



Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen
International Center for
Information Technology

Introduction to ROS

Foundation Course

August 27, 2019

Hassan Umari

1. What is ROS?

1.1 What ROS is

1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

3.1 Language independent

3.2 Distributed and Modular

3.3 A lot of libraries and tools

3.4 Bad Things About ROS

4. ROS Concepts

4.1 File system level

4.2 Computation graph level

4.3 Community level

5. References



1. What is ROS?

1.1 What ROS is

1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

3.1 Language independent

3.2 Distributed and Modular

3.3 A lot of libraries and tools

3.4 Bad Things About ROS

4. ROS Concepts

4.1 File system level

4.2 Computation graph level

4.3 Community level

5. References



What ROS is

Robot Operating System

- Short for: Robot Operating System.
- A collection of libraries and tools.
- It helps software developers create robot applications.



How Robotics
Research Keeps...

Re-Inventing the Wheel

First, someone publishes...



...and they write code that barely works but lets them publish...



...a paper with a proof-of-concept robot.



This prompts another lab to try to build on this result...



But inevitably, time runs out...



...and countless sleepless nights are spent writing code from scratch.



So, a grandiose plan is formed to write a new software API...



...and all the code used by previous lab members is a mess.

What ROS is

Robot Operating System

- A way to standardize writing software for robots.

- It enhances **code reusability** 

- ROS is open-source 

- It is a meta-operating system.

- ROS is installed on top of Linux. 



1. What is ROS?

1.1 What ROS is

1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



What ROS is NOT

Robot Operating System

- It is NOT a programming language.
- It is NOT an integrated development environment (IDE).
- It is NOT a stand-alone operating system



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



Analogy Between ROS and Operating Systems



Software Applications

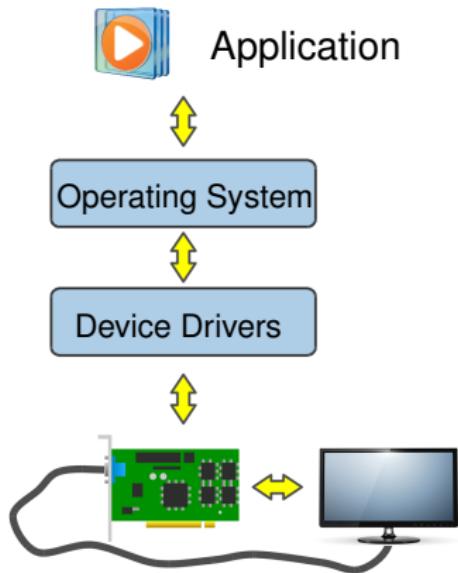
work on



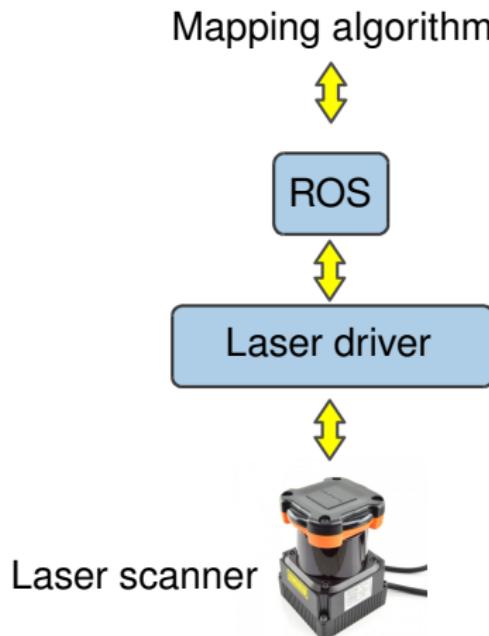
Different hardware



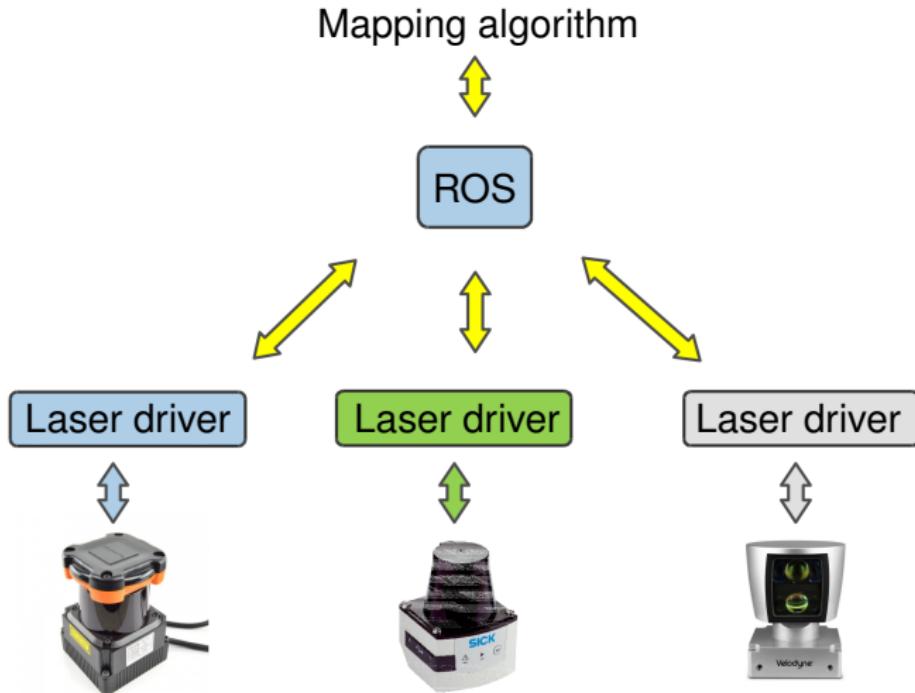
Analogy Between ROS and Operating Systems



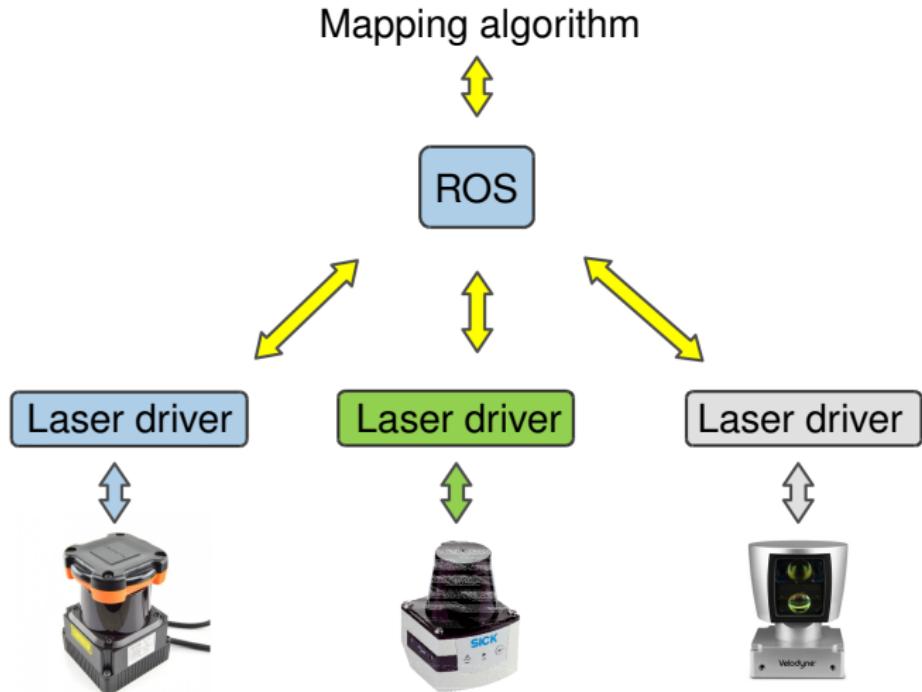
Analogy Between ROS and Operating Systems



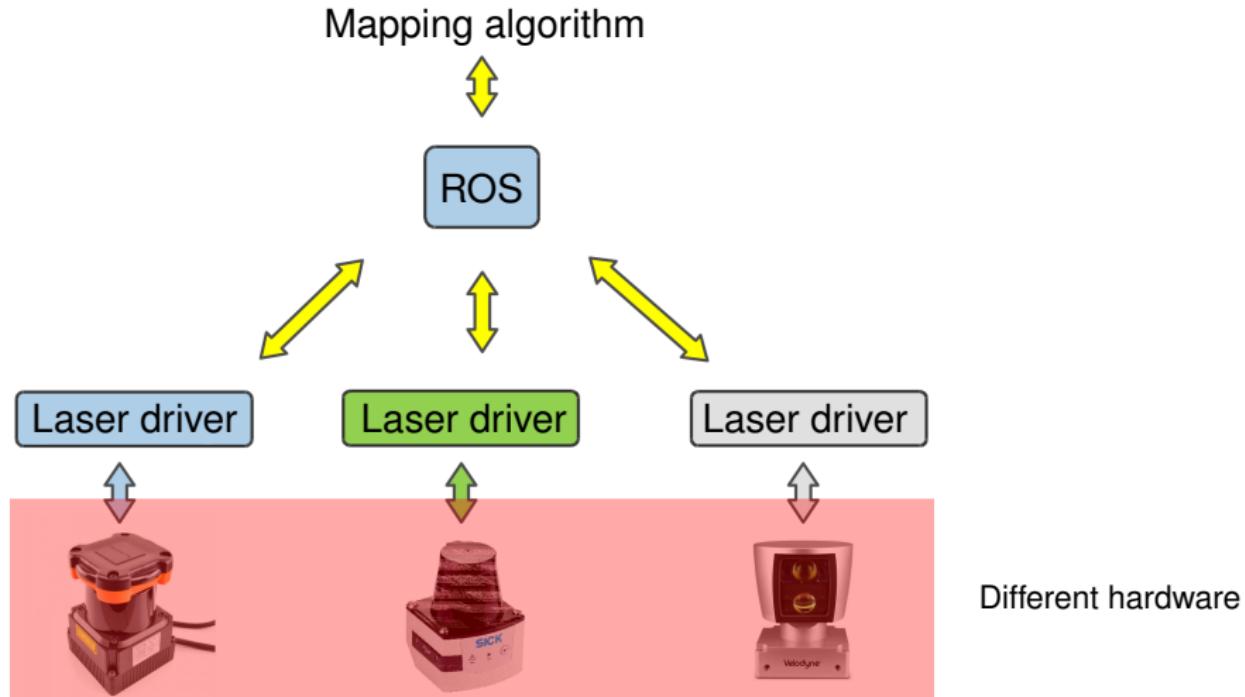
Analogy Between ROS and Operating Systems



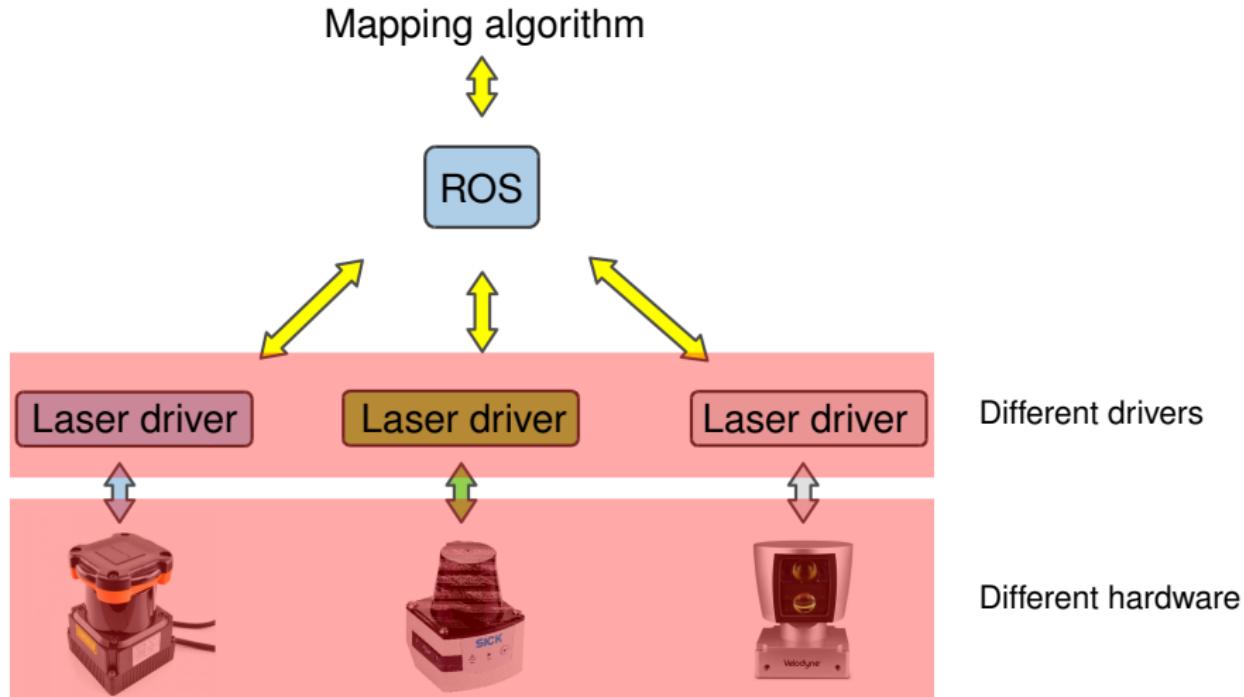
Analogy Between ROS and Operating Systems



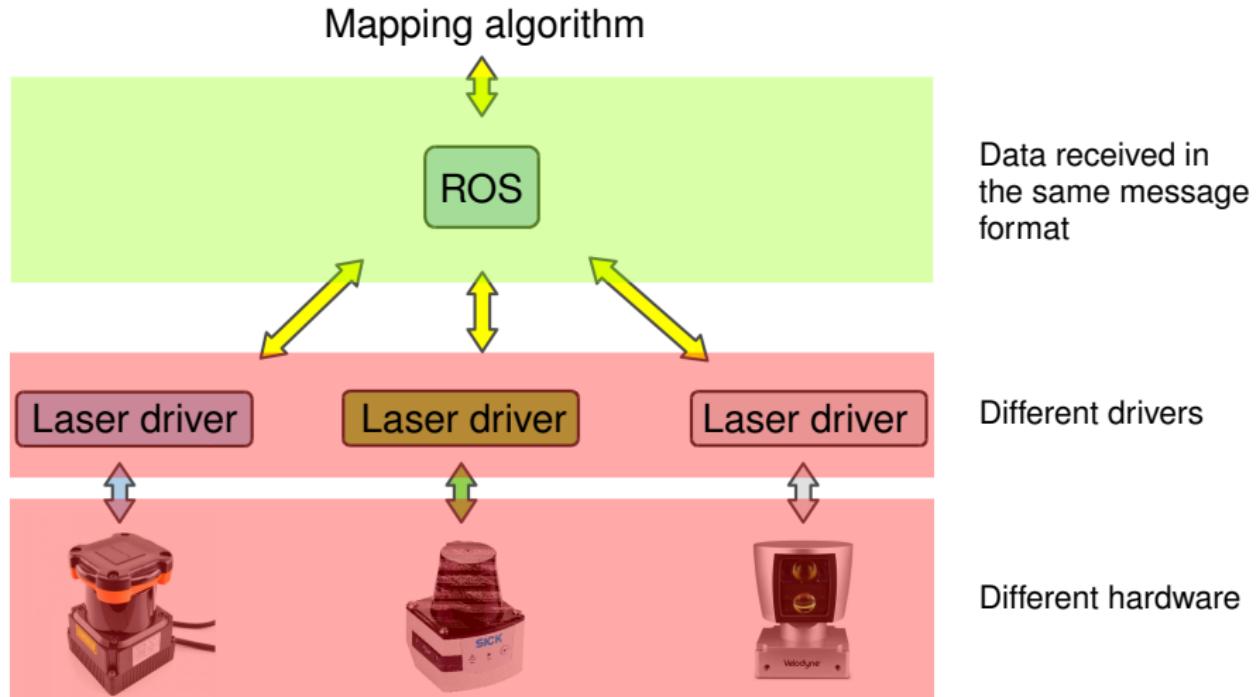
Analogy Between ROS and Operating Systems



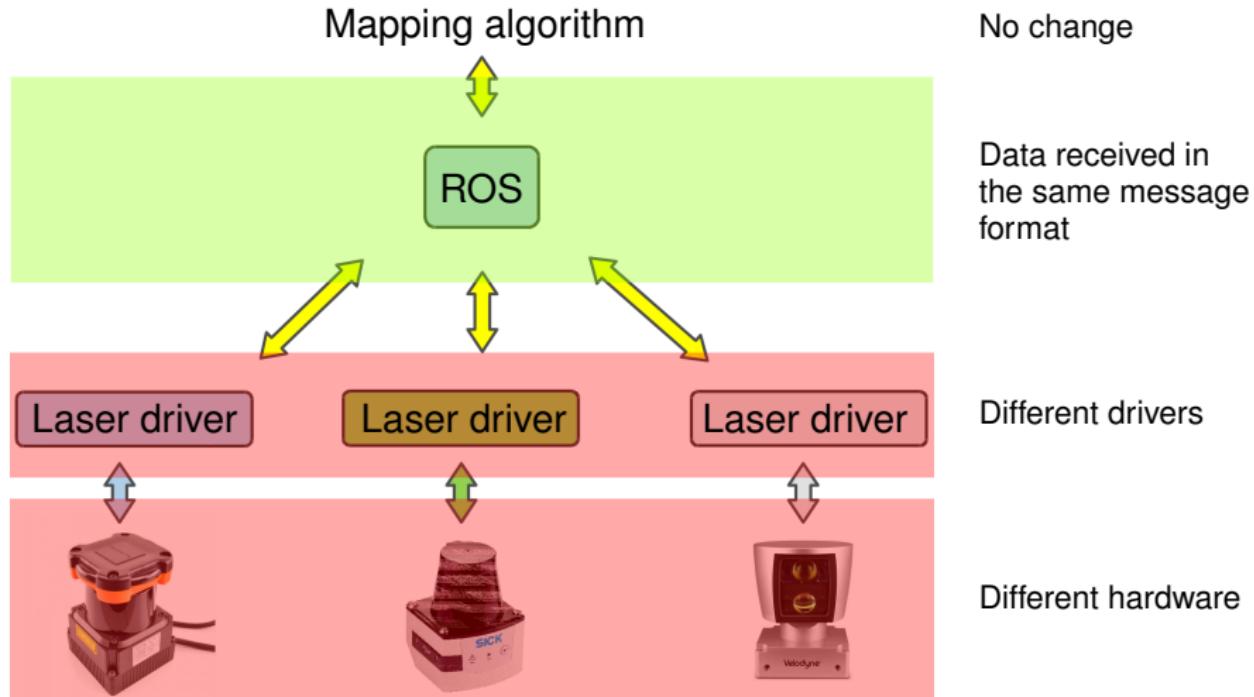
Analogy Between ROS and Operating Systems



Analogy Between ROS and Operating Systems



Analogy Between ROS and Operating Systems



Analogy Between ROS and Operating Systems

Mapping

Navigation

pick & place

Robot Applications

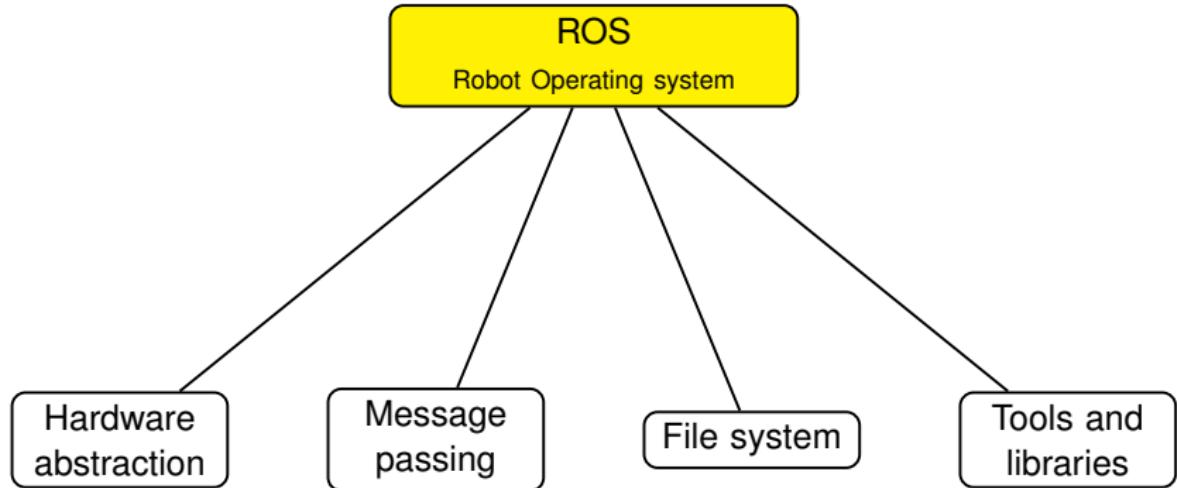
work on



Different hardware



Analogy Between ROS and Operating Systems



1. What is ROS?

1.1 What ROS is

1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

3.1 Language independent

3.2 Distributed and Modular

3.3 A lot of libraries and tools

3.4 Bad Things About ROS

4. ROS Concepts

4.1 File system level

4.2 Computation graph level

4.3 Community level

5. References



Features of ROS

- Language independent.
- Distributed and Modular.
- A lot of libraries and tools.
- Open Source.
- Active Community.



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



Features of ROS

Language independent

- ROS functionalities are implemented as a library in different programming languages.
- These libraries are referred to as ROS client libraries.



Language independent

Features of ROS

ROS client libraries.

- Main ROS Client libraries:
 - roscpp
 - rospy
 - roslib
- Experimental ROS client libraries:
 - rosjava
 - rosruby
 - and some others..
- ROS support on MATLAB:
 - Robotics System Toolbox



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



Distributed and Modular

Features of ROS

- ROS supports running processes on multiple computers connected together through a LAN.
- In a system running ROS, there will be multiple of processes where each process can do certain task. A process can be changed without altering the remaining processes.



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



A lot of libraries and tools

Features of ROS

- Examples of libraries:
 - Navigation stack.
 - SLAM (gmapping, hector SLAM, etc..).
 - Localization (amcl, etc..).
 - Motion planning for manipulators (MoveIt)
 - Support for popular libraries (OpenCV, PCL).
- Examples of tools:
 - RVIZ:3D Visualization.
 - ROS bag files: Logging Sensor Data.
 - Catkin: A Build System.
 - Command line tools.



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



Bad Things About ROS

- Learning ROS needs time.
- It needs a computer. Does not work on a microcontroller!
- Not optimized for multiple robots.
- Supported only on Linux, no support for Windows or macOS.
- ROS imposes a communication overhead.
- In the Python world, it forces us to “live in the past”. ROS python client library (`rospy`) uses python 2 instead of 3.



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



ROS Concepts

ROS concepts

- File system level
- Computation graph level
- Community level



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

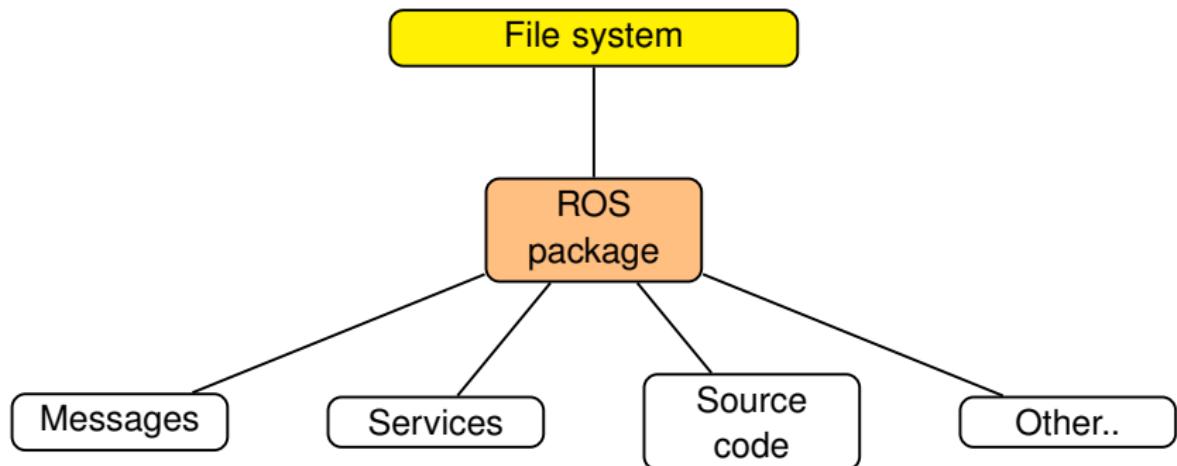
- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



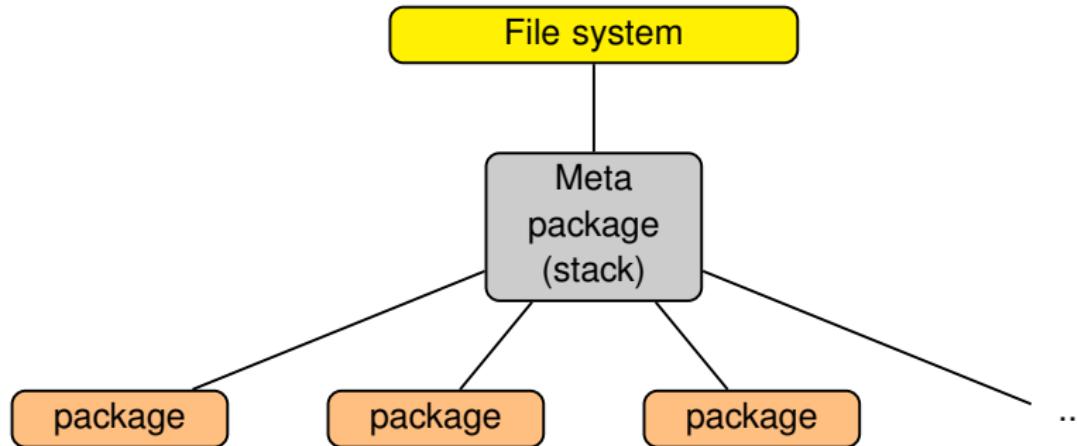
File system level

ROS Concepts



File system level

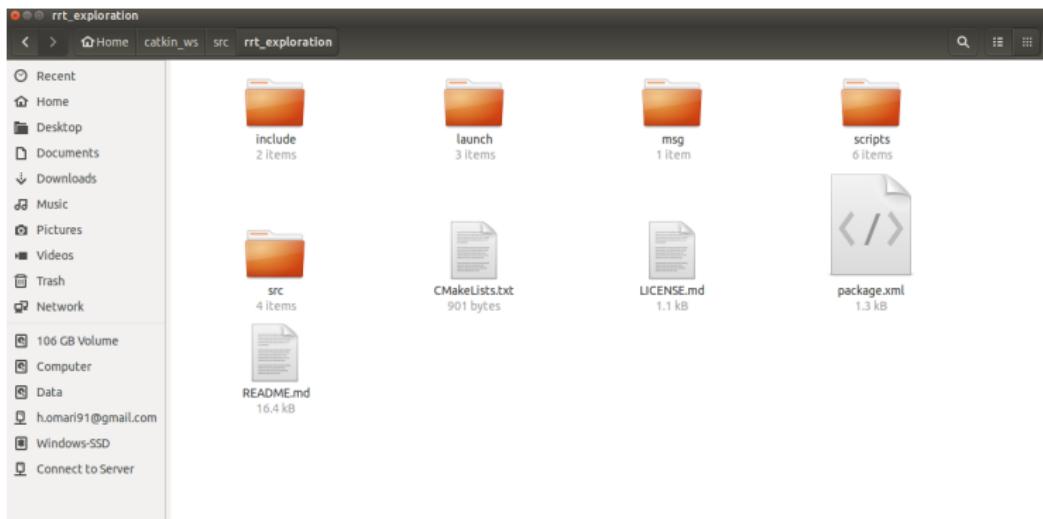
ROS Concepts



File system level

ROS Concepts

Inside a ROS package:



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



Computation graph level

ROS Concepts

- In an application that uses ROS, the computations are executed by a collection of processes called Nodes.
- Nodes are connected together in a peer-to-peer network.
- This network of nodes do all the computation and is referred to as ROS computation graph.
- ROS Nodes can be run on single or multiple computers.



Computation graph level

ROS Concepts

Concepts related to ROS computation graph:

1. Nodes.
2. Topics.
3. Messages.
4. Master.
5. Services.
6. Actions
7. Parameter Server.
8. Bags.



Computation graph level

ROS Concepts

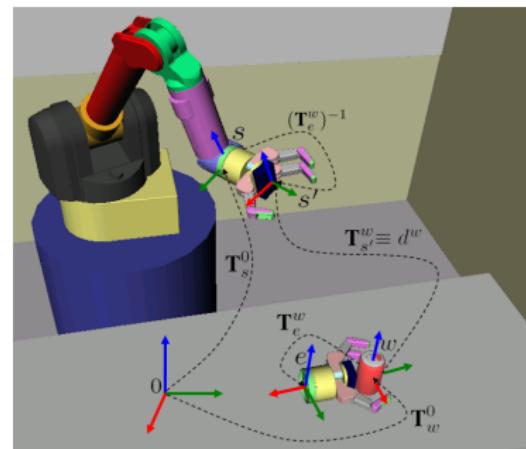
Nodes:

- A ROS node is a process that exchanges data with other processes through ROS network.
- It may be a python script, a C++ written process, or even a MATLAB script.
- Nodes perform computation.

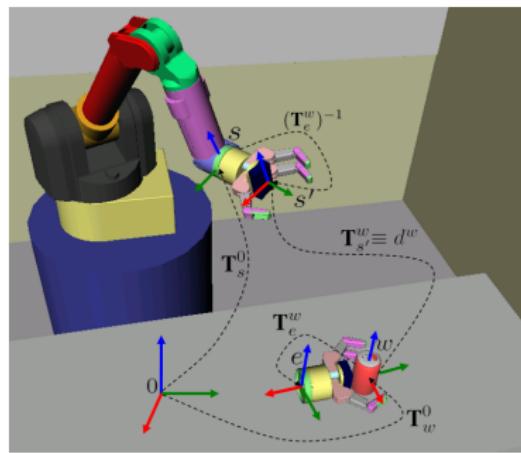


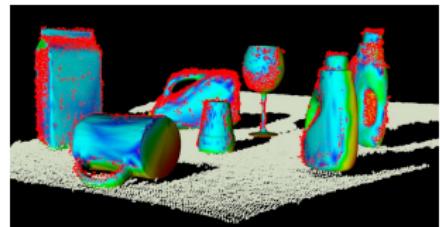




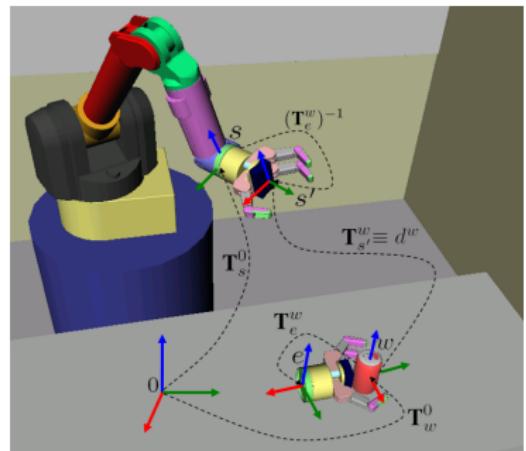


<http://arm.eecs.umich.edu/images/TSR.png>



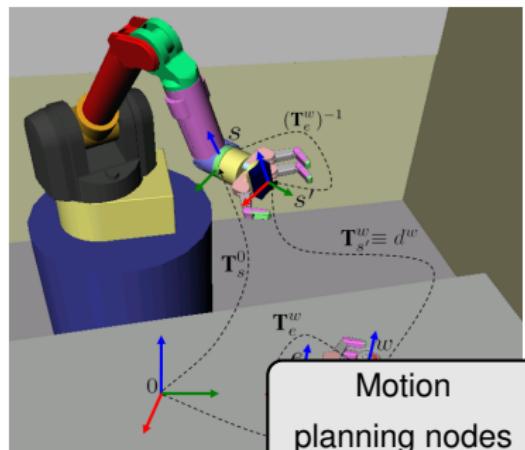
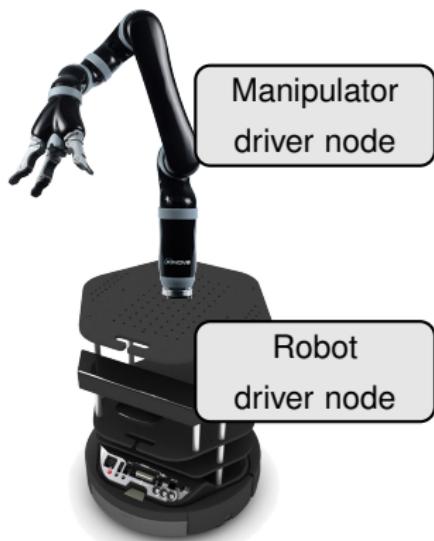
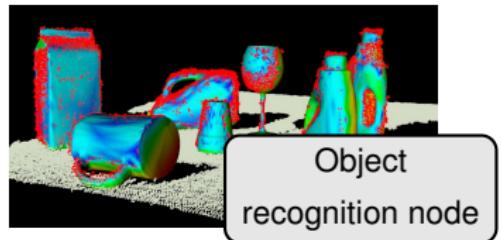
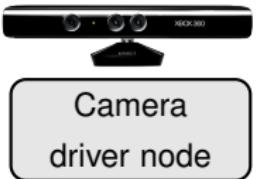


http://www.pointclouds.org/blog/_images/cvfh1.jpg





ROS master
node



Computation graph level

ROS Concepts

Topics and Messages:

- Nodes send data by publishing messages on a named topic.
- Nodes receive data by subscribing to a topic.
- Multiple nodes can publish/subscribe to the same topic.



Computation graph level

ROS Concepts

Topics and Messages:

- Publisher node publishes the messages on a topic at a chosen frequency.
- This **publish/subscribe** communication paradigm is a many-to-many one-way transport mechanism of data.
- The publishing node and subscribing node are not aware of each other's existence.



Computation graph level

ROS Concepts

Master:

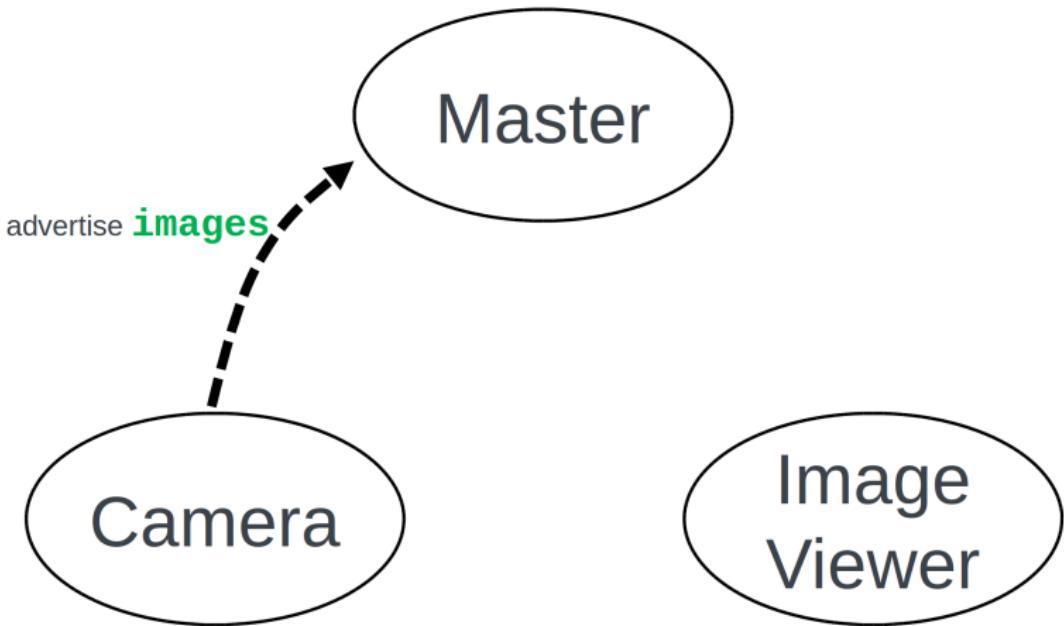
- The first process to run in an application that uses ROS, is the Master.
- The ROS Master provides name registration and lookup to the rest of the nodes.
- In a distributed system, we should run the master on one computer, and other remote nodes can find each other by communicating with this master.

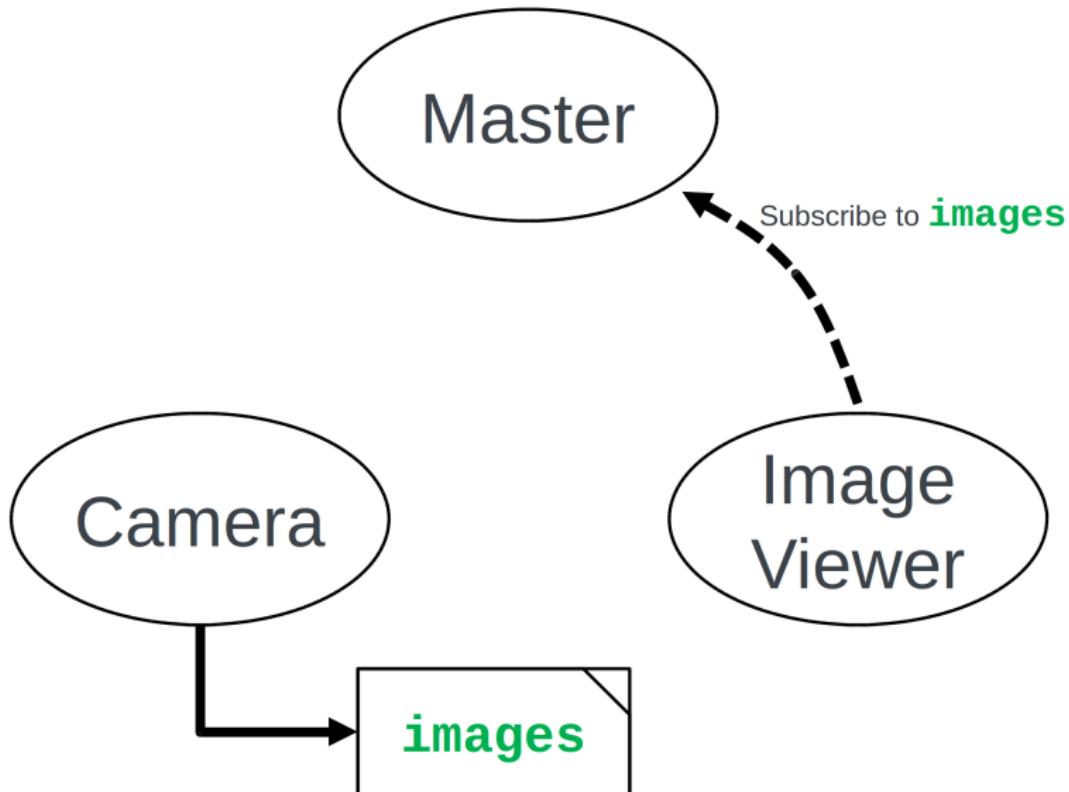


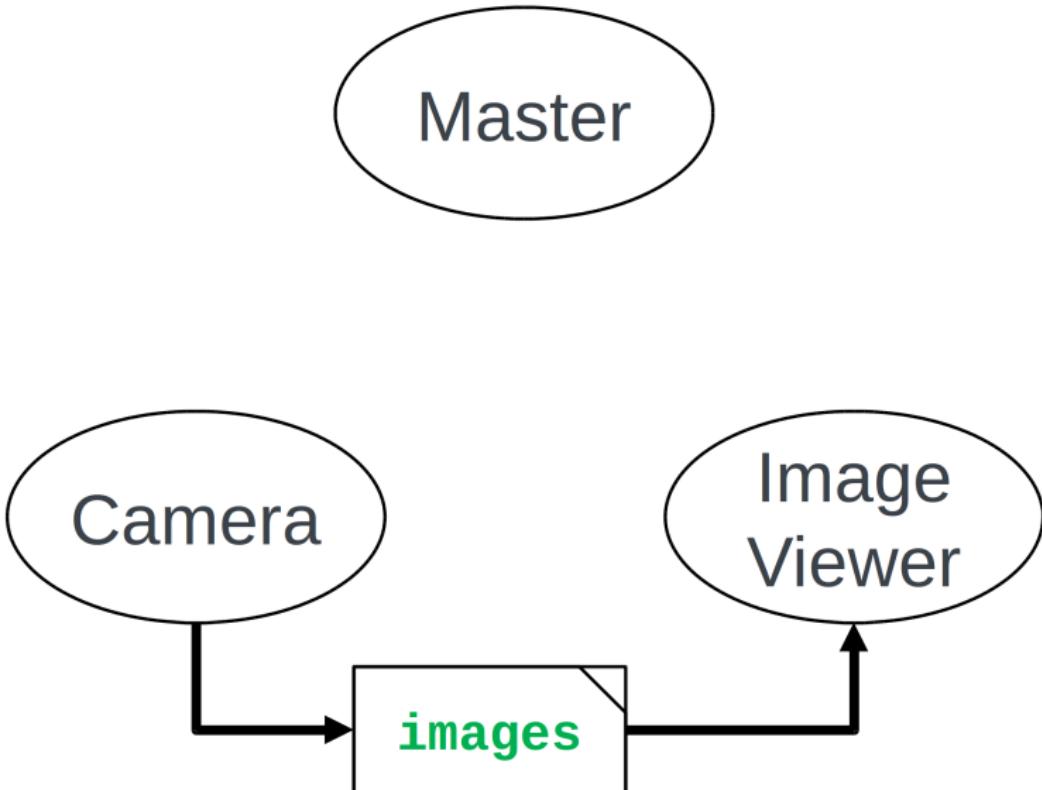
Master

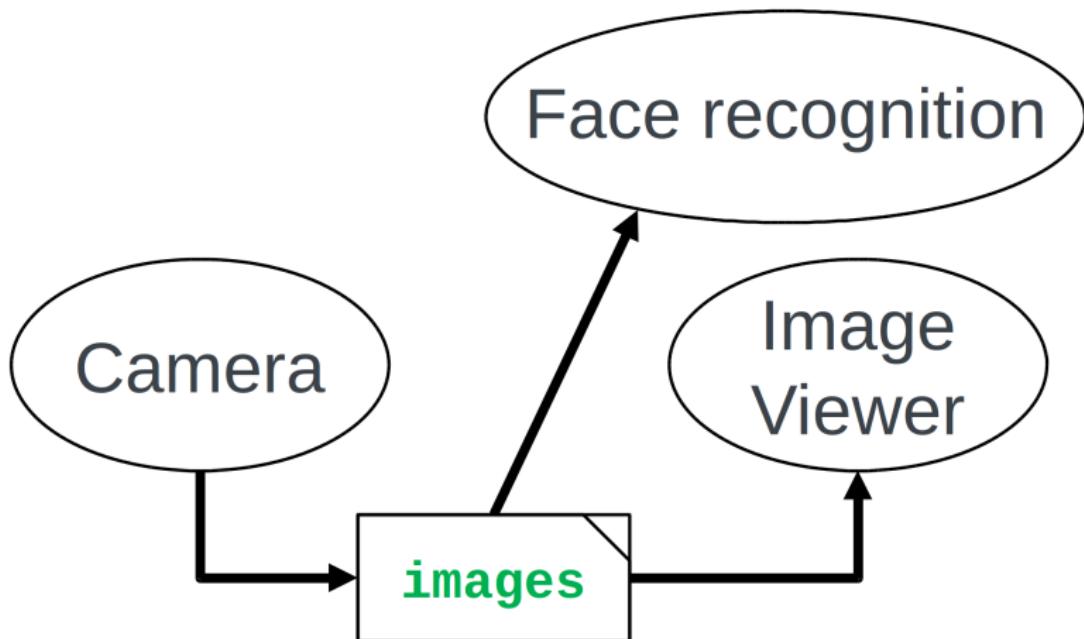
Camera

Image
Viewer







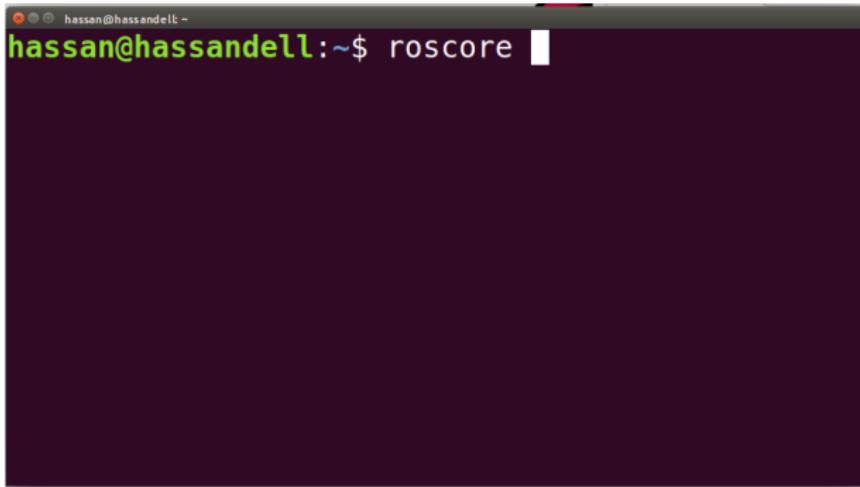


Computation graph level

ROS Concepts

Master:

- ROS master is invoked by this command:



A screenshot of a terminal window on a dark background. The window title bar shows three small icons. The terminal prompt is "hassan@hassandell:~\$". Below the prompt, the command "roscore" is typed and followed by a red cursor character. The rest of the terminal window is blank.

```
hassan@hassandell:~$ roscore
```



Example (TurtleSim)

Computation graph level

ROS Concepts

Concepts related to ROS computation graph:

1. Nodes.
2. Topics.
3. Messages.
4. Master.
5. Services.
6. Actions
7. Parameter Server.
8. Bags.



Computation graph level

ROS Concepts

Services:

- In many scenarios a publish/subscribe model is not enough, it's a one-way communication.
- Example scenario: plan a path service.
- ROS Services provide an additional way of communication between nodes, a **request / reply** interaction.



Computation graph level

ROS Concepts

Services:

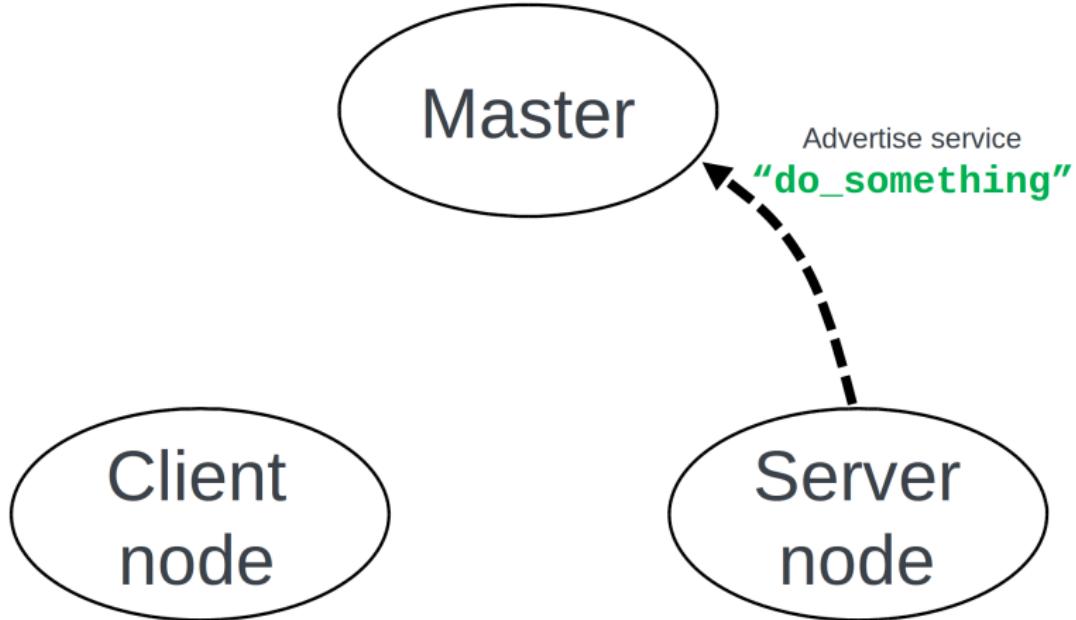
- It happens between two nodes, the service **server** node, and the service **client** node.
- A Client node sends a request for a named service and waits for the response, a node serving this service responds, and the communication is over.
- it is a one-to-one, two-way, one-time communication.



Master

Client
node

Server
node

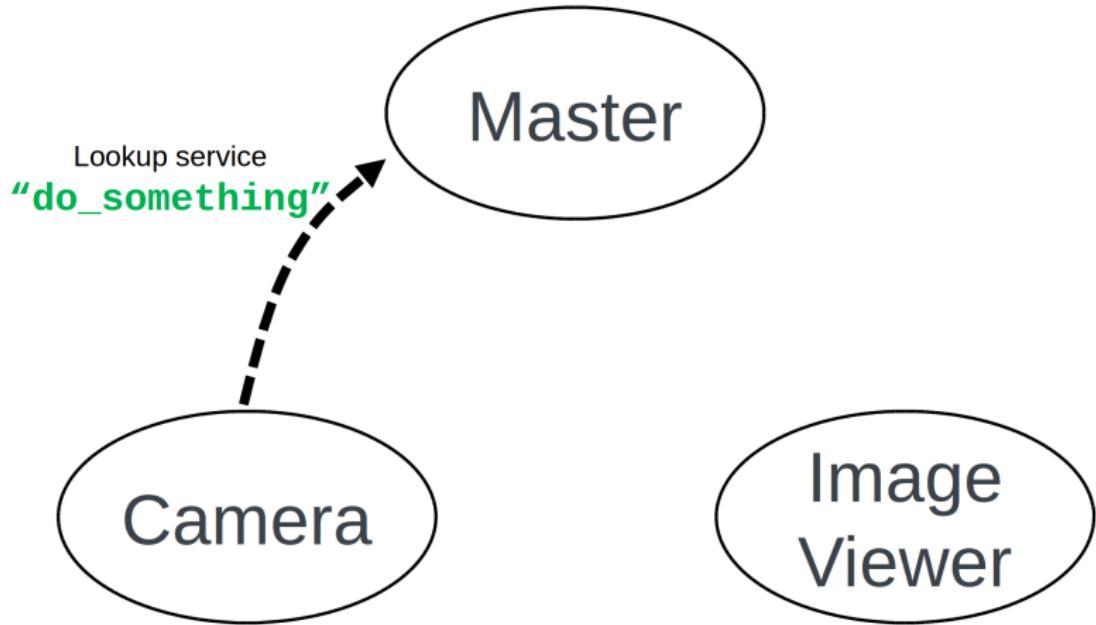


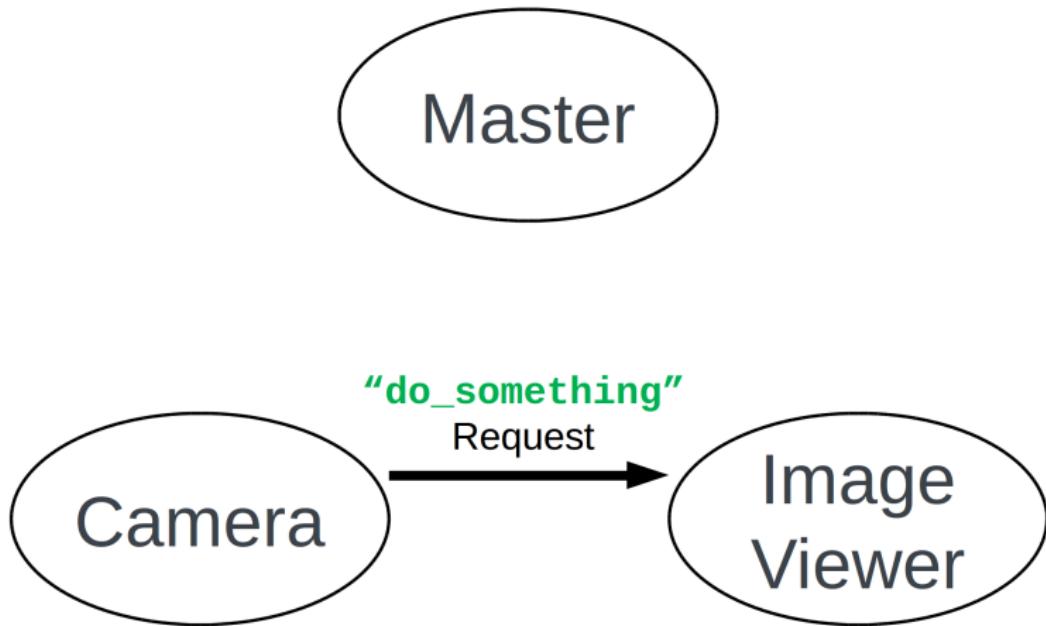
```
graph TD; Master((Master)); Client((Client node)); Server((Server node));
```

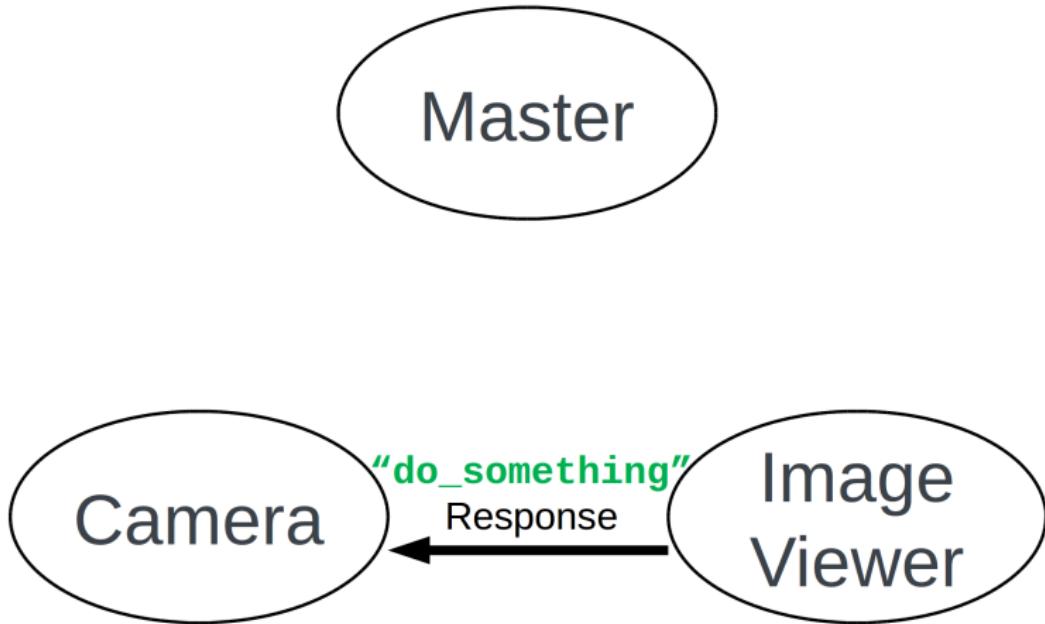
Master

Client
node

Server
node







```
graph TD; Master((Master)); Client((Client node)); Server((Server node));
```

Master

Client
node

Server
node

Example
(TurtleSim again)

Computation graph level

ROS Concepts

Actions:

- ROS services are not suitable for long-term tasks, a client that have sent a service request keeps on waiting for the response from the server. ROS actions solves this.
- In ROS actions, an action client sends a request to the server, the client doesn't have to wait for the response.



Computation graph level

ROS Concepts

Actions:

- Action client can request for feedback which the action server provides during execution.
- Once the server finishes executing the task, it send a result message to the client.



Computation graph level

ROS Concepts

Parameter Server:

- A network-shared dictionary accessible to all nodes.
- Typically used to store static data, like parameters and configurations.
- A central location to store static values.
- All nodes can access and modify those values.
- Parameter server is a part of ROS Master.



Computation graph level

ROS Concepts

Bags:

- ROS bag is a mechanism for recording data for later playback.
- You can record a complete session, with all the topics and messages being exchanged along with their time stamping.



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



Community level

ROS Concepts

Concepts related to ROS development process and it's community:

1. The ROS Wiki.
2. Repositories.
3. Mailing Lists.
4. ROS Answers.
5. ROS Distributions.



Community level

ROS Concepts

ROS Distributions:



ROS Melodic
5.2018 - 5.2023
(LTS)
(Ubuntu 18 EOL)



ROS Lunar
5.2017 - 4.2019



ROS Kinetic
5.2016 - 4.2021
(LTS)
(Ubuntu 16 EOL)
(what we use)



ROS Indigo
7.2014 - 4.2019
(LTS)
(Ubuntu 14 EOL)

ROS posters are from: <http://wiki.ros.org/Distributions>



Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences



1. What is ROS?

- 1.1 What ROS is
- 1.2 What ROS is NOT

2. Analogy Between ROS and Operating Systems

3. Features of ROS

- 3.1 Language independent
- 3.2 Distributed and Modular
- 3.3 A lot of libraries and tools
- 3.4 Bad Things About ROS

4. ROS Concepts

- 4.1 File system level
- 4.2 Computation graph level
- 4.3 Community level

5. References



References

1. ROS Wiki.
2. ROS2 Wiki, on the drawbacks of ROS:
http://design.ros2.org/articles/why_ros2.html.
3. Overview on ROS services: <https://www.youtube.com/watch?v=qhnImrGQVvM>.
4. Overview on ROS actions: <https://www.youtube.com/watch?v=LoRXdNMusIQ>.
5. ROS introduction slides by Rada:
https://wiki.ros.org/Events/CoTeSys-ROS-School?action=AttachFile&do=get&target=ros_tutorial.pdf.



Thank you

Any questions?