changes` (tf/tfMessage)
Reduced transform tree.

7.2.3 Parameters

`~polling_frequency` (`float`, default: 10.0)
Frequency (hz) at which to check for any changes to the transform tree.

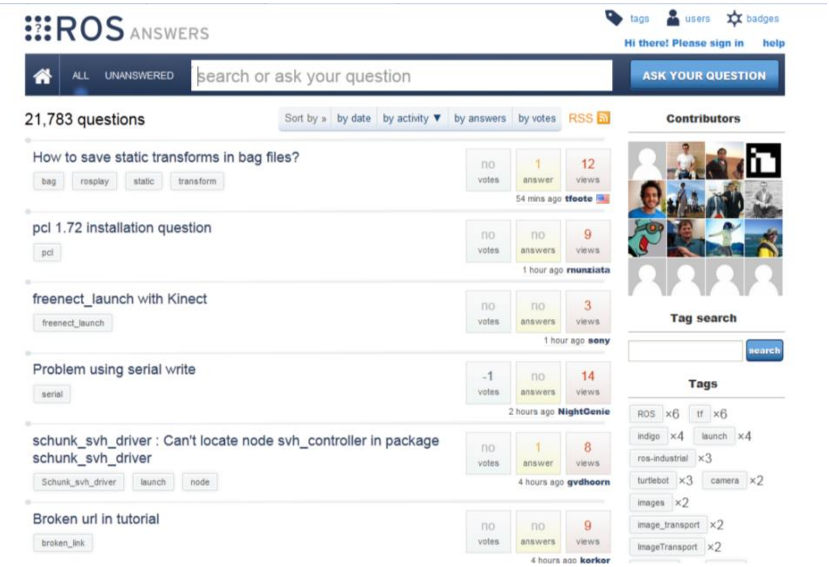
`~translational_update_distance` (`float`, default: 0.1)
Minimum distance between the origin of two frames for the transform to be considered changed.

`~angular_update_distance` (`float`, default: 0.1)
Minimum angle between the rotation of two frames for the transform to be considered changed.

- Description / Usage
- Tutorials
- Code / Msg API
- Source-Code link
- Bug Reporting

ROS Answers

http://answers.ros.org



ROS ANSWERS

tags users badges

Hi there! Please sign in help

ALL UNANSWERED search or ask your question ASK YOUR QUESTION

21,783 questions

Sort by » by date by activity ▼ by answers by votes RSS

Question	Tags	Votes	Answers	Views	Activity
How to save static transforms in bag files?	bag, rosplay, static, transform	no votes	1 answer	12 views	54 mins ago tfoote
pcl 1.72 installation question	pcl	no votes	no answers	9 views	1 hour ago rnunziata
freenect_launch with Kinect	freenect_launch	no votes	no answers	3 views	1 hour ago sony
Problem using serial write	serial	-1 votes	no answers	14 views	2 hours ago NightGenie
schunk_svh_driver : Can't locate node svh_controller in package schunk_svh_driver	Schunk_svh_driver, launch, node	no votes	1 answer	8 views	4 hours ago gvdhoorn
Broken url in tutorial	broken_link	no votes	no answers	9 views	4 hours ago korkor

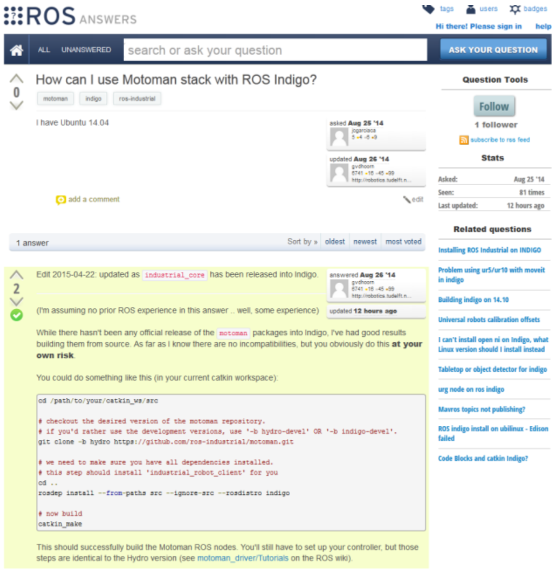
Contributors

Tag search

search

Tags

Tag	Count
ROS	×6
tf	×6
indigo	×4
launch	×4
ros-industrial	×3
turtlebot	×3
camera	×2
images	×2
image_transport	×2
ImageTransport	×2



ROS ANSWERS

tags users badges

Hi there! Please sign in help

ALL UNANSWERED search or ask your question ASK YOUR QUESTION

0 How can I use Motoman stack with ROS Indigo?

motoman indigo ros-industrial

I have Ubuntu 14.04

asked **Aug 25 '14** jogarciaca 5 •4 •6 •9

updated **Aug 26 '14** gvdhoorn 6741 •16 •45 •99 http://robotics.tudelft.n...

add a comment edit

1 answer Sort by » oldest newest most voted

2 Edit 2015-04-22: updated as `industrial_core` has been released into Indigo.

answered **Aug 26 '14** gvdhoorn 6741 •16 •45 •99 http://robotics.tudelft.n...

updated **12 hours ago**

(I'm assuming no prior ROS experience in this answer .. well, some experience)

While there hasn't been any official release of the `motoman` packages into Indigo, I've had good results building them from source. As far as I know there are no incompatibilities, but you obviously do this **at your own risk**.

You could do something like this (in your current catkin workspace):

```
cd /path/to/your/catkin_ws/src

# checkout the desired version of the motoman repository.
# if you'd rather use the development versions, use '-b hydro-devel' OR '-b indigo-devel'.
git clone -b hydro https://github.com/ros-industrial/motoman.git

# we need to make sure you have all dependencies installed.
# this step should install 'industrial_robot_client' for you
cd ..
rosdep install --from-paths src --ignore-src --rosdistro indigo

# now build
catkin_make
```

This should successfully build the Motoman ROS nodes. You'll still have to set up your controller, but those steps are identical to the Hydro version (see motoman_driver/Tutorials on the ROS wiki).

Question Tools

Follow

1 follower

subscribe to rss feed

Stats

Asked:	Aug 25 '14
Seen:	81 times
Last updated:	12 hours ago

Related questions

- Installing ROS Industrial on INDIGO
- Problem using ur5/ur10 with moveit in indigo
- Building indigo on 14.10
- Universal robots calibration offsets
- I can't install open ni on Indigo, what Linux version should I install instead
- Tabletop or object detector for indigo
- urg node on ros indigo
- Mavros topics not publishing?
- ROS indigo install on ubilinux - Edison failed
- Code Blocks and catkin Indigo?

- Quick responses to Good Questions
- Search by text or tag
- Don’t re-invent the wheel!

ROS is a Community

- No Central "Authority" for Help/Support
  - Many users can provide better (?) support
  - ROS-I Consortium can help fill that need

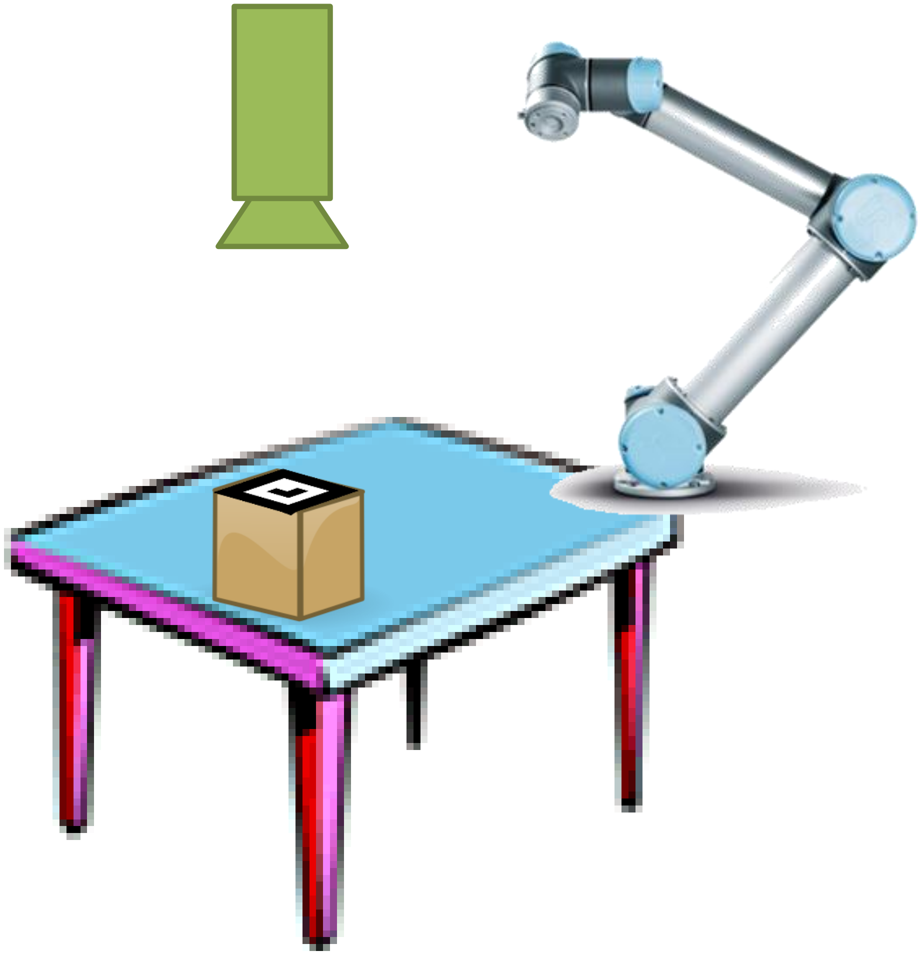
- Most ROS-code is open-source
  - can be reviewed / improved by everyone
  - we count on **YOU** to help ROS grow!

What is ROS to you?

Training Goals:

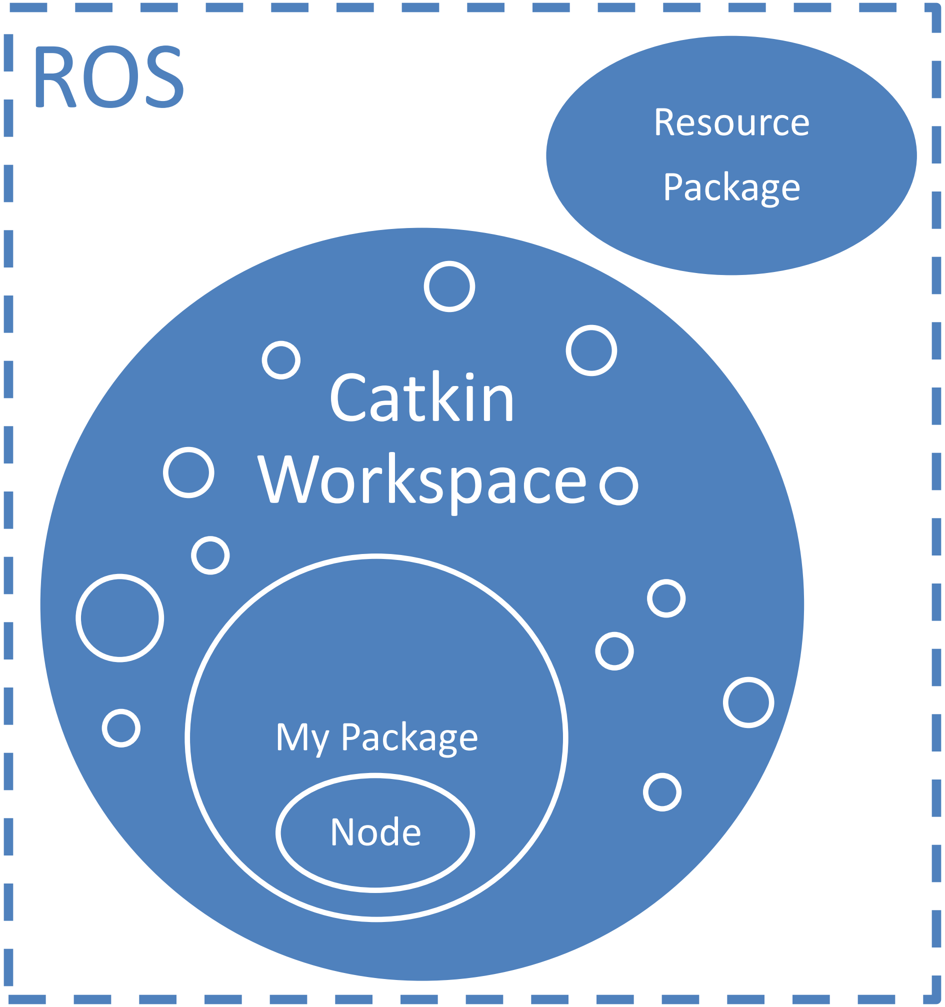
- Show you ROS as a software framework
- Show you ROS as a tool for problem solving
- Apply course concepts to a sample application
- Ask lots of questions and break things.

Scan & Plan "Application"



Day 1 Progression

- ❑ Install ROS
- ❑ Create Workspace
- ❑ Add “resources”
- ❑ Create Package
- ❑ Create Node
  - ❑ Basic ROS Node
  - ❑ Interact with other nodes
    - ❑ Messages
    - ❑ Services
- ❑ Run Node
  - ❑ rosrun
  - ❑ roslaunch
  - ❑ rosparam



Installing ROS

Getting ROS

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

Roscore

roscore is a collection of nodes and programs that are pre-requisites of a ROS-based system

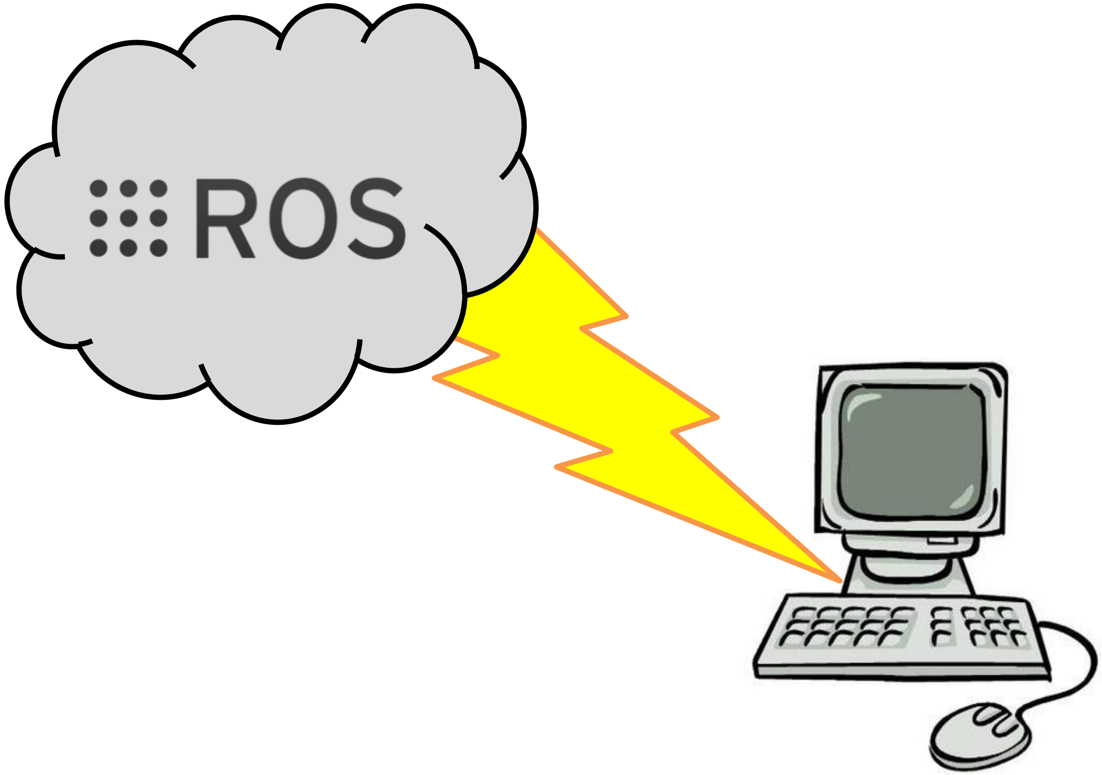
To check your install, open a terminal and type:

roscore

To kill the process, press ***Ctrl+C*** while in the window running ***roscore***

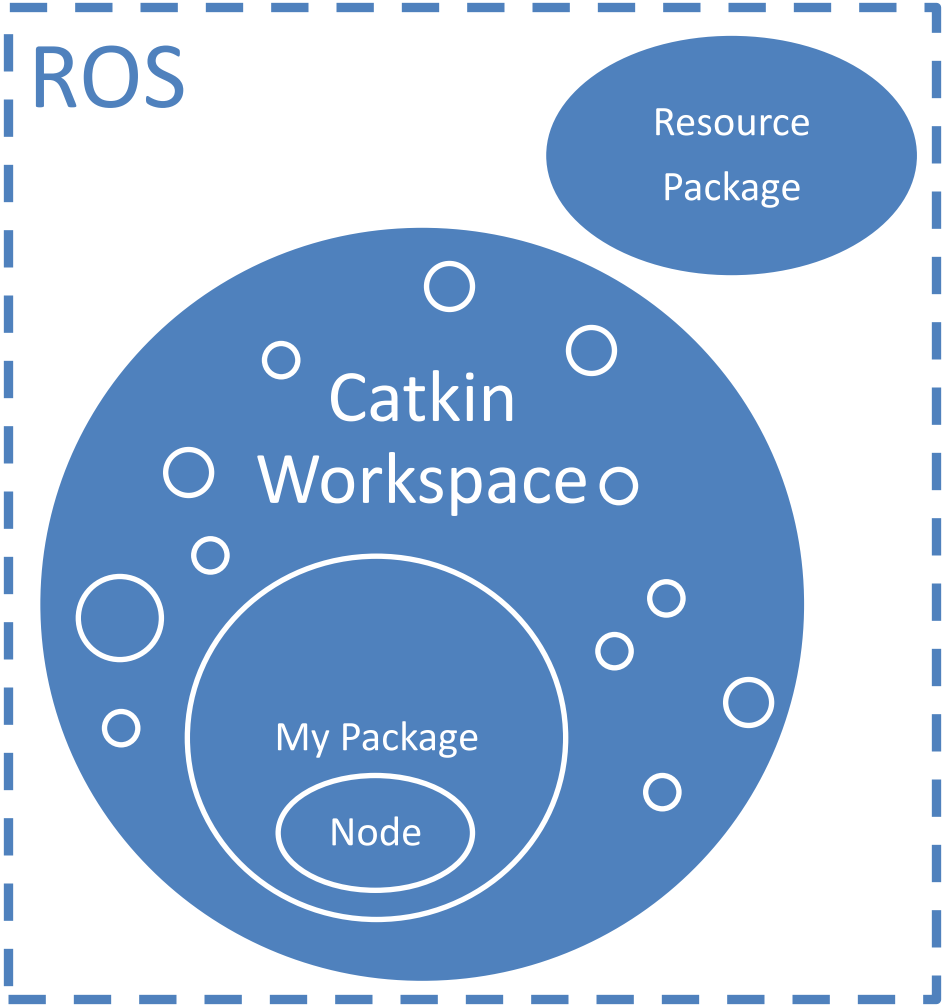
Exercise 1.0

Basic ROS Install/Setup



Day 1 Progression

- ✓ Install ROS (check install)
- ❑ Create Workspace
- ❑ Add “resources”
- ❑ Create Package
- ❑ Create Node
  - ❑ Basic ROS Node
  - ❑ Interact with other nodes
    - ❑ Messages
    - ❑ Services
- ❑ Run Node
  - ❑ rosrun
  - ❑ roslaunch



Creating a ROS Workspace

Catkin

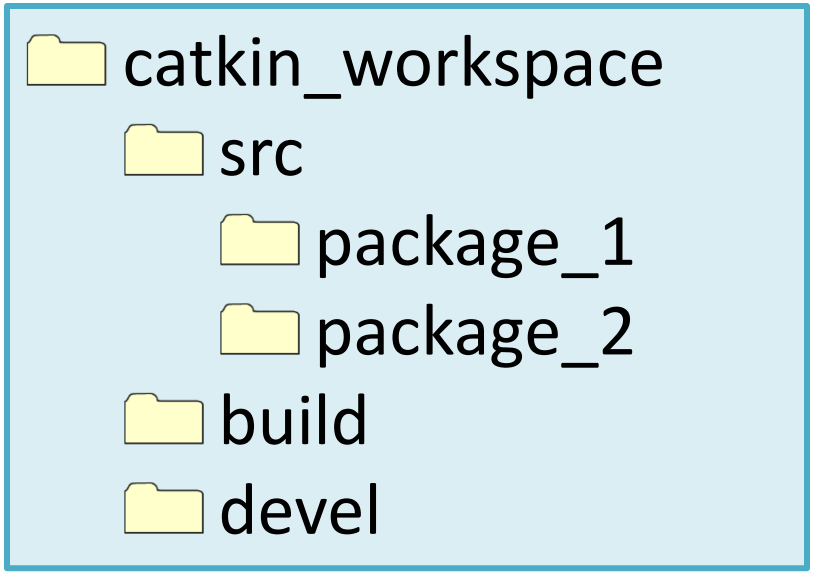
- ROS uses the **catkin** build system
  - based on CMAKE
  - cross-platform (Ubuntu, Windows, embedded...)
  - replaces older **rosbuild** system
    - different build commands, directory structure, etc.
    - most packages have already been upgraded to catkin
    - rosbuild: `manifest.xml`, catkin: `package.xml`

```
[ 80%] Building C object enu/src/libswiftnav/src/CMakeFiles/swiftnav-static.di
almanac.c.o
[ 84%] Building C object enu/src/libswiftnav/src/CMakeFiles/swiftnav.dir/gpsti
.c.o
[ 88%] Building C ob                               es/swiftnav-static.di
gpstime.c.o
Linking C shared lit                               /libswiftnav.so
Linking C static lit                               /libswiftnav-static.a
[ 88%] Built target
[ 88%] Built target
Scanning dependencie
[ 92%] Building CXX                                /enu_ros.cpp.o
Linking CXX shared l                               ib/libenu_ros.so
[ 92%] Built target
Scanning dependencie
Scanning dependencie
[ 96%] Building CXX                                src/to_fix.cpp.o
[100%] Building CXX                                r/src/from_fix.cpp.o
Linking CXX executable /home/rgariepy/ros/enu/devel/lib/enu/from_fix
```

http://www.clearpathrobotics.com/blog/introducing-catkin/

Catkin Workspace

- Catkin uses a specific directory structure:
  - each “project” typically gets its own **catkin workspace**
  - all packages/source files go in the **src** directory
  - temporary build-files are created in **build**
  - results are placed in **devel**



```
catkin_workspace
    src
        package_1
        package_2
    build
    devel
```

Catkin Build Process

Setup (one-time)

1. Create a catkin workspace somewhere
   - `catkin_ws`
   - `src` sub-directory must be created manually
   - `build`, `devel` directories created automatically
2. Run `catkin init` from workspace root
3. Download/create **packages** in `src` subdir

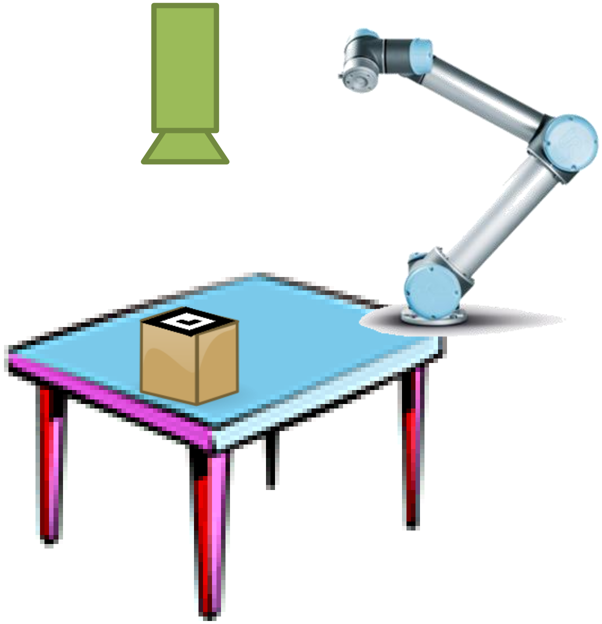
Compile-Time

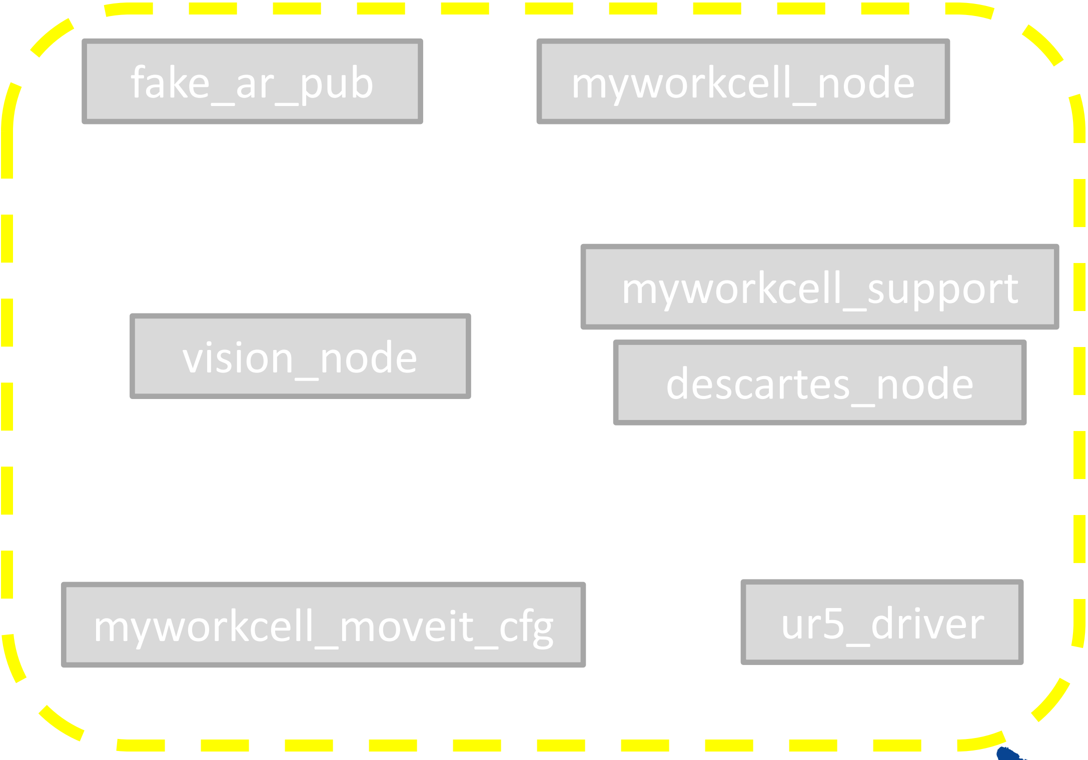
1. Run `catkin build` anywhere in the workspace
2. Run `source devel/setup.bash` to make workspace visible to ROS
   - Must re-execute in **each** new terminal window
   - Can add to `~/.bashrc` to automate this process

Exercise 1.1

Exercise 1.1

Create a Catkin Workspace





fake_ar_pub

myworkcell_node

myworkcell_support

vision_node

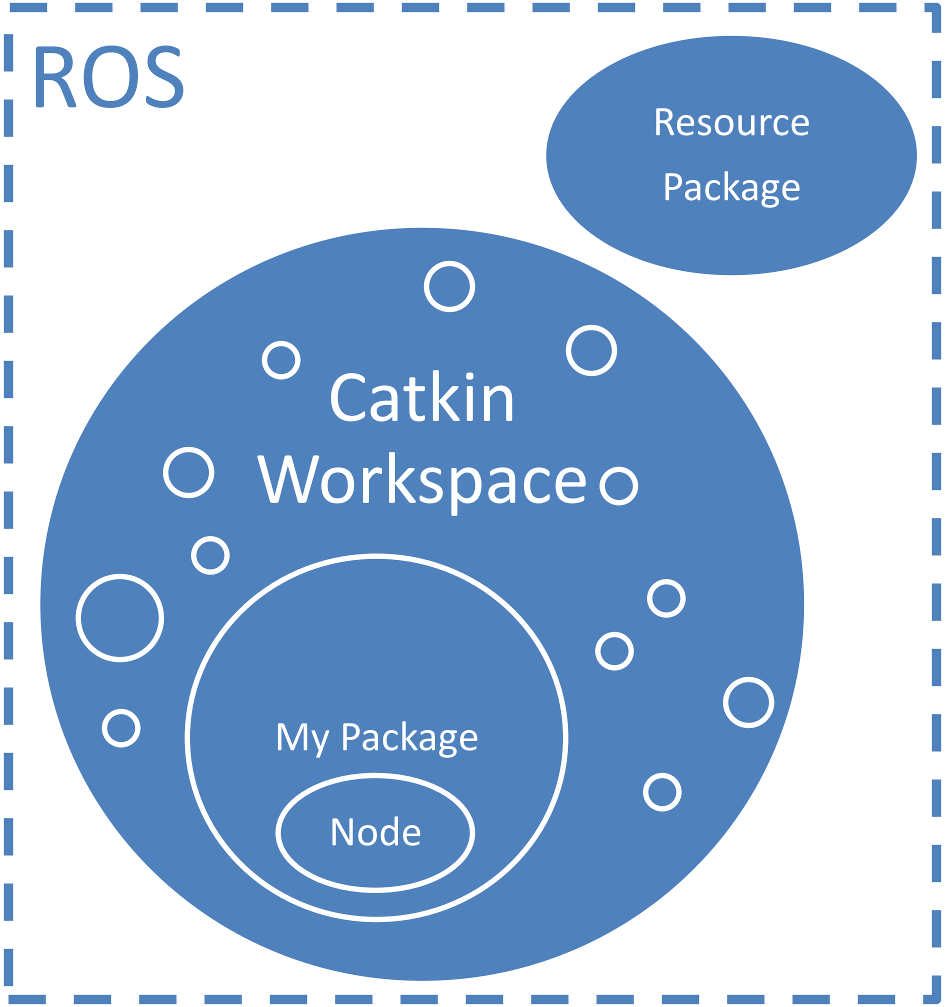
descartes_node

myworkcell_moveit_cfg

ur5_driver

Day 1 Progression

- ✓ Install ROS
- ✓ Create Workspace
- ❑ Add “resources”
- ❑ Create Package
- ❑ Create Node
  - ❑ Basic ROS Node
  - ❑ Interact with other nodes
    - ❑ Messages
    - ❑ Services
- ❑ Run Node
  - ❑ rosrun
  - ❑ roslaunch



Add 3rd-Party Packages

(a.k.a. “Resource” Packages)

Install options

Debian Packages

- Nearly “automatic”
- Recommended for end-users
- Stable
- Easy

Source Repositories

- Access “latest” code
- Most at Github.com
- More effort to setup
- Unstable*

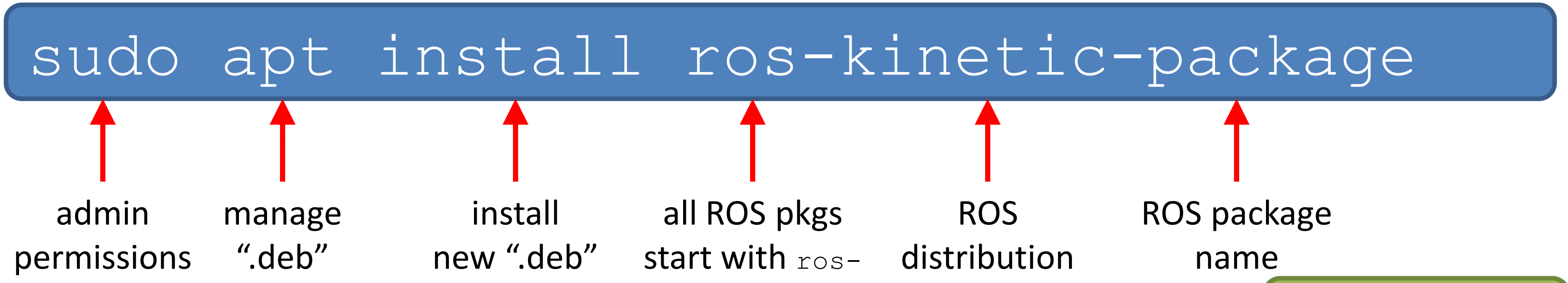
Can mix both options, as needed

Finding the Right Package

- ROS Website (http://ros.org/browse/)
  - – Browse/Search for known packages

- ROS Answers (http://answers.ros.org)
  - – When in doubt... ask someone!

Install using Debian Packages



```
sudo apt install ros-kinetic-package
```

admin permissions

manage ".deb"

install new ".deb"

all ROS pkgs start with `ros-`

ROS distribution

ROS package name

Use "-" not "_"

- Fully automatic install:
  - Download .deb package from central ROS repository
  - Copies files to standard locations (`/opt/ros/kinetic/...`)
  - ➢ Also installs any other required dependencies

- `sudo apt-get remove ros-distro-package`
  - Removes software (but not dependencies!)

Installing from Source

- Find GitHub repo
- Clone repo into your workspace src directory

```
cd catkin_ws/src
git clone http://github.com/user/repo.git
```

- Build your catkin workspace

```
cd catkin_ws
catkin build
```

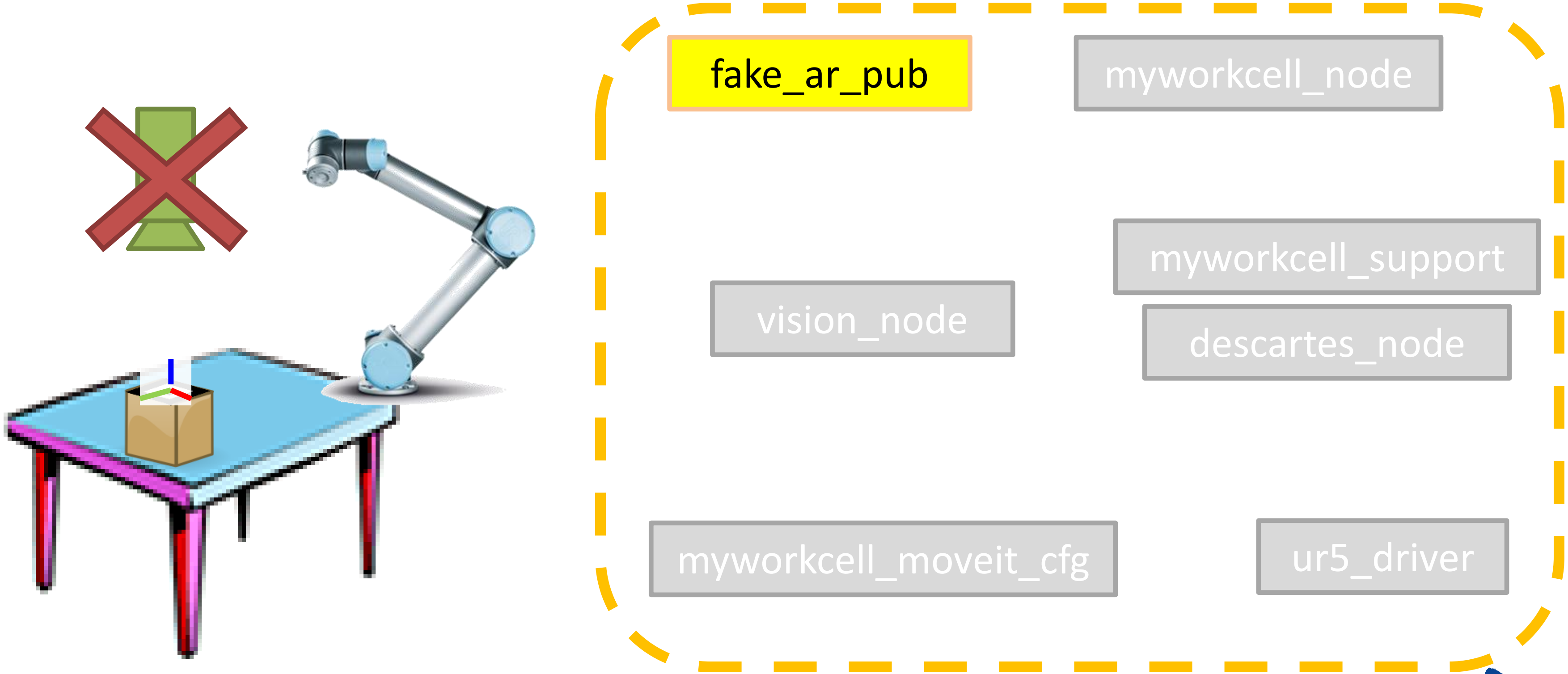
- Now the package and its resources are available to you

http://www.clearpathrobotics.com/blog/introducing-catkin/

Exercise 1.2

Exercise 1.2

Install "resource" packages



fake_ar_pub

myworkcell_node

myworkcell_support

vision_node

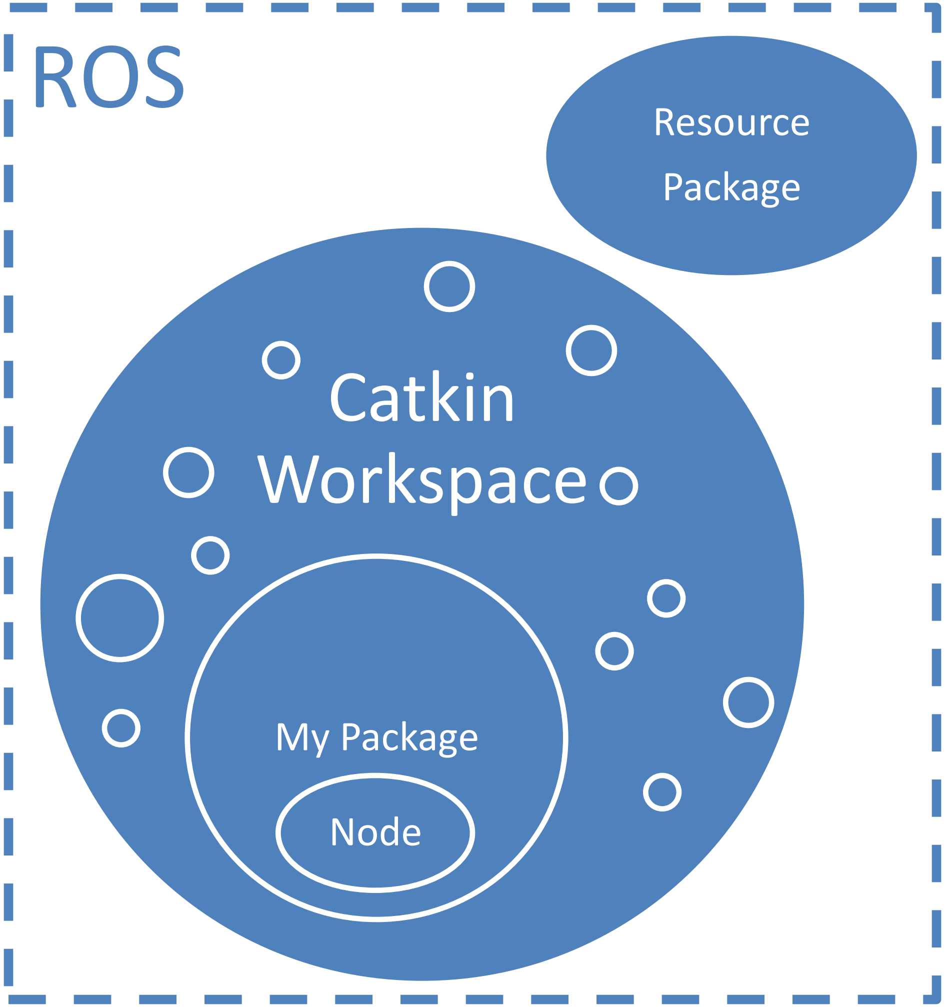
descartes_node

myworkcell_moveit_cfg

ur5_driver

Day 1 Progression

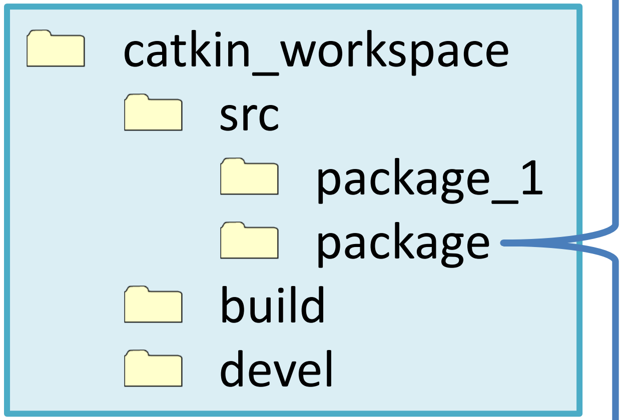
- ✓ Install ROS
- ✓ Create Workspace
- ✓ Add “resources”
- ❑ Create Package
- ❑ Create Node
  - ❑ Basic ROS Node
  - ❑ Interact with other nodes
    - ❑ Messages
    - ❑ Services
- ❑ Run Node
  - ❑ rosrun
  - ❑ roslaunch



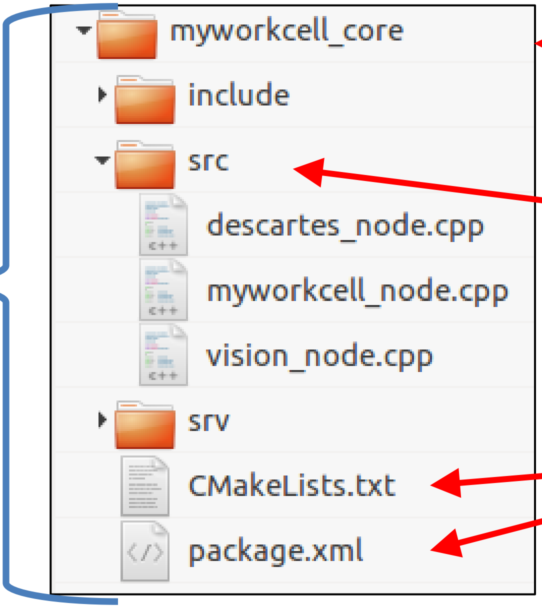
ROS Packages

ROS Package Contents

- ROS components are organized into **packages**
- Packages contain several **required files**:
  - `package.xml`
    - **metadata** for ROS: package name, description, dependencies, ...
  - `CMakeLists.txt`
    - **build rules** for catkin



catkin_workspace
src
package_1
package
build
devel



myworkcell_core
include
src
descartes_node.cpp
myworkcell_node.cpp
vision_node.cpp
srv
CMakeLists.txt
package.xml

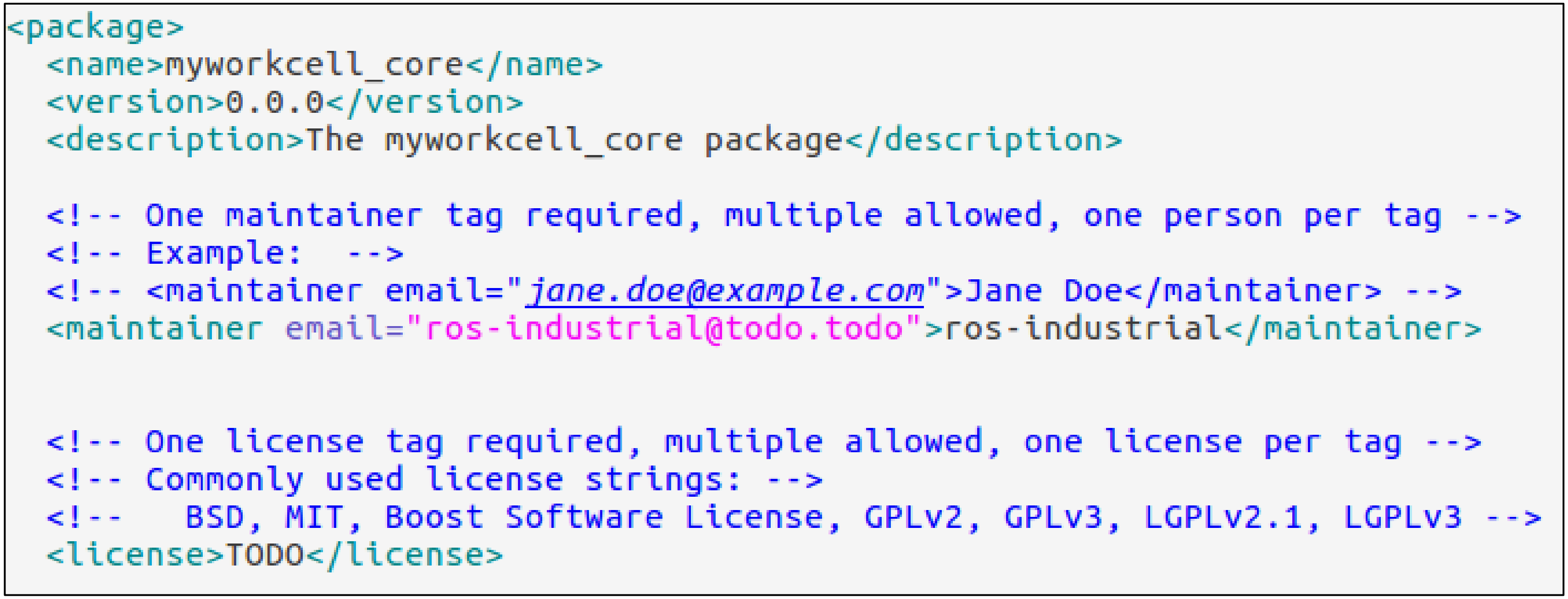
package directory

package source-files *(vs. catkin workspace src dir)*

required files

package.xml

- Metadata: name, description, author, license …



```
<package>
  <name>myworkcell_core</name>
  <version>0.0.0</version>
  <description>The myworkcell_core package</description>

  <!-- One maintainer tag required, multiple allowed, one person per tag -->
  <!-- Example:  -->
  <!-- <maintainer email="jane.doe@example.com">Jane Doe</maintainer> -->
  <maintainer email="ros-industrial@todo.todo">ros-industrial</maintainer>


  <!-- One license tag required, multiple allowed, one license per tag -->
  <!-- Commonly used license strings: -->
  <!--   BSD, MIT, Boost Software License, GPLv2, GPLv3, LGPLv2.1, LGPLv3 -->
  <license>TODO</license>
```

package.xml

- Metadata: name, description, author, license ...
- Dependencies:
  - Common
    - `<buildtool_depend>`: Needed to **build** itself. (Typically ***catkin****)*
    - `<depend>`: Needed to **build**, **export**, and **execution** dependency. (format “2” only)
  - Sometimes
    - `<build_depend>`: Needed to **build** this package.
    - `<build_export_depend>`: Needed to **build against** this package.
    - `<exec_depend>`: Needed to **run** code in this package.
  - Uncommon
    - `<test_depend>`: Only ***additional*** dependencies for unit tests.
    - `<doc_depend>`: Needed to generate documentation.

CMakeLists.txt

- Provides **rules** for **building software**
  - template file contains many examples

```
include_directories(include ${catkin_INCLUDE_DIRS})
```

Adds directories to CMAKE include rules

```
add_executable(myNode src/myNode.cpp src/widget.cpp)
```

Builds program `myNode`, from `myNode.cpp` and `widget.cpp`

```
target_link_libraries(myNode ${catkin_LIBRARIES})
```

Links node `myNode` to dependency libraries

ROS Package Commands

- `roscd package_name`

  Change to package directory

- **`rospack`**
  - `rospack find package_name`

    Find directory of `package_name`
  - `rospack list`

    List all ros packages installed
  - `rospack depends package_name`

    List all dependencies of `package_name`

Create New Package

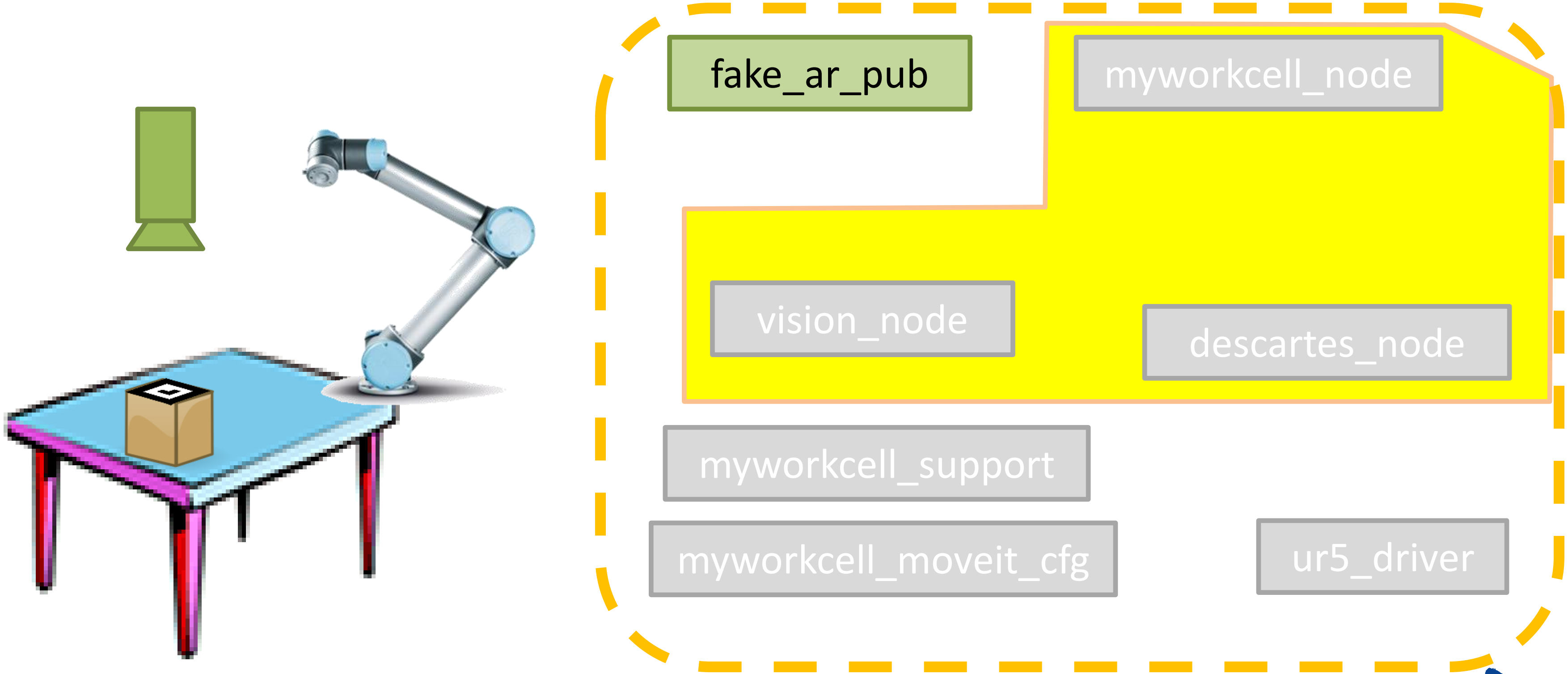
```
catkin create pkg mypkg --catkin-deps dep1 dep2
```

Easiest way to start a new package

- create directory, required files
- `mypkg` : name of package to be created
- `dep1/2` : dependency package names
  - automatically added to `CMakeLists` and `package.xml`
  - can manually add additional dependencies later

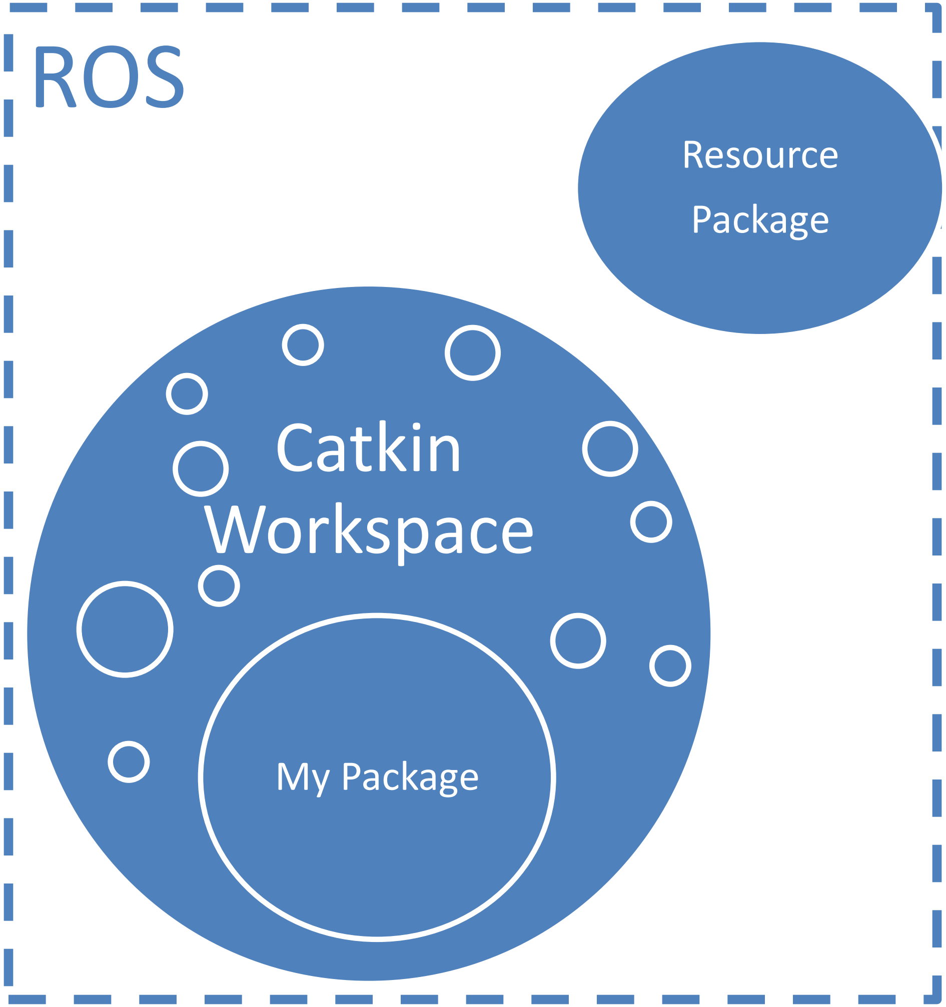
Exercise 1.3.1

Create Package



Day 1 Progression

- ✓ Install ROS
- ✓ Create Workspace
- ✓ Add "resources"
- ✓ Create Package
- ❑ Create Node
  - ❑ Basic ROS Node
  - ❑ Interact with other nodes
    - ❑ Messages
    - ❑ Services
- ❑ Run Node
  - ❑ rosrun
  - ❑ roslaunch



ROS Nodes

A Simple C++ ROS Node

Simple C++ Program

```
#include <iostream>

int main(int argc, char* argv[])
{


   std::cout << "Hello World!";

   return 0;
}
```

Simple C++ ROS Node

```
#include <ros/ros.h>

int main(int argc, char* argv[])
{
  ros::init(argc, argv, “hello”);
  ros::NodeHandle node;

  ROS_INFO_STREAM("Hello World!“);

  return 0;
}
```

ROS Node Commands

- `rosrun package_name node_name`

  execute ROS node

- **`rosnode`**
  - `rosnode list`

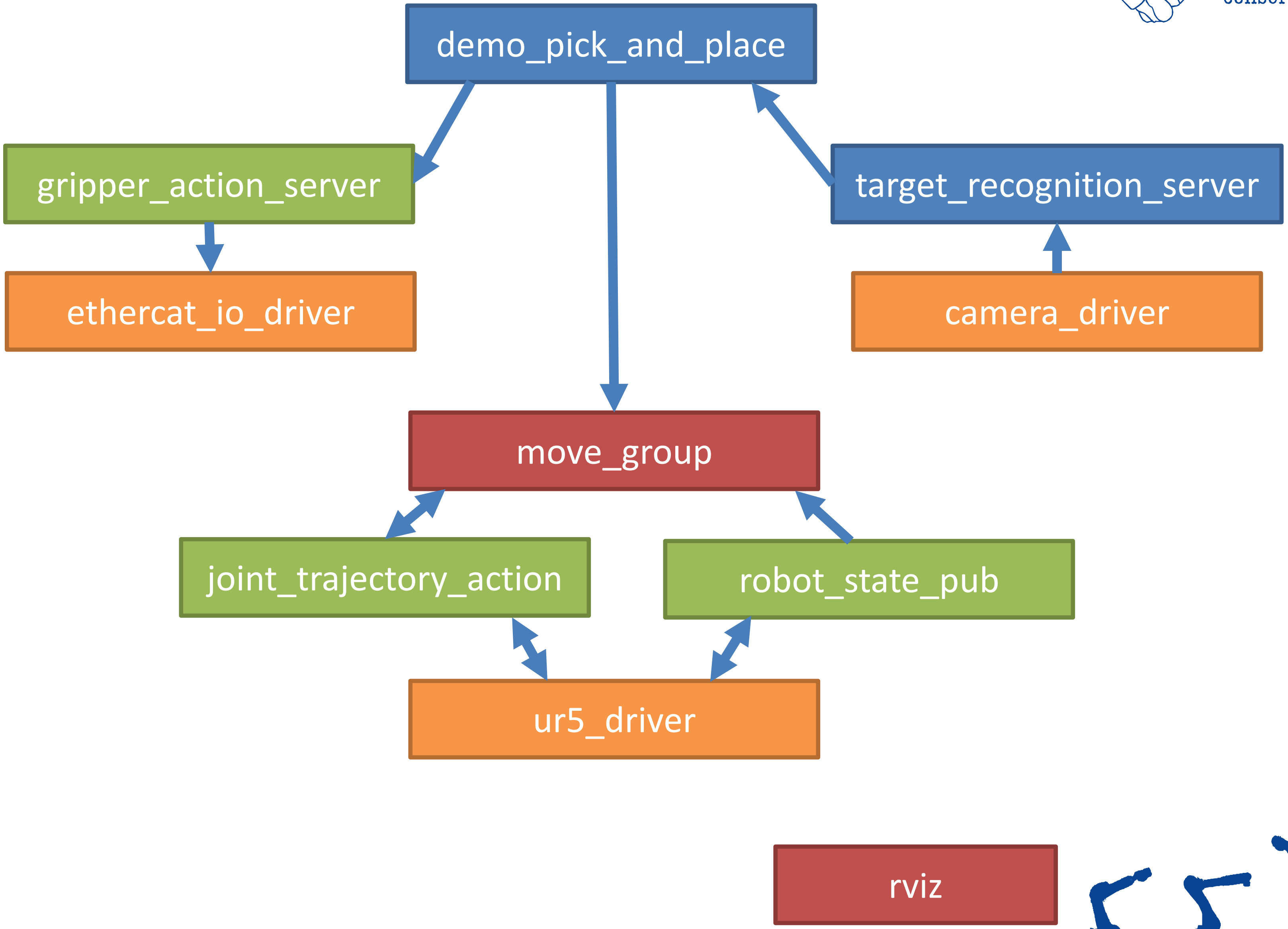
    View running nodes
  - `rosnode info node_name`

    View node details (publishers, subscribers, services, etc.)
  - `rosnode kill node_name`

    Kill running node; good for remote machines

    ➢ *`Ctrl+C` is usually easier*

“Real World” – Nodes

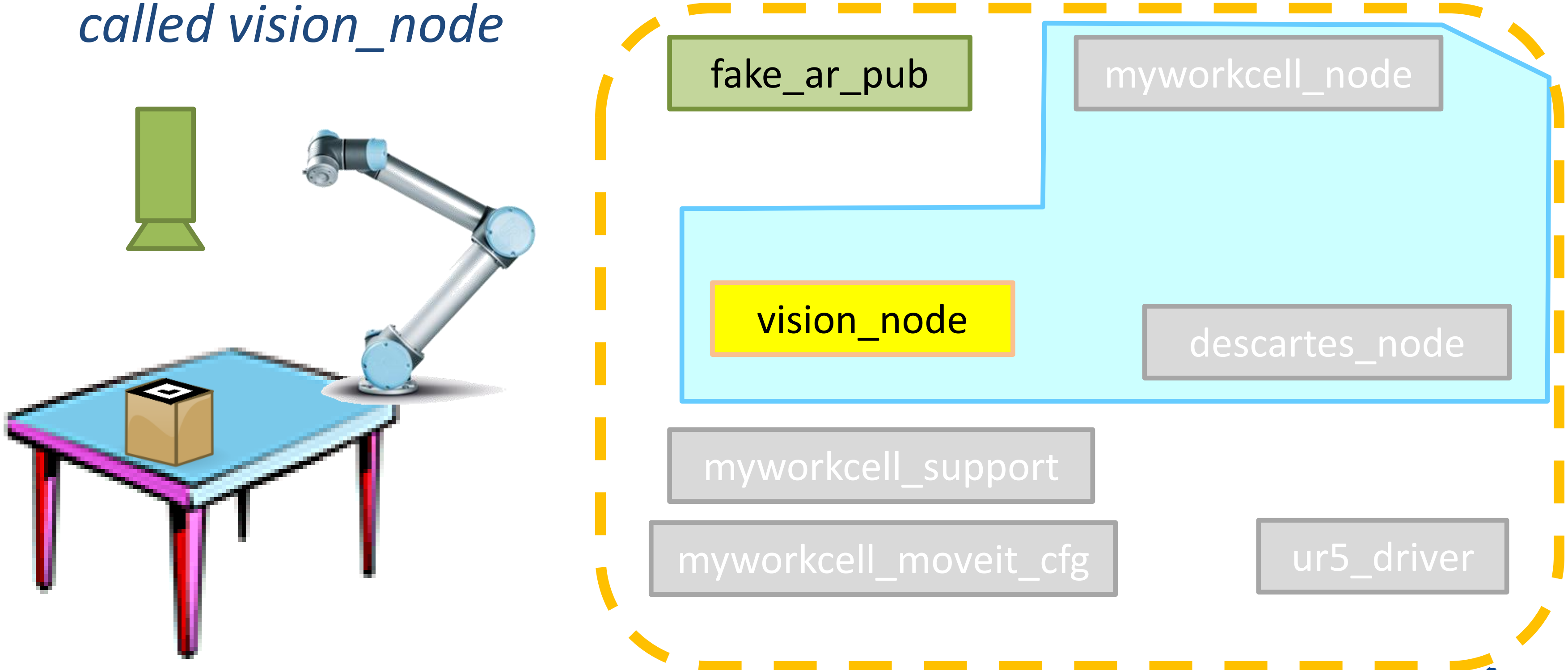


Exercise 1.3.2

Create a Node:

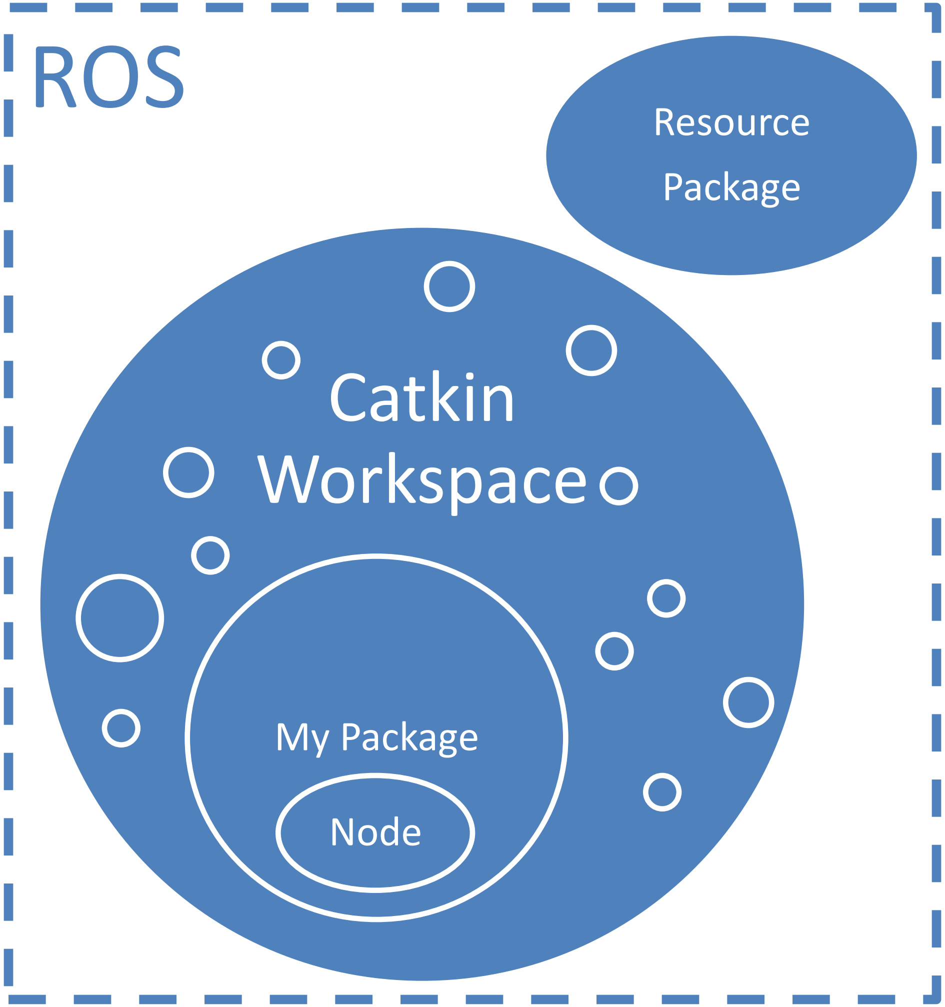
In myworkcell_core package

called vision_node



Day 1 Progression

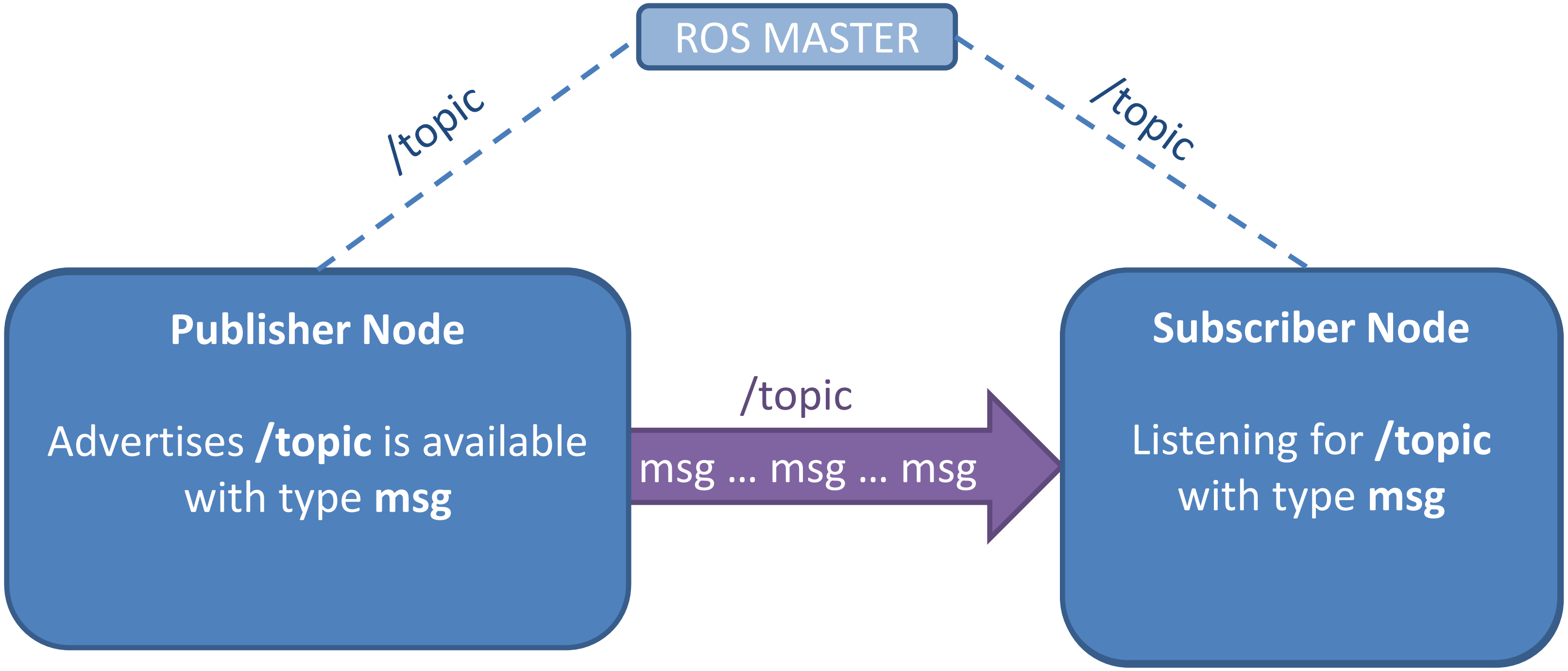
- ✓ Install ROS
- ✓ Create Workspace
- ✓ Add “resources”
- ✓ Create Package
- ✓ Create Node
  - ✓ Basic ROS Node
  - ❑ Interact with other nodes
    - ❑ Messages
    - ❑ Services
- ✓ Run Node
  - ✓ rosrun
  - ❑ roslaunch



Topics and Messages

ROS Topics/Messages

Topics are for **Streaming Data**



ROS MASTER

/topic

/topic

Publisher Node

Advertises **/topic** is available with type **msg**

/topic

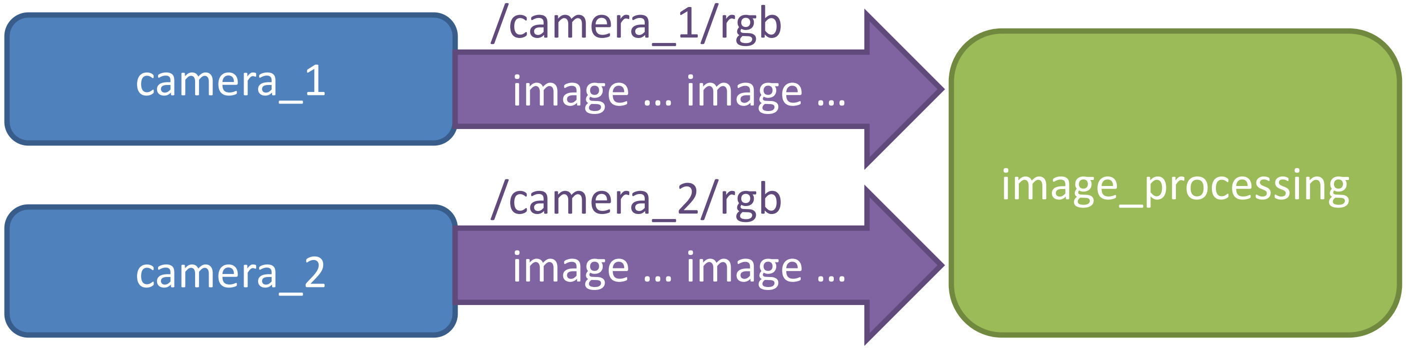
msg … msg … msg

Subscriber Node

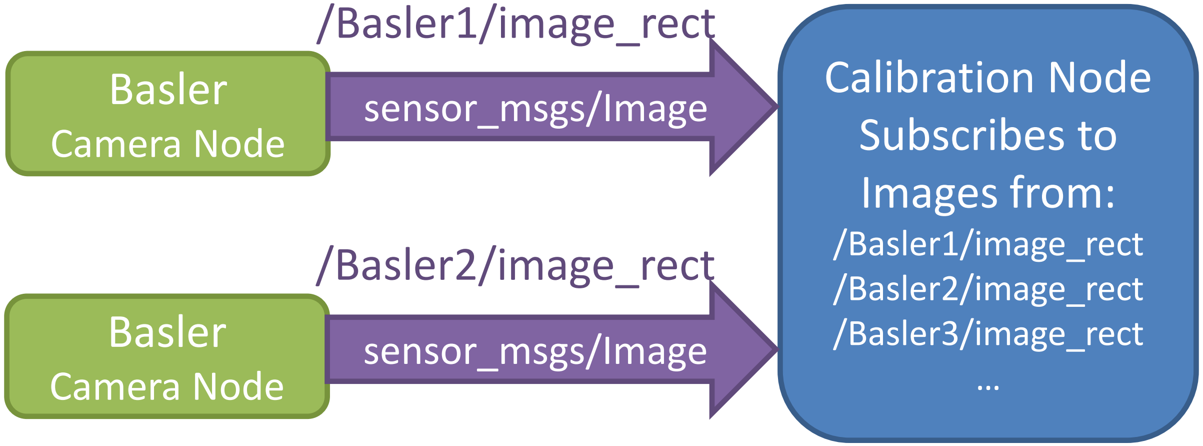
Listening for **/topic** with type **msg**

Topics vs. Messages

- Topics are **channels**, Messages are **data types**
  - Different topics can use the same Message type



Practical Example



/Basler1/image_rect

Basler Camera Node

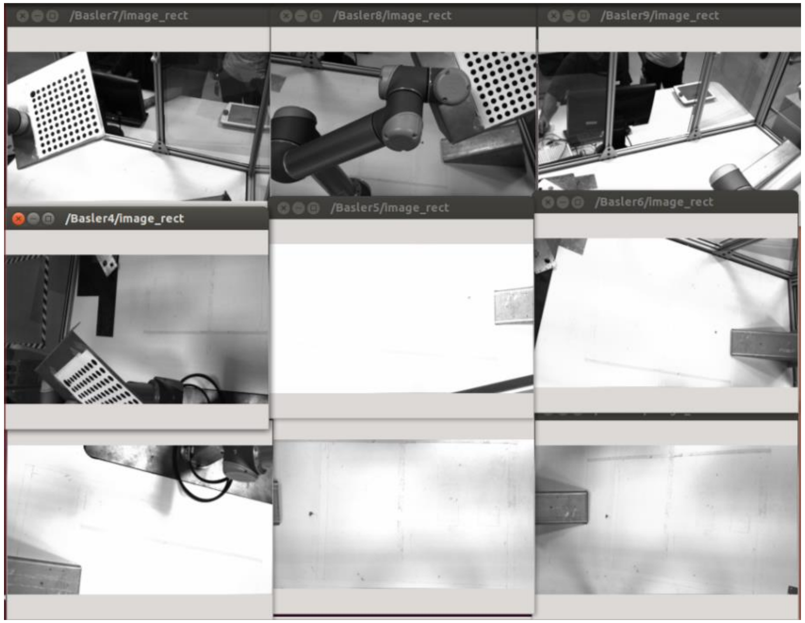
sensor_msgs/Image

/Basler2/image_rect

Basler Camera Node

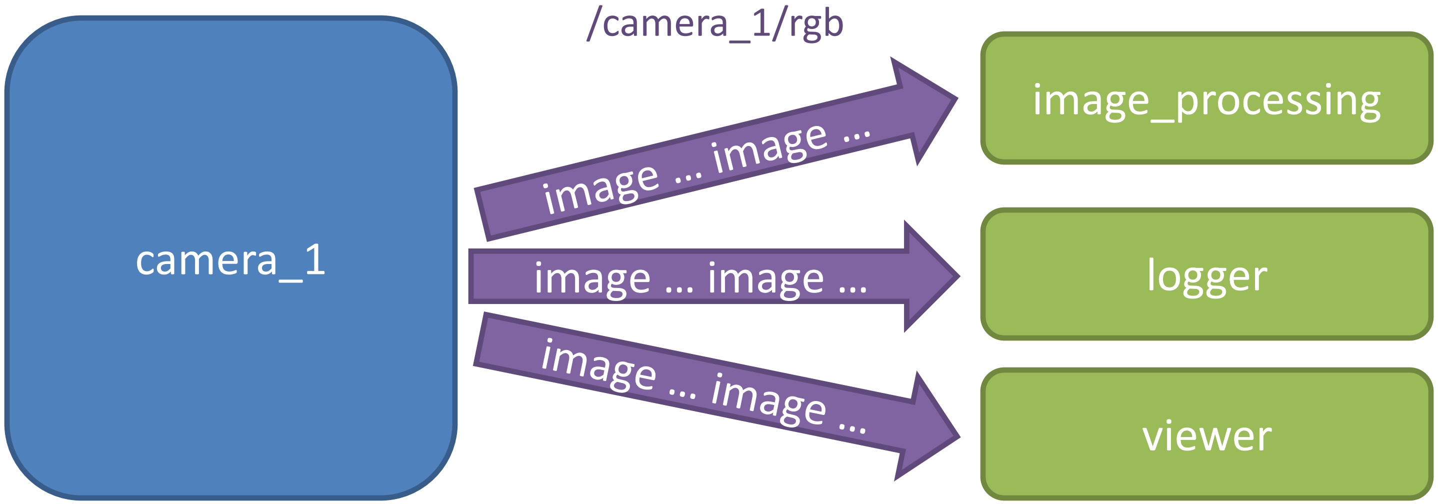
sensor_msgs/Image

Calibration Node Subscribes to Images from:
/Basler1/image_rect
/Basler2/image_rect
/Basler3/image_rect
…



Multiple Pub/Sub

- Many nodes can pub/sub to same topic
  - – comms are direct node-to-node



Topics : Details

- Each **Topic** is a stream of **Messages**:
  - sent by **publisher(s)**, received by **subscriber(s)**

- Messages are **asynchronous**
  - publishers don't know if anyone's listening
  - messages may be dropped
  - subscribers are event-triggered (by incoming messages)

- Typical Uses:
  - Sensor Readings: camera images, distance, I/O
  - Feedback: robot status/position
  - Open-Loop Commands: desired position

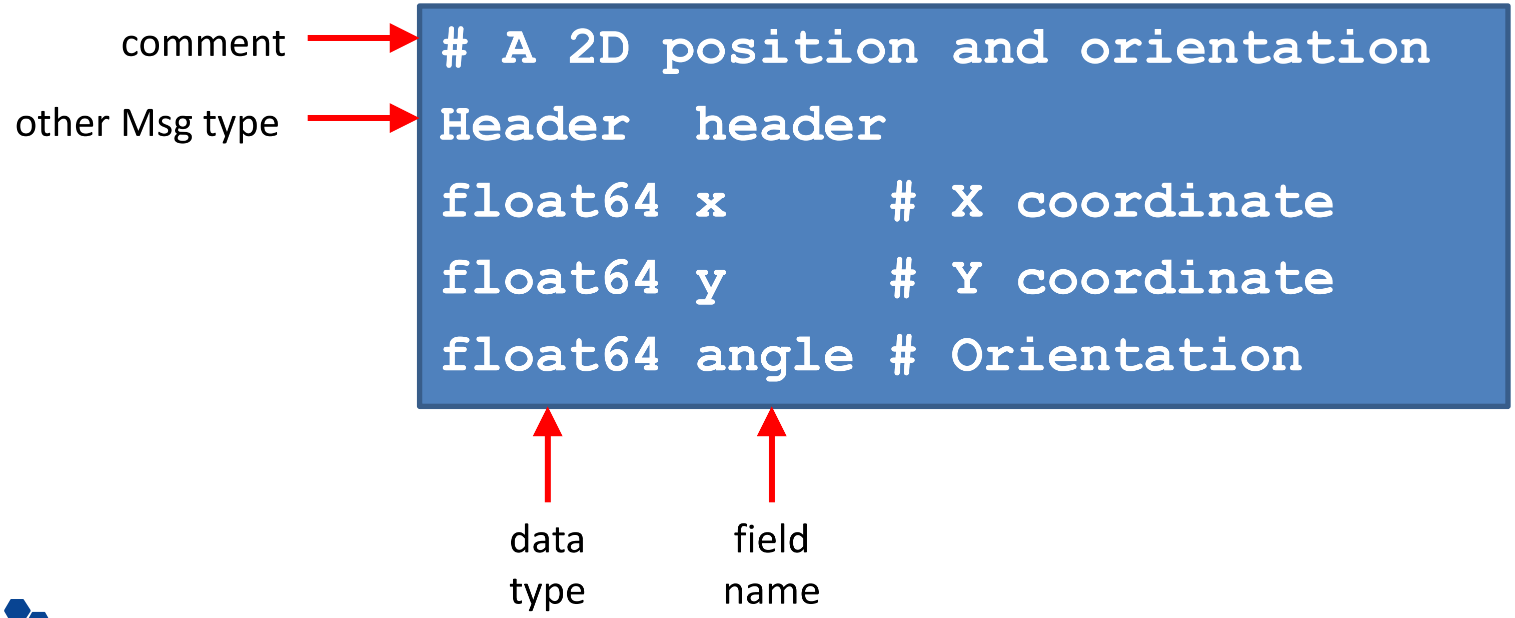
ROS Messages Types

- Similar to C structures
- Standard data primitives
  - Boolean: `bool`
  - Integer: `int8,int16,int32,int64`
  - Unsigned Integer: `uint8,uint16,uint32,uint64`
  - Floating Point: `float32, float64`
  - String: `string`
- Fixed length arrays: `bool[16]`
- Variable length arrays: `int32[]`
- Other: Nest message types for more complex data structure

Message Description File

- All Messages are defined by a **.msg** file

PathPosition.msg



```
# A 2D position and orientation
Header  header
float64 x     # X coordinate
float64 y     # Y coordinate
float64 angle # Orientation
```

comment

other Msg type

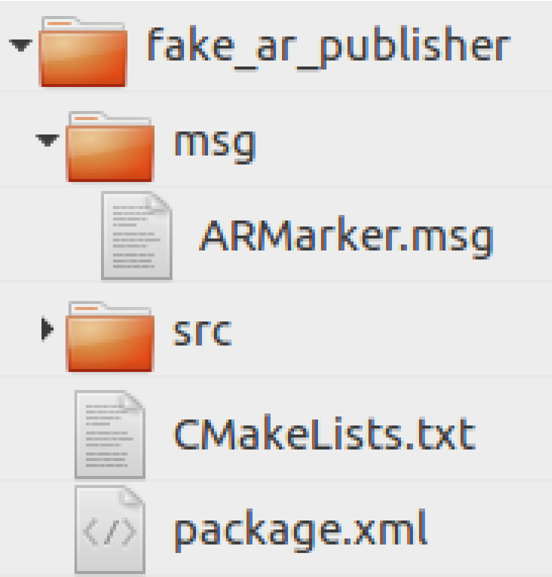
data type

field name

Custom ROS Messages

- Custom message types are defined in `msg` subfolder of packages

- Modify `CMakeLists.txt` to enable message generation.



```
fake_ar_publisher
  msg
    ARMarker.msg
  src
  CMakeLists.txt
  package.xml
```

CMakeLists.txt

- Lines needed to generate custom msg types

```
find_package(catkin REQUIRED COMPONENTS
 message_generation)

add_message_files(custom.msg ...)

generate_messages(DEPENDENCIES ...)

catkin_package(CATKIN_DEPENDS roscpp
message_runtime)
```

package.xml

<build_depend> **message_generation** </build_depend>

<build_export_depend>**message_runtime**</build_export_depend>

<run_depend>**message_runtime**</run_depend>

ROS Message Commands

- `rosmsg list`
  - Show all ROS topics currently installed on the system
- `rosmsg package <package>`
  - Show all ROS message types in package `<package>`
- `rosmsg show <package>/<message_type>`
  - Show the structure of the given message type

ROS Topic Commands

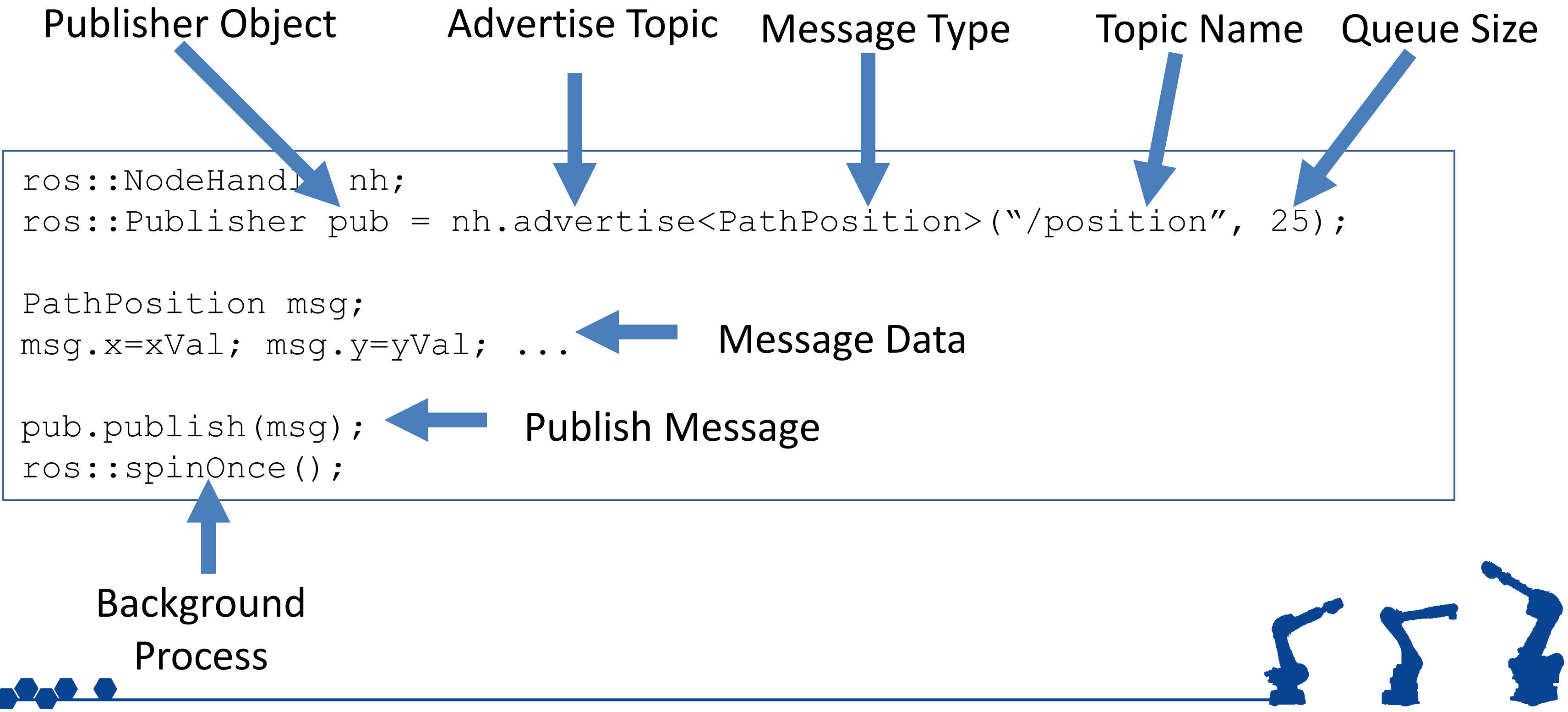
- `rostopic list`
  - List all topics currently subscribed to and/or publishing
- `rostopic type <topic>`
  - Show the message type of the topic
- `rostopic info <topic>`
  - Show topic message type, subscribers, publishers, etc.
- `rostopic echo <topic>`
  - Echo messages published to the topic to the terminal
- `rostopic find <message_type>`
  - Find topics of the given message type

“Real World” – Messages

- Use *rqt_msg* to view:
  - `sensor_msgs/JointState`
  - `trajectory_msgs/JointTrajectory`
  - `sensor_msgs/Image`
  - `rosgraph_msgs/Log`

Topics: Syntax

- Topic **Publisher**
  - Advertises available topic *(Name, Data Type)*
  - Populates message data
  - Periodically publishes new data



Publisher Object

Advertise Topic

Message Type

Topic Name

Queue Size

```
ros::NodeHandle nh;
ros::Publisher pub = nh.advertise<PathPosition>(“/position”, 25);

PathPosition msg;
msg.x=xVal; msg.y=yVal; ...

pub.publish(msg);
ros::spinOnce();
```

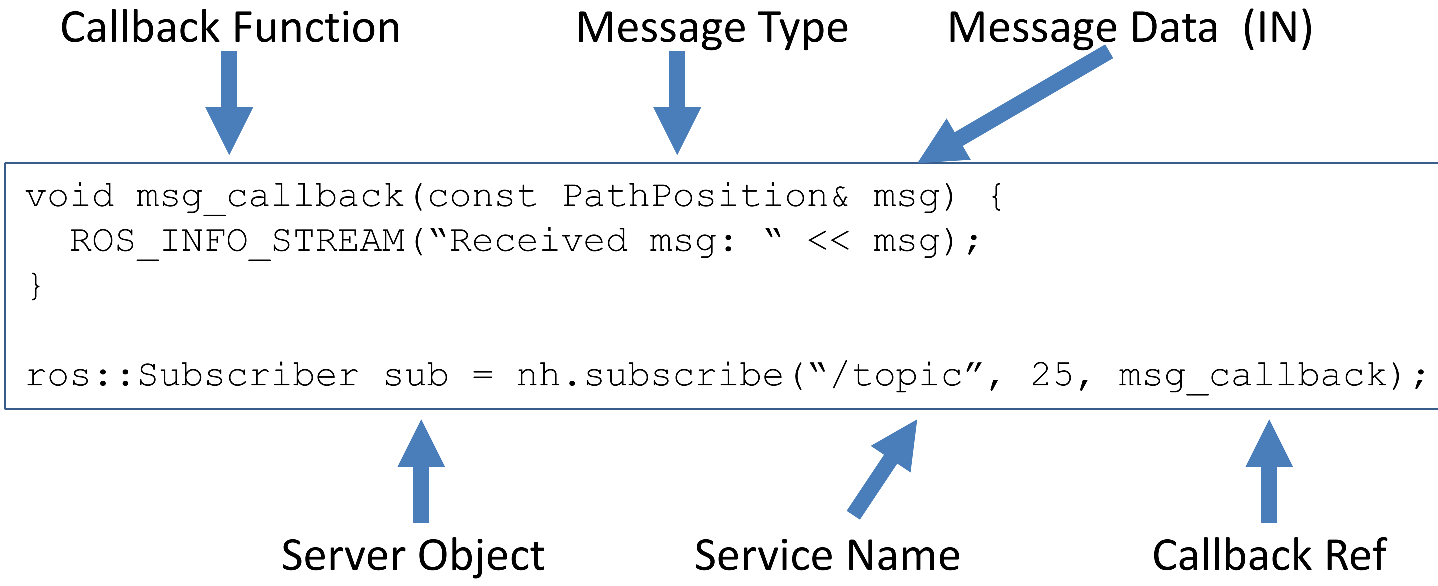
Message Data

Publish Message

Background Process

Topics: Syntax

- Topic **Subscriber**
  - Defines callback function
  - Listens for available topic *(Name, Data Type)*



```
void msg_callback(const PathPosition& msg) {
  ROS_INFO_STREAM(“Received msg: “ << msg);
}

ros::Subscriber sub = nh.subscribe(“/topic”, 25, msg_callback);
```

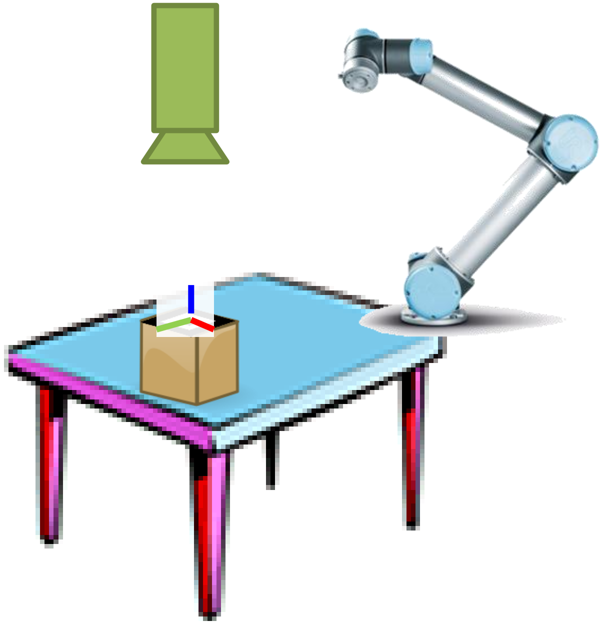
Qt

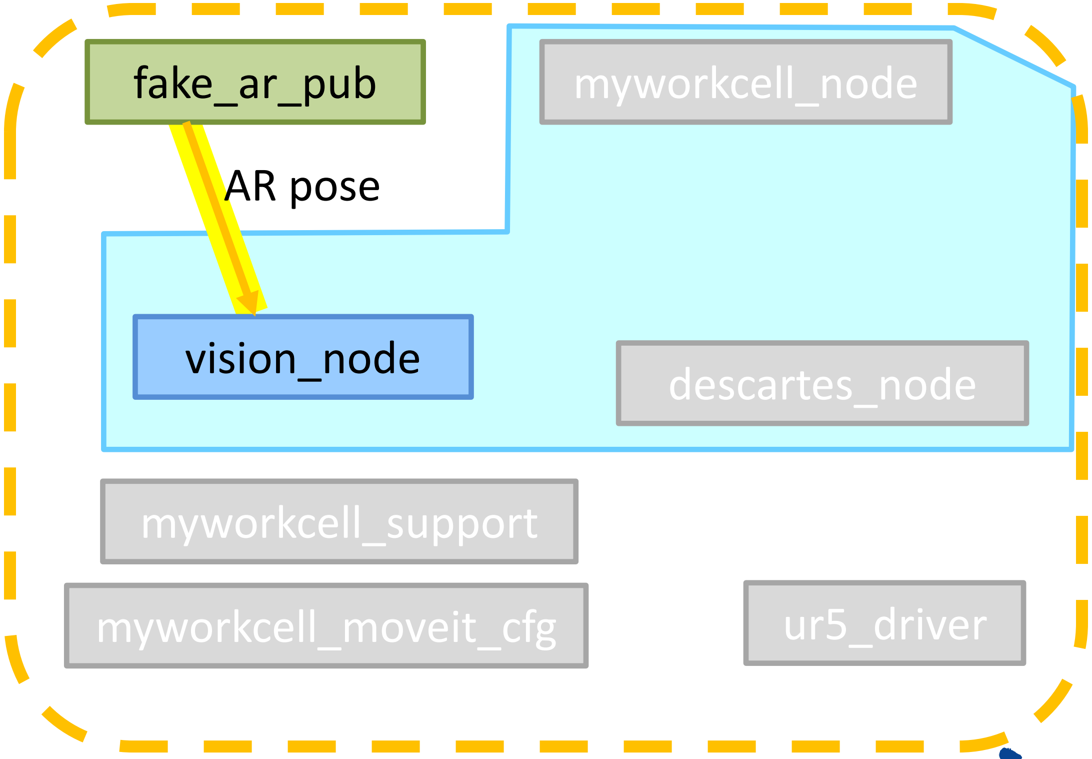
Instead of text editor and building from terminal…

Use an IDE! Wiki instructions here

Exercise 1.4

Subscribe to fake_ar_publisher




fake_ar_pub
AR pose
vision_node
myworkcell_node
descartes_node
myworkcell_support
myworkcell_moveit_cfg
ur5_driver