

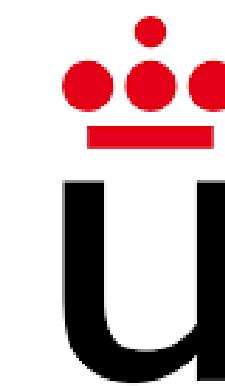


Course of  
Robot Programming  
with ROS 2

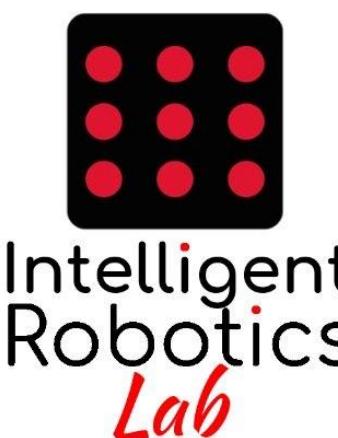
Day 1

## 1. Introduction

ikerlan



Universidad  
Rey Juan Carlos



**Pinned**

- [PlanSys2/ros2\\_planning\\_system](#) (Public) Forked from ros-planning/ros2\_planning\_system. This repo contains a PDDL-based planning system for ROS2.
- [ros2\\_dotnet](#) (Public) Forked from ros2-dotnet/ros2\_dotnet. .NET bindings for ROS2.
- [cascade.lifecycle](#) (Public) Managed nodes (or lifecycle nodes, LN) package provides a mechanism to create LifecycleNode activation trees.
- [navigation2](#) (Public) Forked from ros-planning/navigation2. ROS2 Navigation.
- [rclcpp](#) (Public) Forked from ros2/rclcpp. rclcpp (ROS Client Library for C++)
- [popf](#) (Public) The POPF planner from KCL planning group with some modifications to make it work with "modern" compilers...

**Achievements**

- GitHub Octocat
- GitHub Octocat
- GitHub Octocat
- GitHub Octocat x2
- GitHub Octocat x3

**Highlights**

**Organizations**

**Contribution activity**

697 contributions in the last year

Contribution settings ▾

Less More

Learn how we count contributions

Mon Jun Jul Aug Sep Oct Nov Dec Jan Feb Mar Apr May

Wed

Fri

2023

2022

2021

2020

2019

2018

2017

2016

2015

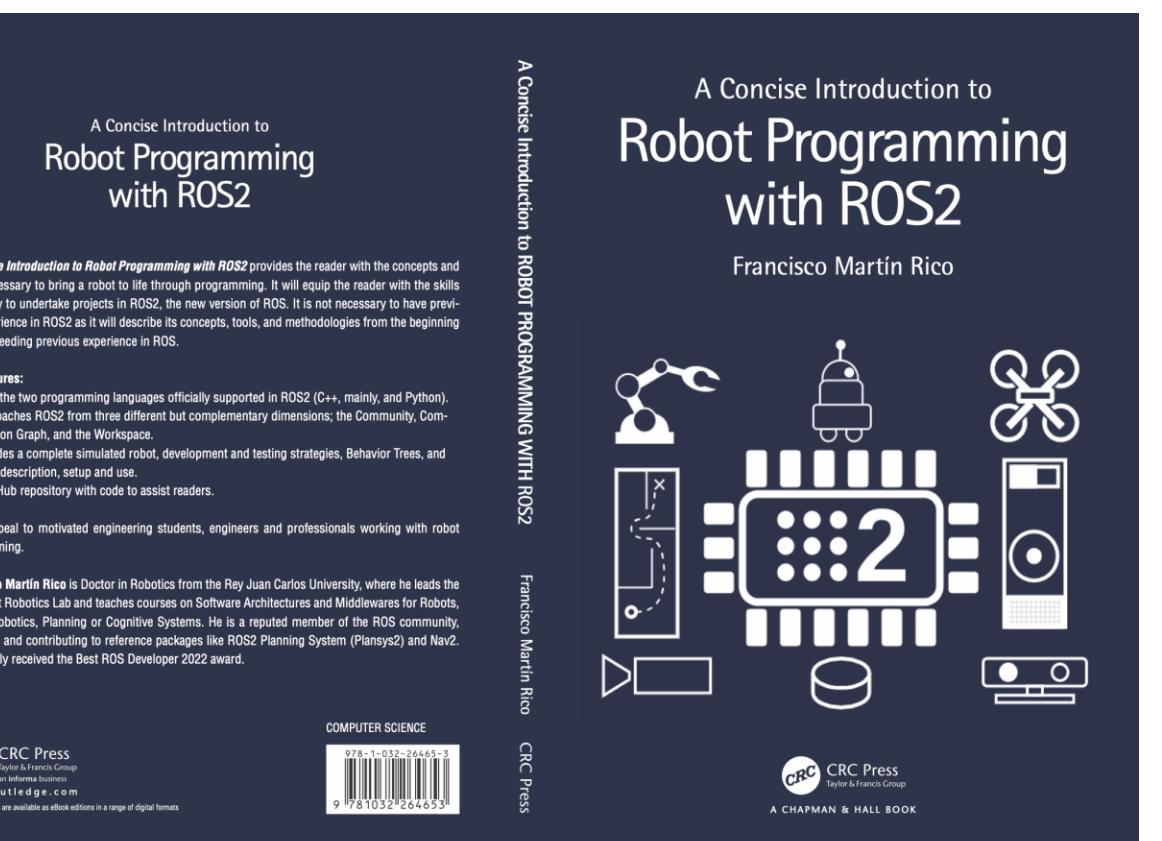
fmrico has no activity yet for this period.

Show more activity

Seeing something unexpected? Take a look at the GitHub profile guide.



 Universidad  
Rey Juan Carlos



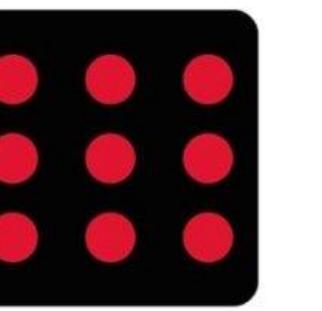
2

TSC

Intelligent  
Robotics  
Lab



José Miguel Guerrero  
Profesor CD  
Visión Artificial / Virtualización



Intelligent  
Robotics  
*Lab*



Rodrigo Pérez  
Profesor AyD  
Rob. Asistencial / Behavior Trees



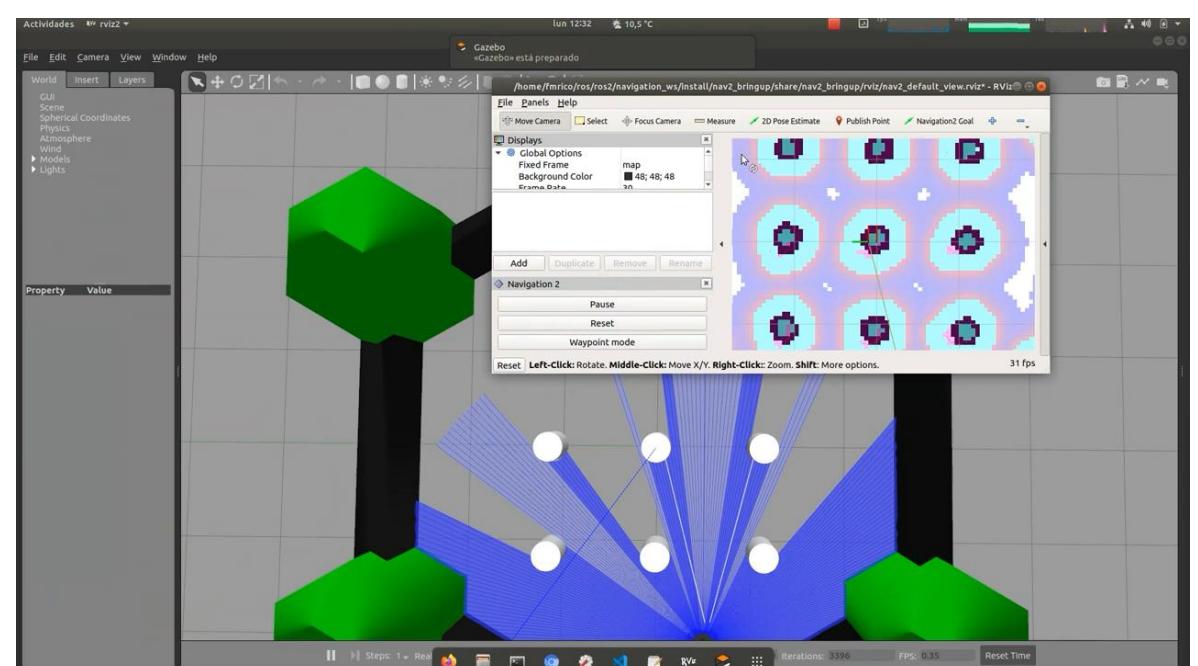
Juan Diego Peña  
PhD Std / Inv. Contratdo  
Mobile Robots / Network



Alberto García  
PhD Std / Inv. Contratdo  
Mobile Robots / Navigation



Juan Carlos Manzanares  
Técnico de Laboratorio  
Mobile Robots / Navigation



## PlanSys<sup>:::</sup>2

IROS 2021  
A PLANNING SYSTEM  
FRAMEWORK

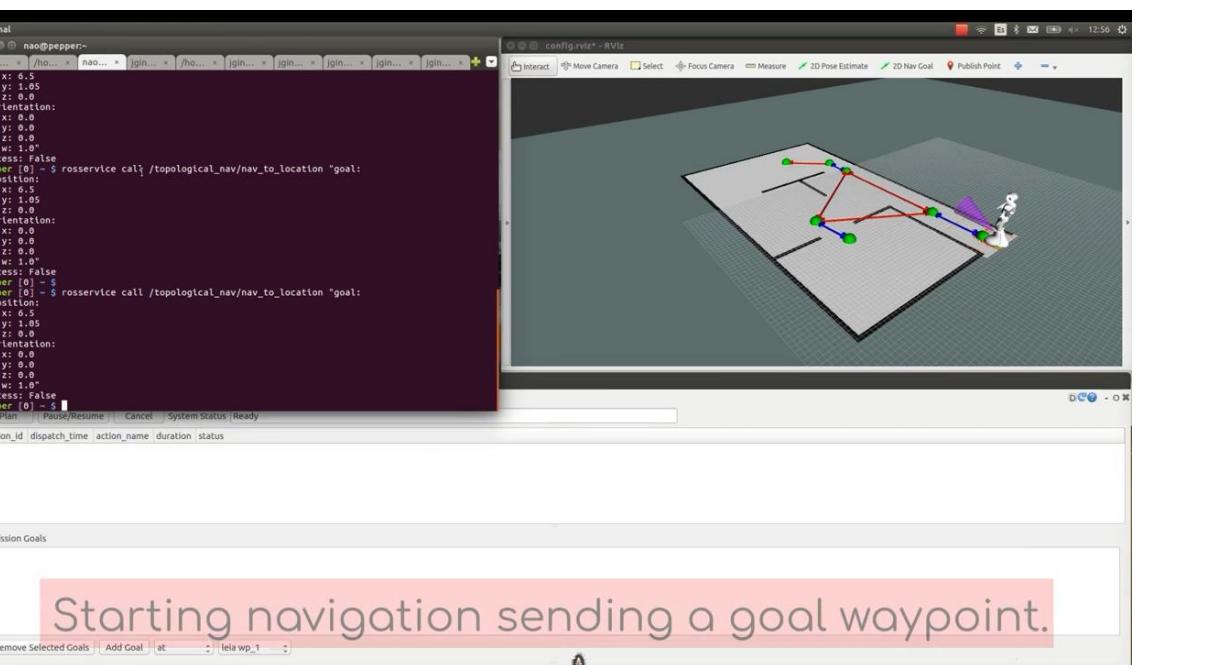
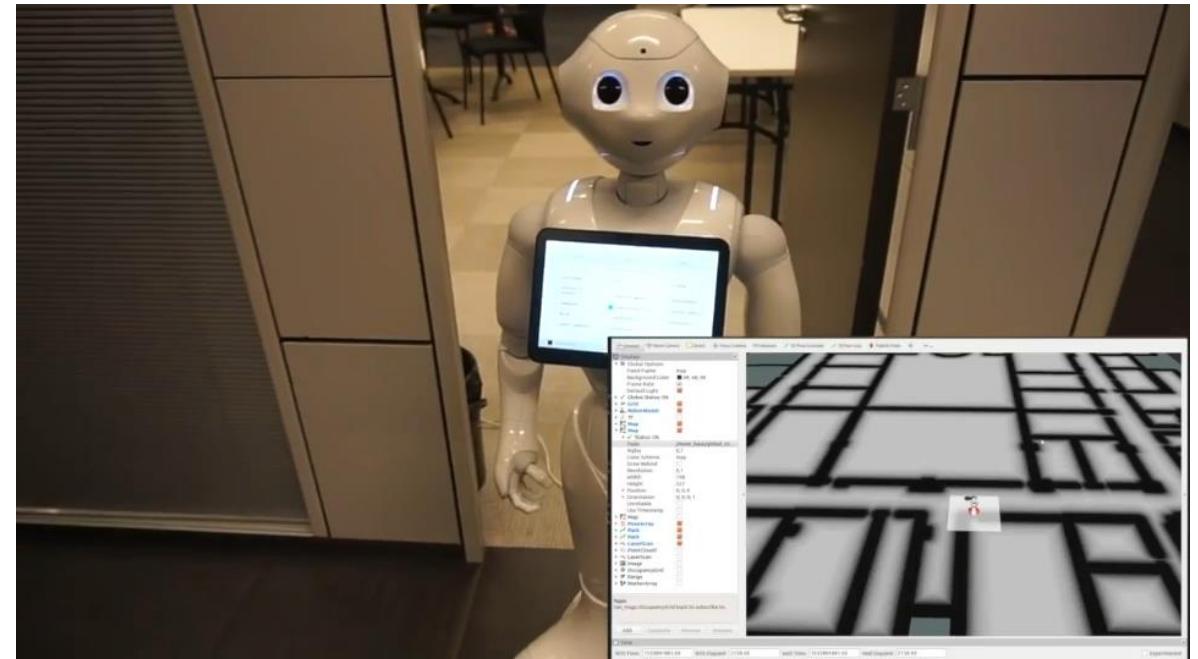
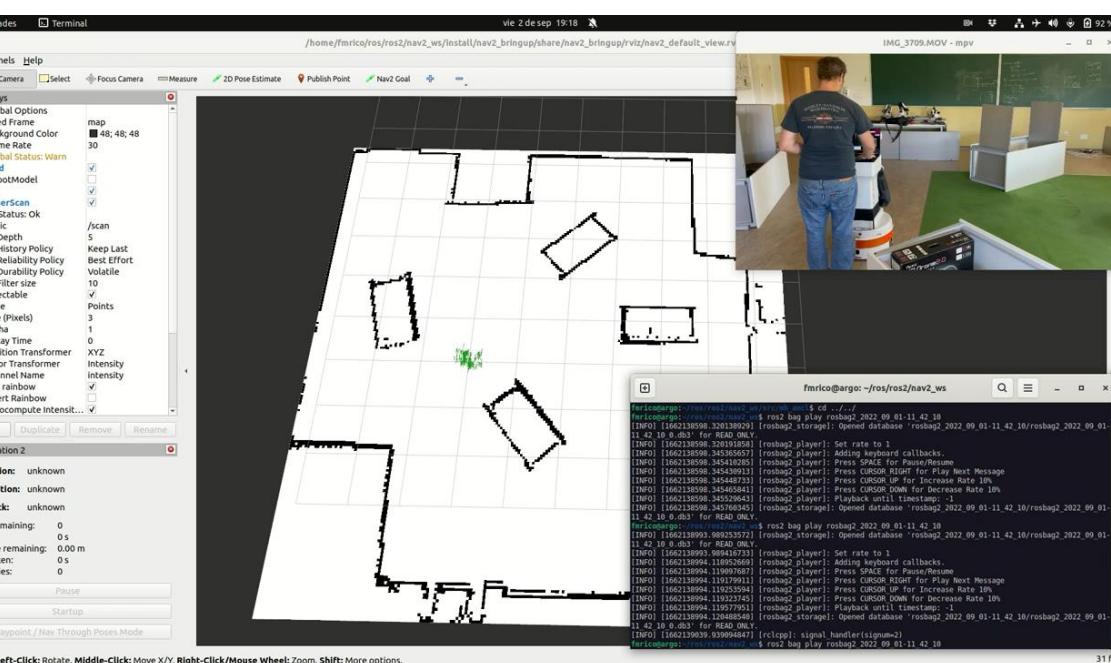
FRANCISCO MARTÍN, JONATAN GINÉS, VICENTE MATELLÁN, AND FRANCISCO J. RODRÍGUEZ



IROS 2021

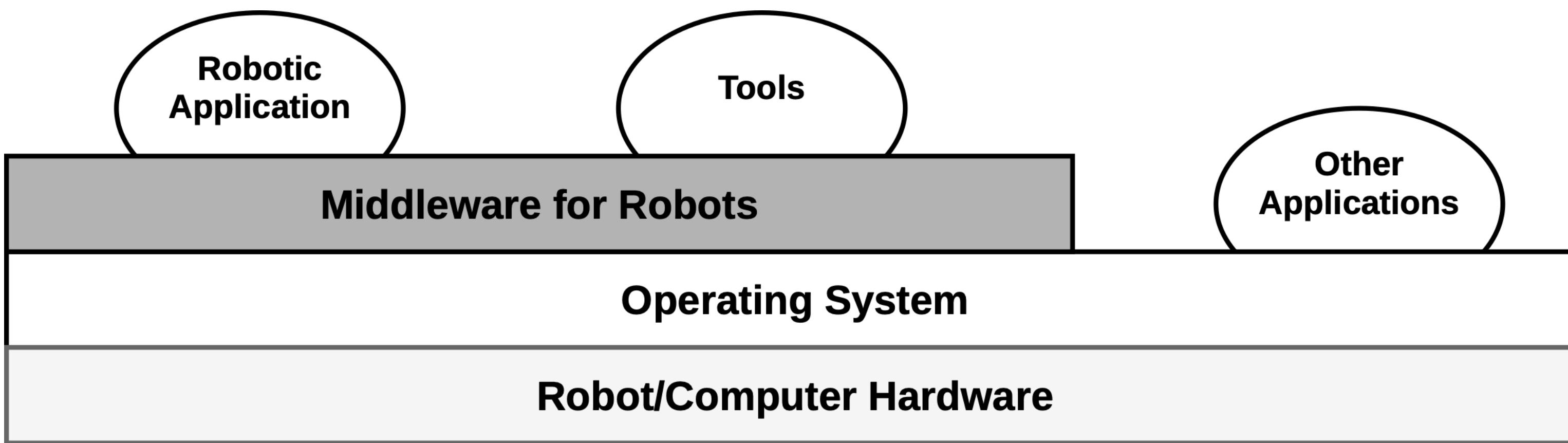
### REGULATED PURE PURSUIT FOR ROBOT PATH TRACKING

STEVE MACENSKI, SHRIJIT SINGH, FRANCISCO MARTÍN AND JONATAN GINES



# Programming Robots

- Robots must be programmed to be useful
- We need Middlewares
- Robot programming middlewares provide drivers, libraries, and methodologies
- Few of them have survived the robot for which they were designed or have expanded from the laboratories where they were implemented
- The big difference is the ROS developers community around the world.



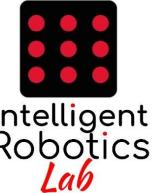
# ROS (Robot Operating System)

*“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.”*

<http://www.ros.org/>



Open Source Robotics Foundation

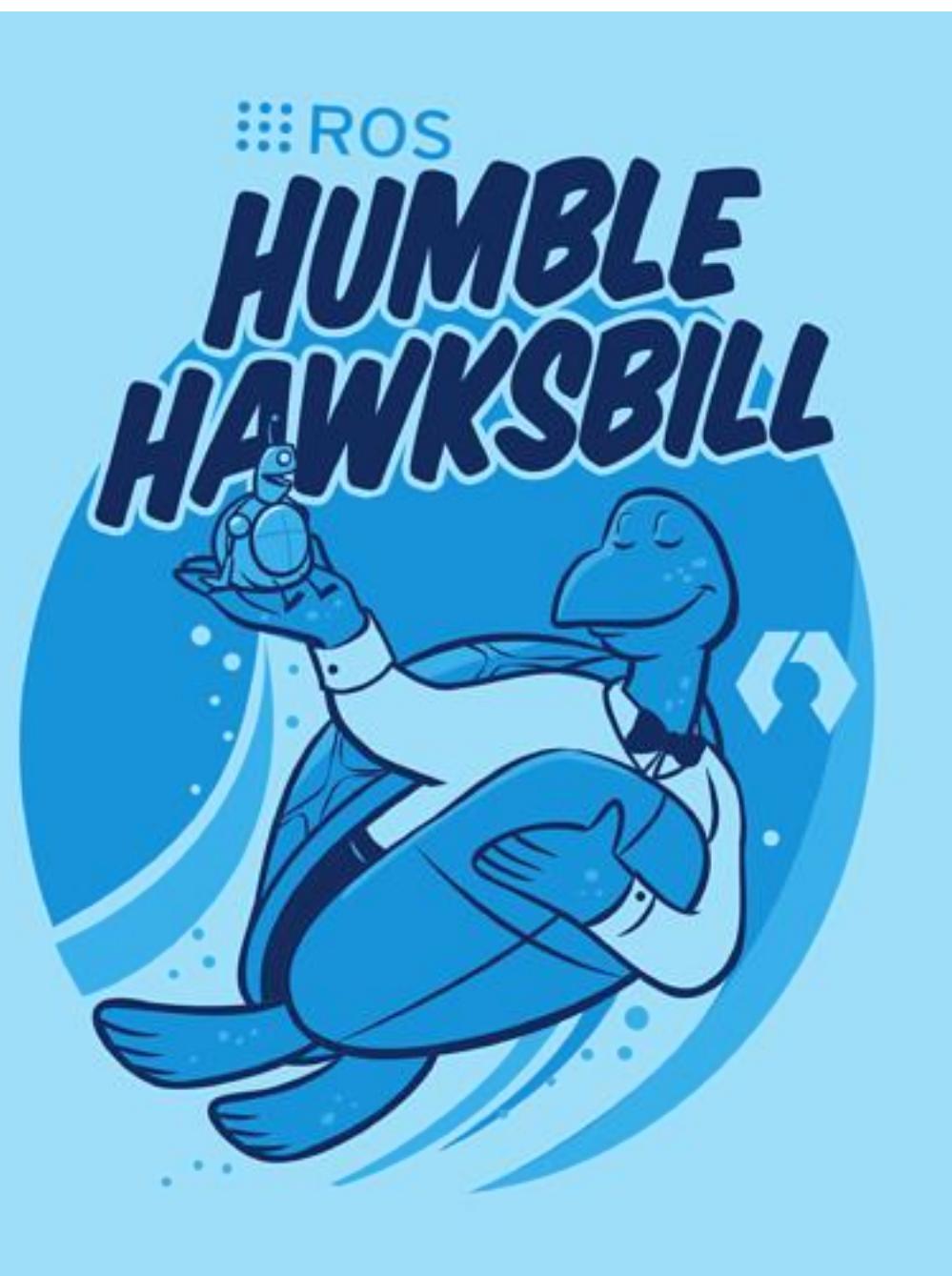


- ROS and ROS2
- Lot of tutorials and documentation
- We will use Ubuntu 22.04 + Humble



ROS

2



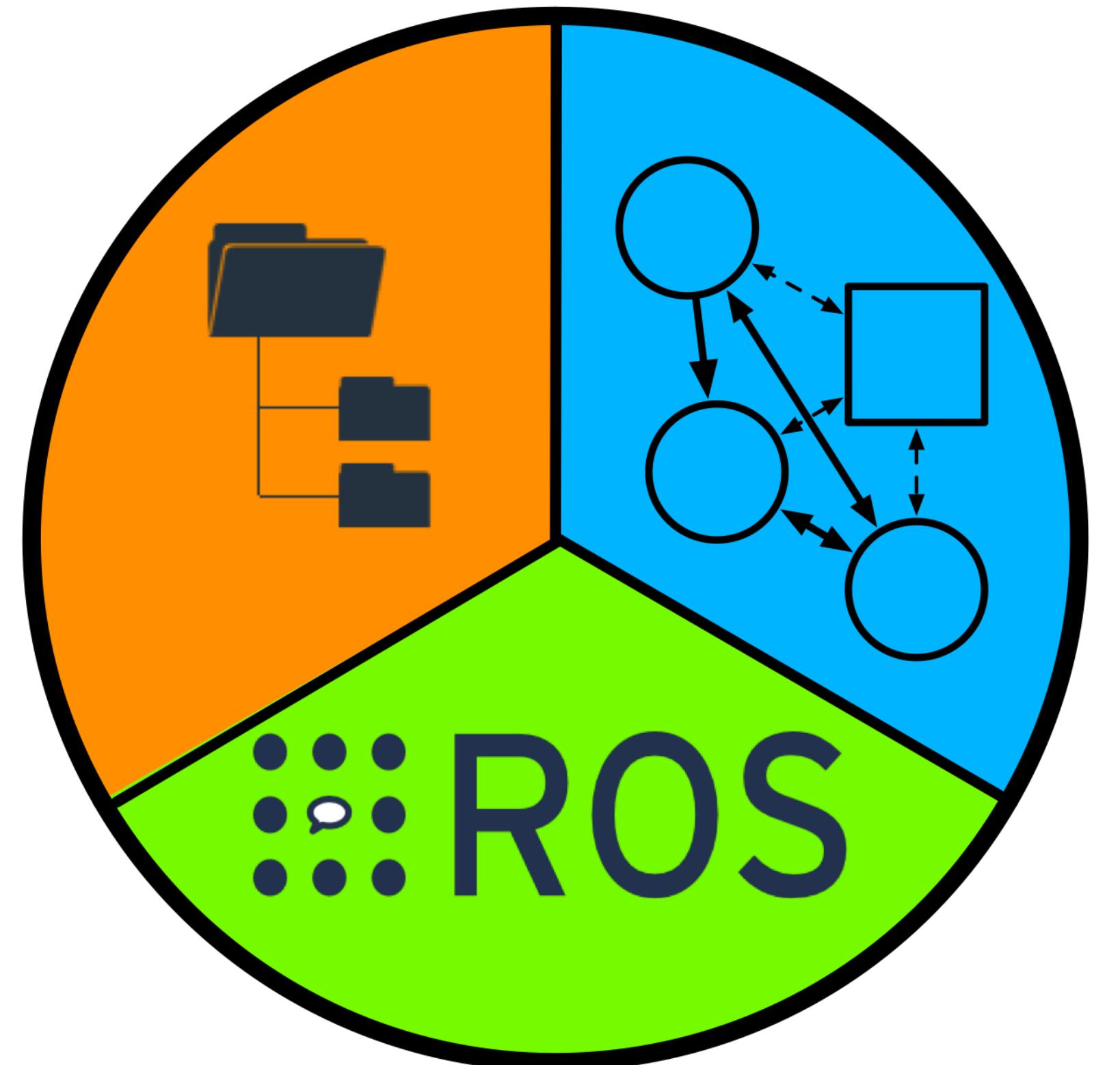
# History

- ROS was born in 2006 at Stanford, within STAIR project
- Willow Garage in 2007
- Open Source Robotics Foundation in 2013
- Since 2013 is considered the standard in Robotics



# ROS Dimensions

**Workspace:** the set of software installed on the robot or computer, the programs that the user develops, and tools to build

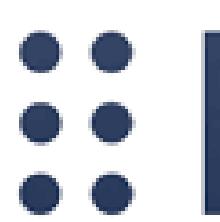


**Computation Graph:**  
a running ROS2 application

**Community:** vast community of developers who contribute with their own applications and utilities through public repositories, to which other developers can contribute

# The Community

- Open Source and Licenses
- ROS2 organizes software development in federal model
- Packages and distributions
- Online resources

 ROS

 Open Source Robotics Foundation



MIT LICENSE    GNU LICENSE

BSD-2

**OPEN SOURCE LICENSE**

APACHE LICENSE 2.0

BSD-3



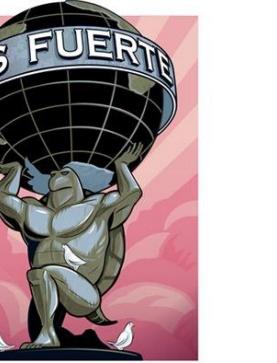
# ROS Distributions



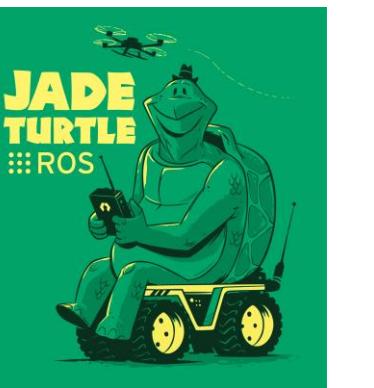
:::Box Turtle



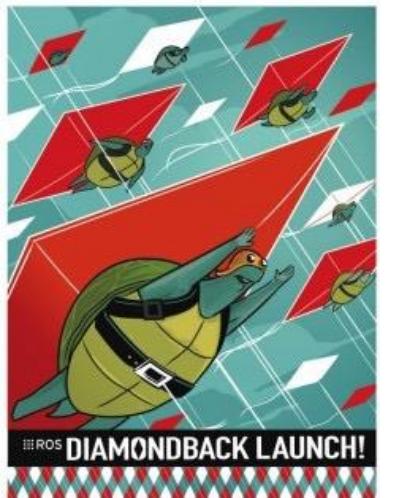
Mar 2, 2010



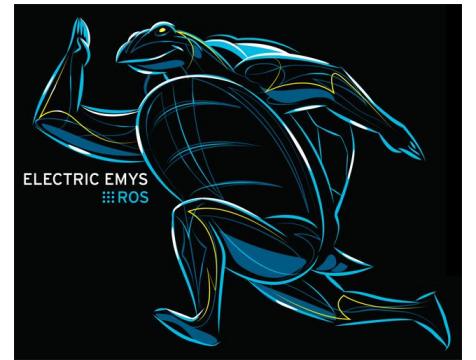
Apr 23, 2012



May 23rd, 2015



Mar 2, 2011



Aug 30, 2011



Sept 4th, 2013



Jul 22nd, 2014



May 23rd, 2016



May 23rd, 2017

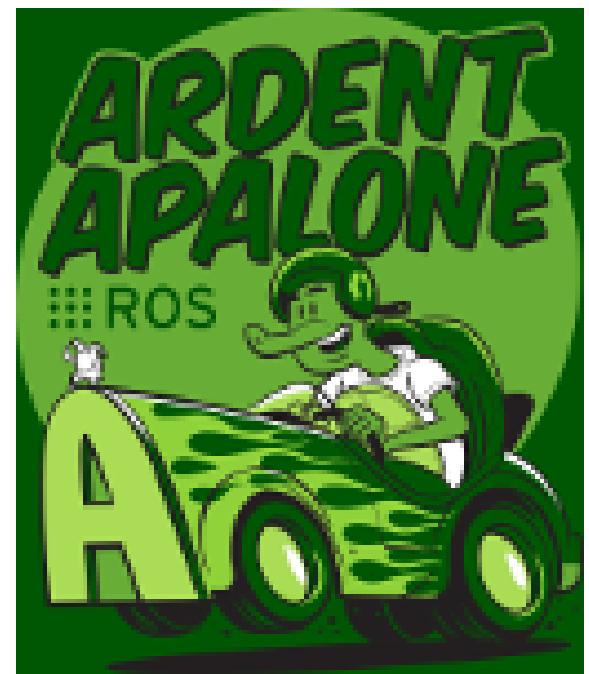


May 23rd, 2018



May 23rd, 2020

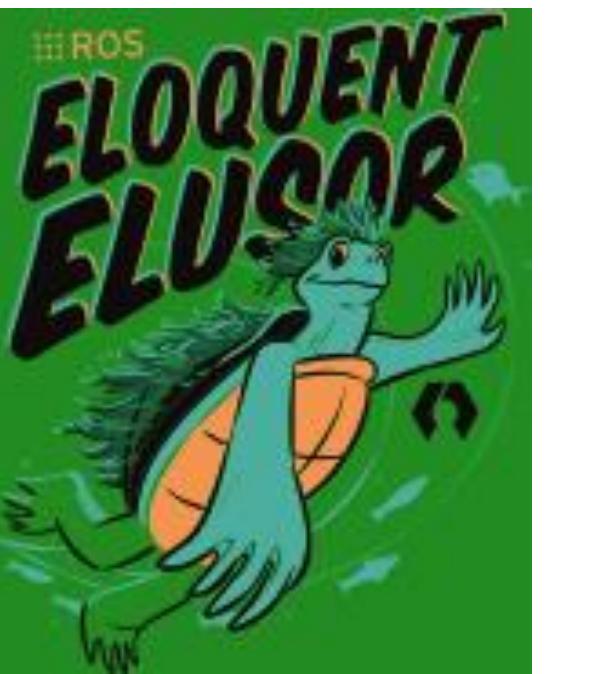
# ROS 2 Distributions



Dec 8th, 2017



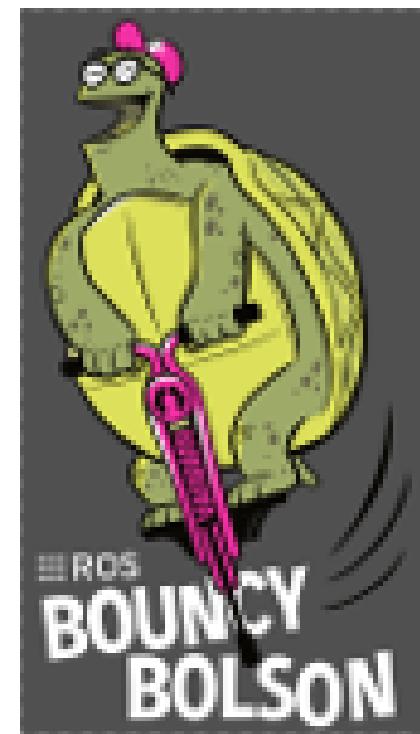
Dec 14th, 2018



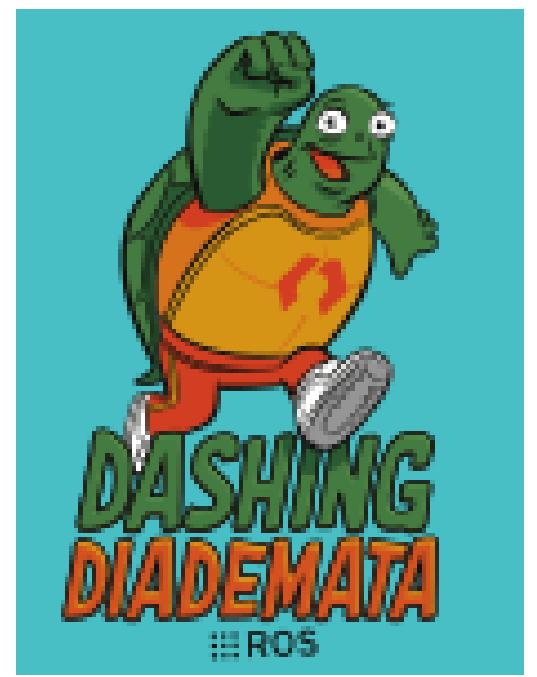
Nov 22nd, 2019



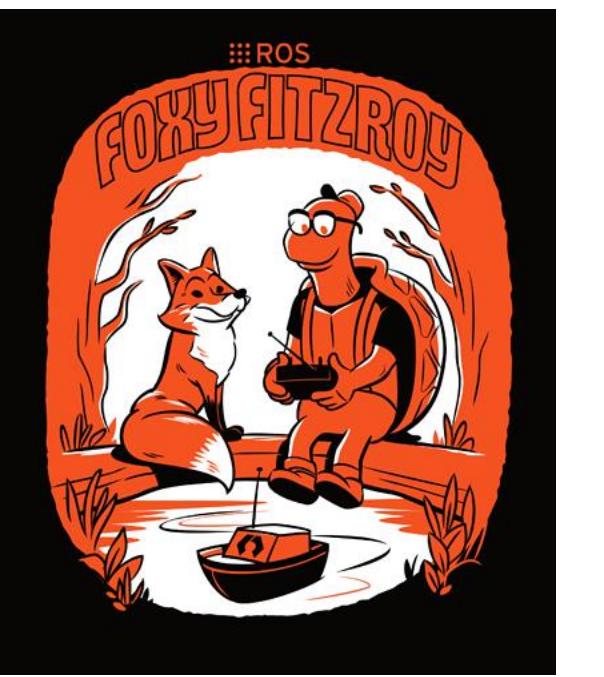
May 23rd, 2021



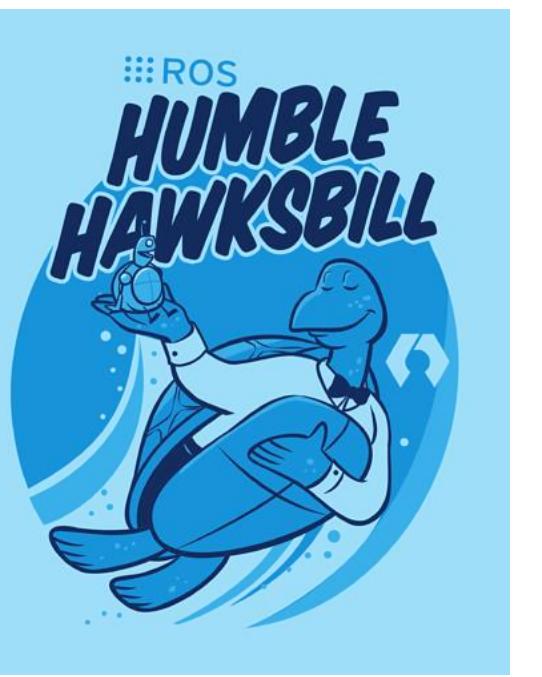
July 2nd, 2018



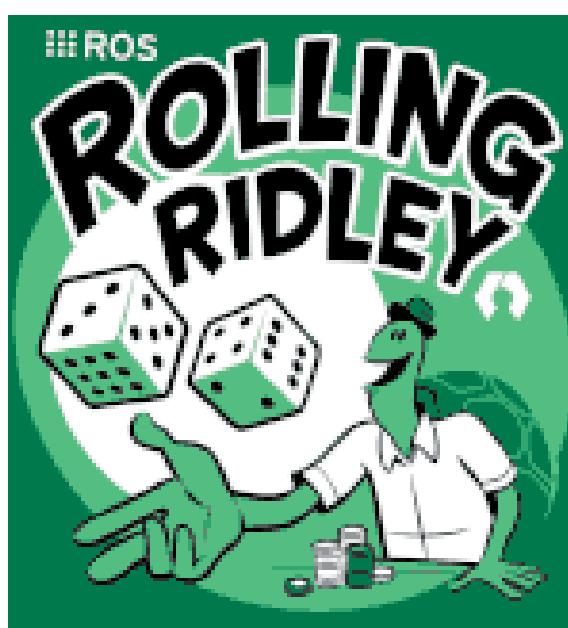
May 31st, 2019



Jun 5th, 2020



May 23rd, 2022



# ROS Resources

- [ROS.org \[link\]](#)
- [Distributions \[link\]](#)
- [Package documentation \[link\]\[link\]](#)
- [ROS Wiki \[link\]](#)
- [ROS Answers \[link\]](#)
- [ROS Discourse \[link\]](#)
- [Blog \[link\]](#)

**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 Melodic Morenia**  
Melodic Morenia is the 12th official ROS release. It is supported on Ubuntu Artful and Bionic, along with Debian Stretch. Get Melodic Morenia now!

[Download](#)

**ROS Kinetic Kame**  
Kinetic Kame is the 10th official ROS release. It is supported on Ubuntu Wily and Xenial. Get Kinetic Kame now!

[Download](#)

**Wiki** Find tutorials and learn more

**ROS Answers** Ask questions. Get answers

**Blog** Get the latest news

**ROS Discourse**

[ROS Resources: Documentation | Support | Discussion Forum | Service Status | Q&A answers.ros.org](#)

[all categories](#) [all tags](#) [Categories](#) [Latest](#) [New \(1\)](#) [Unread \(5\)](#) [Top](#) [+ New Topic](#)

Category	Topics	Latest
<b>General</b>	896	1 unread
<b>ROS</b>	896	0
<b>Autoware</b>	171	1 unread
<b>Next Generation ROS</b>	469	3 unread
<b>Quality Assurance</b>	62	1 new

**ROS ANSWERS**

[tags](#) [users](#) [badges](#) [Hi there! Please sign in](#) [help](#)

[ASK YOUR QUESTION](#)

**TF Lookup would require extrapolation into the future**

**4** [tf](#) [robot\\_localization](#) [gps](#)

I would appreciate some guidance to understand what may be going on here.

asked Jul 28 '14 [jordui](#) 93 2 5 8

updated Aug 22 '14 [Munro](#) 786 5 9 20

When running my configuration with sensor data coming from a recorded bag file, I get the following error from the `ekf_localization_node` node:

```
[WARN] [1406560584.464395417, 1406223582.586891240]: Could not obtain transform from utm to nav. Error was Lookup would require extrapolation into the future. Requested time 1406223583.213800000 but the latest data is at time 1406223582.576822606, when looking up transform from frame [utm] to frame [nav]
```

I've been trying to understand why this is happening and I've gathered the following information:

- The CPU usage is pretty low. I don't think this could be happening because of lack of processing capacity
- What the code seems to be trying to do is using `lookupTransform` to calculate TF using the timestamp of the GPS message received
- The transformation between [utm] and [nav] is being published at 50Hz. GPS messages are being published at 1Hz.
- I can see GPS messages being published with timestamps matching the warning messages. I can also see TF publishing transformations matching the timestamp of "the latest data is at time..." but also many more later messages:

```
stamp: 1406223582
secs: 1406223582
nsecs: 576022606
frame_id: nav
child_frame_id: utm
transform:
translation:
```

Is there a `tfoMessageFilter` in the `tf` Python API?

**ROS.org**

[About](#) [Support](#) [Discussion Forum](#) [Service Status](#) [Q&A answers.ros.org](#)

[Search](#) [Submit](#)

[Documentation](#) [Browse Software](#) [News](#) [Download](#)

**navigation**

[indigo](#) [kinetic](#) [lunar](#) [melodic](#) [Show EOL distros](#)

**Documentation Status**

**navigation**: amcl | base\_local\_planner | cartesian\_planner | clear\_costmap\_recovery | costmap\_2d | diwa\_local\_planner | fake\_localization | global\_planner | map\_server | move\_base | move\_base\_msgs | move\_slow\_and\_clear | nav\_core | navfn | rotate\_recovery | voxel\_grid

**Package Summary**

[Released](#) [Continuous Integration](#) 91 / 91 [Documented](#)

A 2D navigation stack that takes in information from odometry, sensor streams, and a goal pose and outputs safe velocity commands that are sent to a mobile base.

- Maintainer status: maintained
- Maintainer: Michael Ferguson <mferng7 AT gmail DOT com>, David V. Luu <davidvluu AT gmail DOT com>, Aaron Hey <ahey AT fetchrobotics DOT com>
- Author: contradict@gmail.com, Ethan Marder-Eppstein
- Licenses: BSD, GPL, LGPL (amcl)
- Source: git <https://github.com/ros-planning/navigation.git> (branch: melodic-devel)

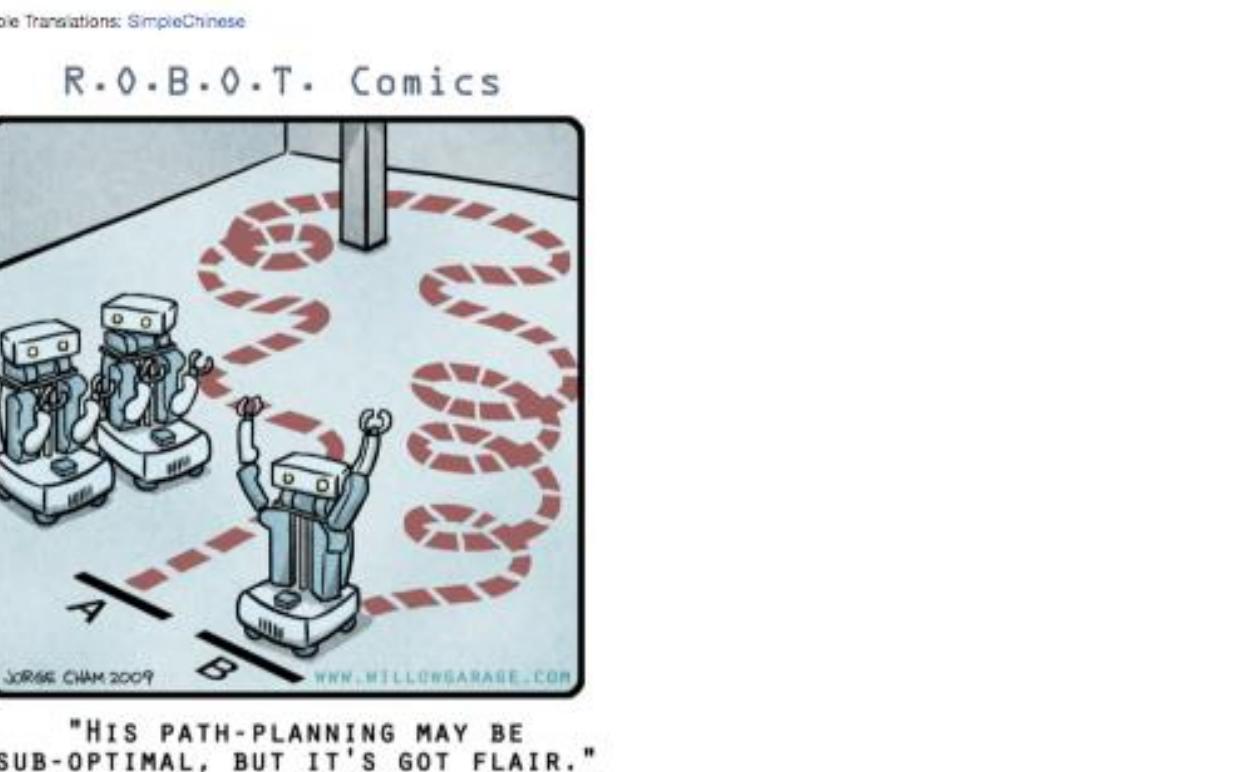
**Package Links**

Tutorials | Troubleshooting | FAQ | ChangeLog | Change List | Reviews | Dependencies (17) Jenkins jobs (9)

**Página**

Página Inicial | Información | Adjuntos | [Mis Acciones](#) | [Usuario](#)

Ingresar



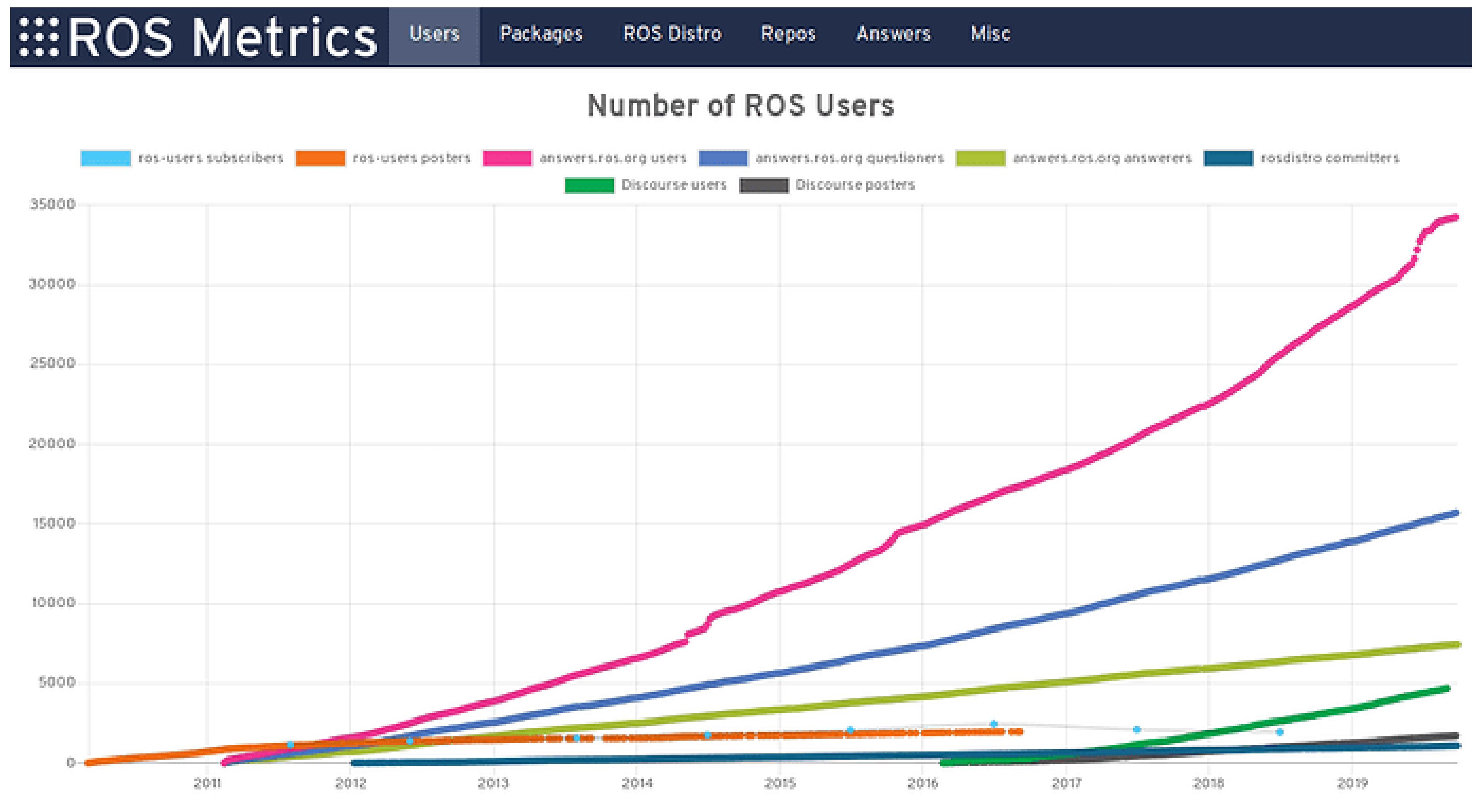
**1. Overview**

The Navigation Stack is fairly simple on a conceptual level. It takes in information from odometry and sensor streams and outputs velocity commands to send to a mobile base. Use of the Navigation Stack on an arbitrary robot, however, is a bit more complicated. As a pre-requisite for navigation stack use, the robot must be running ROS, have a tf transform tree in place, and publish sensor data using the correct ROS Message types. Also, the Navigation Stack needs to be configured for the shape and dynamics of a robot to perform at a high level. To help with this process, this manual is meant to serve as a guide to typical Navigation Stack set-up and configuration.



# ROS Metrics

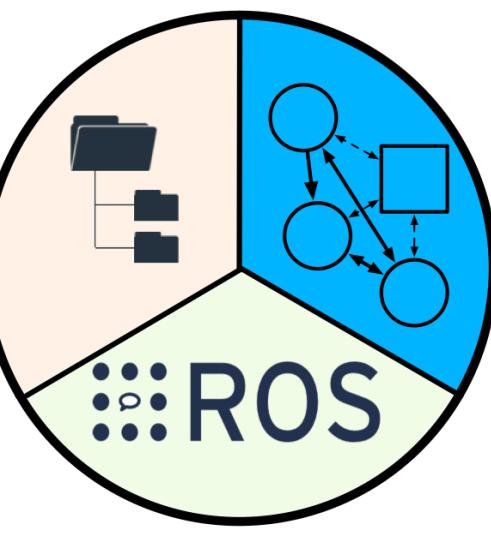
<https://metrics.ros.org/>



ROS Robots

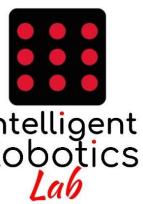
<https://robots.ros.org/>

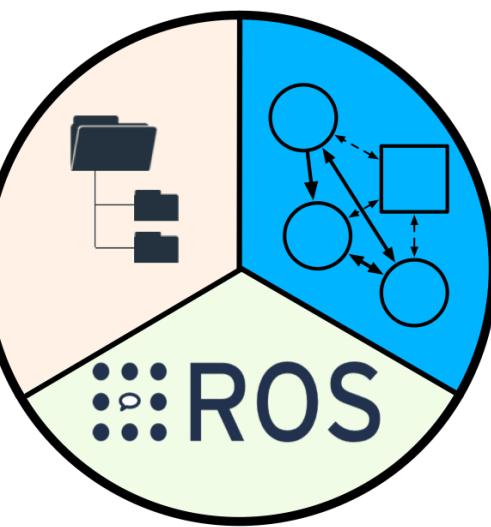




# The Computation Graph

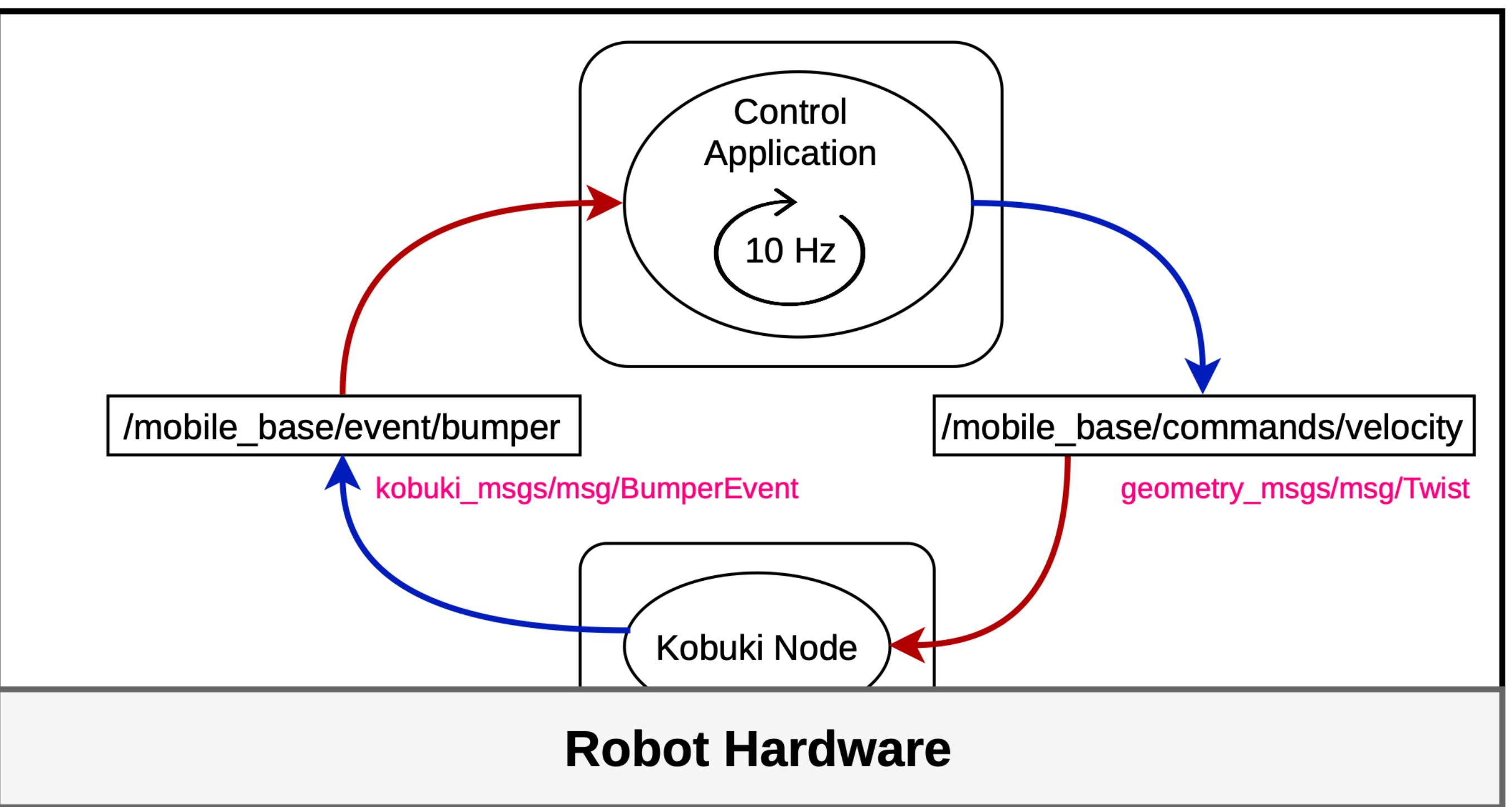
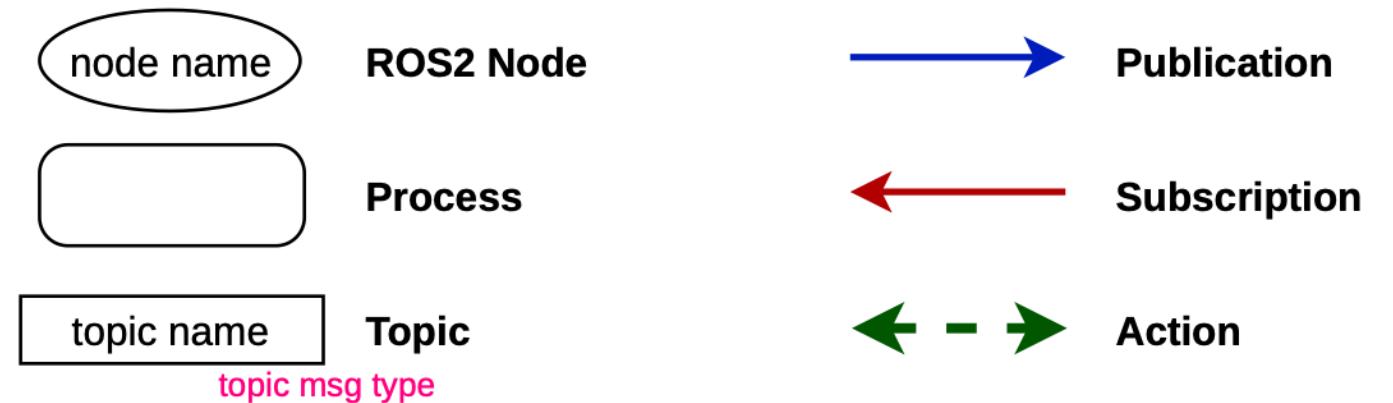
- A robot's software looks like during its execution.
- A Computation Graph contains ROS2 nodes that communicate with each other so that the robot can carry out some tasks.
- The logic of the application is in the nodes, as the primary elements of execution in ROS2.
- Communication mechanisms:
  - **Publication/Subscription:** Asynchronous N:M
  - **Services:** Synchronous 1:1
  - **Actions:** Asynchronous 1:1

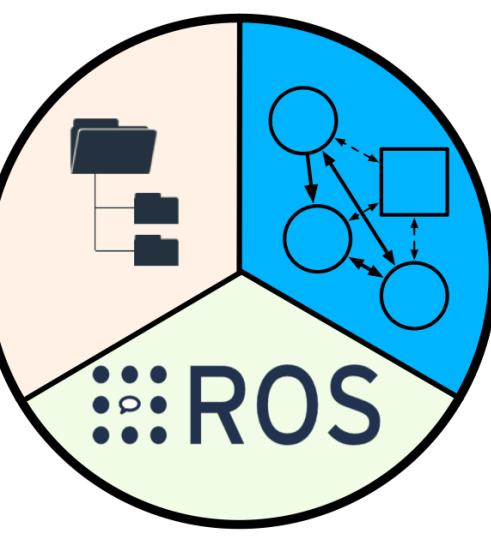




# The Computation Graph

- A robot's software looks like during its execution.
- A Computation Graph contains ROS2 nodes that communicate with each other so that the robot can carry out some tasks.
- The logic of the application is in the nodes, as the primary elements of execution in ROS2.
- Communication mechanisms:
  - **Publication/Subscription:**  
Asynchronous N:M
  - **Services:** Synchronous 1:1
  - **Actions:** Asynchronous 1:1
- Execution model
  - **Iterative**
  - **Event-Oriented**





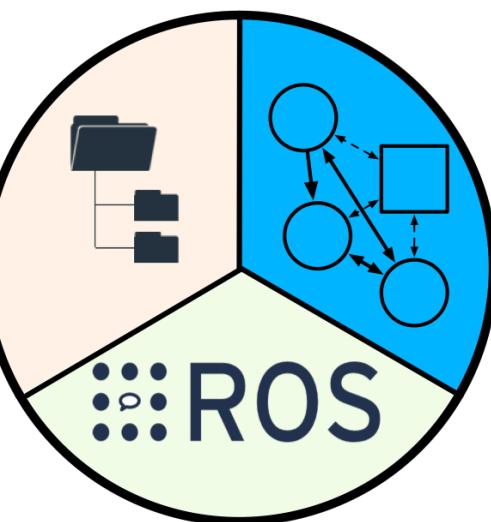
# The Computation Graph

## About names

- Resources in ROS2 follow a name convention
- The name of a resource depends on:
  - The type or name: relative, absolute or private
  - The node name
  - The namespace

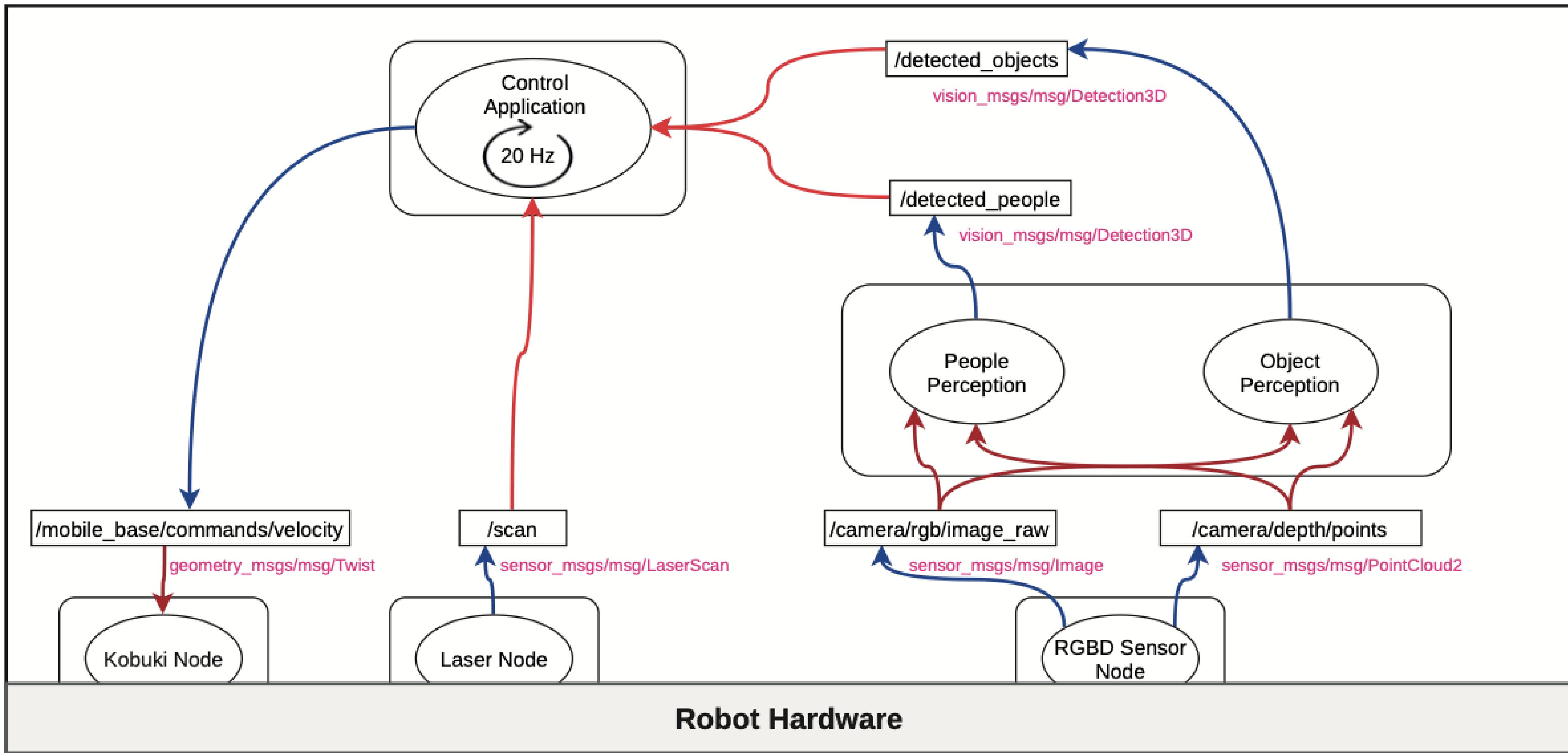
name	Result: (node: my_node / ns: none)	Result: (node: my_node / ns: my_ns)
my_topic	/my_topic	/my_ns/my_topic
/my_topic	/my_topic	/my_topic
~my_topic	/my_node/my_topic	/my_ns/my_node/my_topic





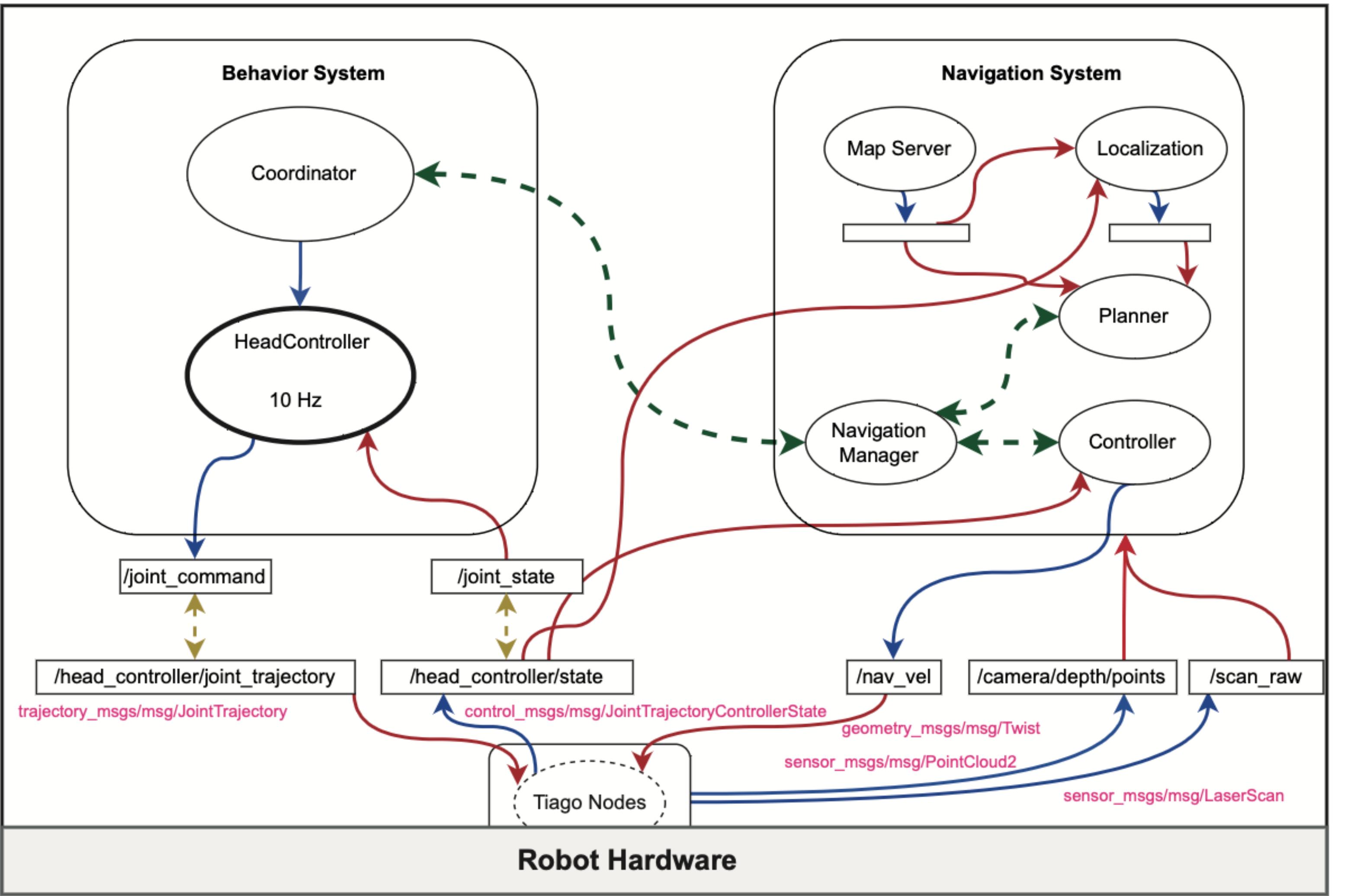
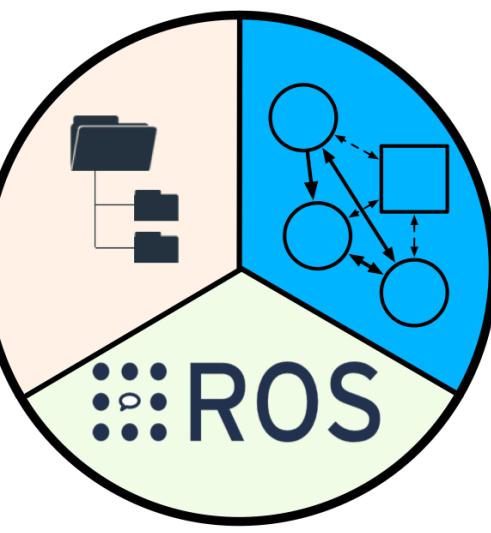
# The Computation Graph

## Examples

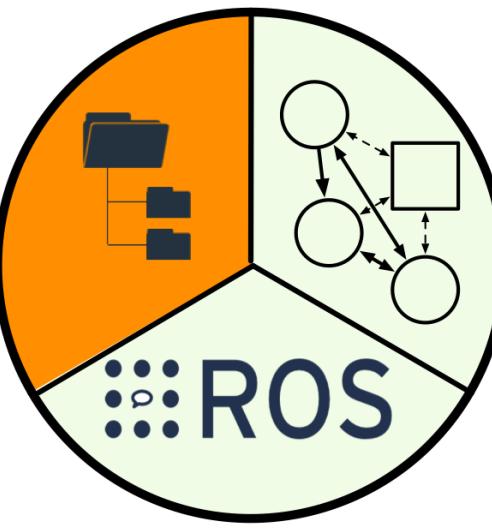


# The Computation Graph

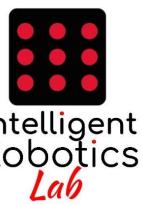
## Examples

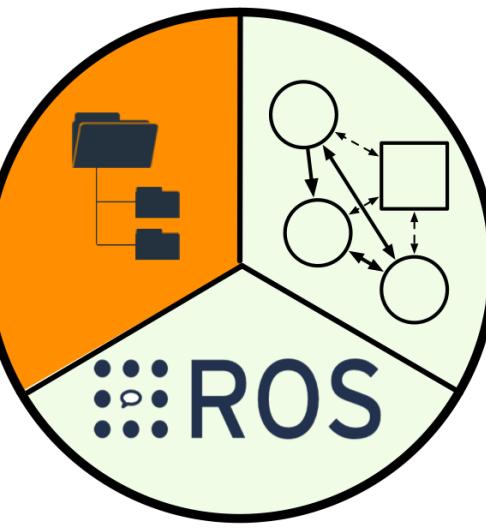


# The Workspace



- Approaches ROS2 software from a static point of view.
- Where the ROS2 software is installed, organized, and all the tools and processes that allow us to launch a computing graph.
- This includes the build system and node startup tools.
- Elements:
  - **Package:**
    - It is the minimum functional set of software.
    - Contains executables, libraries, or message definitions with a common purpose.
  - **Workspace:**
    - A directory that contains packages.
    - Activable to be available to use.
  - ***Underlay* y *overlay***





# The Workspace

## Set up the user workspace

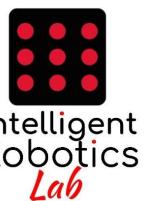
- Build the workspace

```
$ cd ~/ikerlan_ws  
$ colcon build -simlink-install
```

- Build, install, and log
- Activate the user workspace

```
$ source ~/ikerlan_ws/install/setup.bash
```

```
$ echo "source ~/ikerlan_ws/install/setup.bash" >> ~/.bashrc
```



# ROS2 Design

