

ROS:

Taller introductorio





Open Source Robotics Foundation








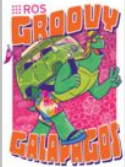

Willow
Garage













¿Qué es ROS?

- ROS = Robot Operating System
- Software libre
- Meta sistema operativo
- Funciones comunes
- Herramientas y bibliotecas
- Reutilización de software
- Manejo de paquetes
- Comunicación de mensajes
- Procesamiento distribuido
- Abstracción de hardware
- Meta build system

Distribuciones de ROS

| | | | | |
|---|---------------------|---|---|-----------------------------|
| ROS Kinetic Kame | May, 2016 |  |  | May, 2021 |
| ROS Jade Turtle (Recommended for Latest) | May 23rd, 2015 |  |  | May, 2017 |
| ROS Indigo Igloo (Recommended for Stability) | July 22nd, 2014 |  |  | April, 2019 (Trusty EOL) |
| ROS Hydro Medusa | September 4th, 2013 |  |  | May, 2015 |
| ROS Groovy Galapagos | December 31, 2012 |  |  | July, 2014 |

| | | | | |
|-------------------|-----------------|---|---|----|
| ROS Fuerte Turtle | April 23, 2012 |  |  | -- |
| ROS Electric Emys | August 30, 2011 |  |  | -- |
| ROS Diamondback | March 2, 2011 |  |  | -- |
| ROS C Turtle | August 2, 2010 |  |  | -- |
| ROS Box Turtle | March 2, 2010 |  |  | -- |

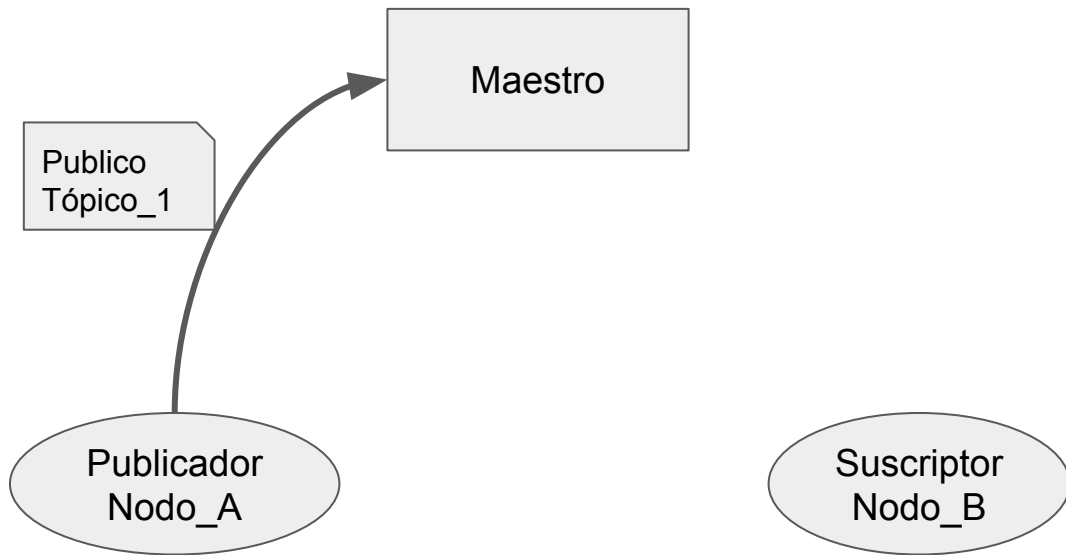
Box Turtle

Arquitectura de un sistema ROS



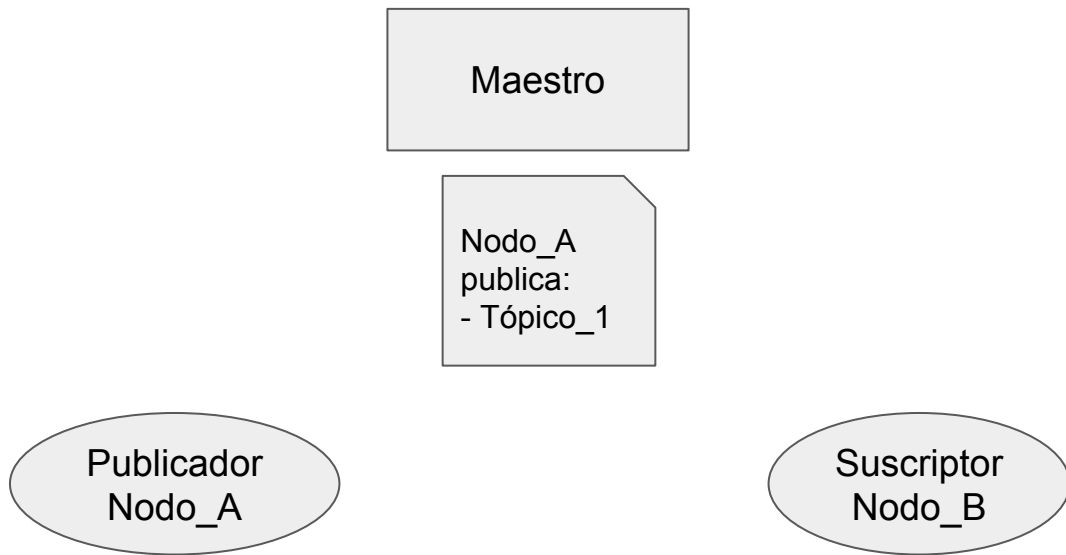
Arquitectura de un sistema ROS

Nodos, tópicos y mensajes



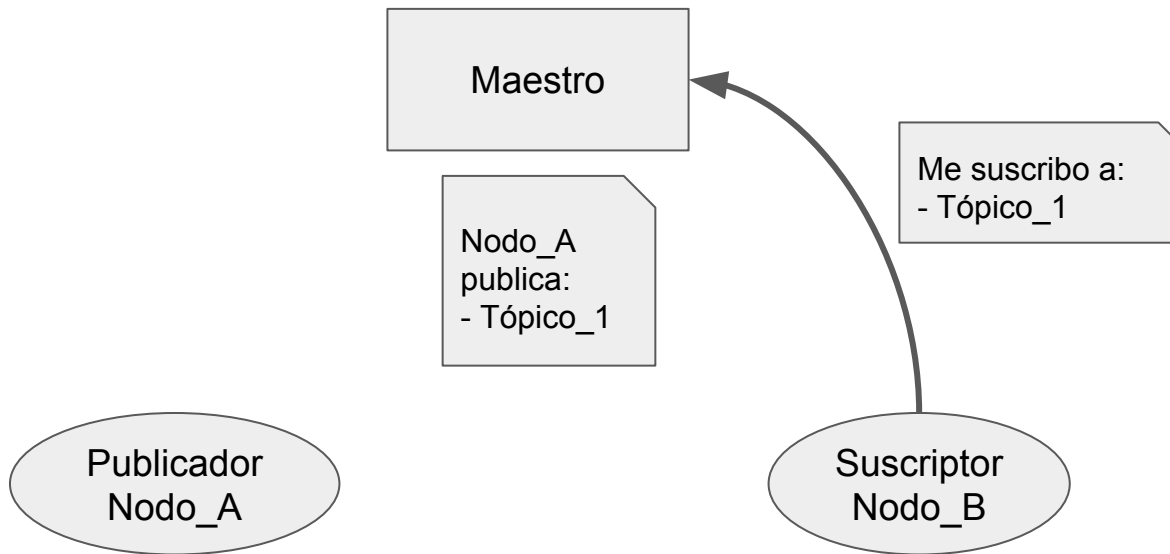
Arquitectura de un sistema ROS

Nodos, tópicos y mensajes



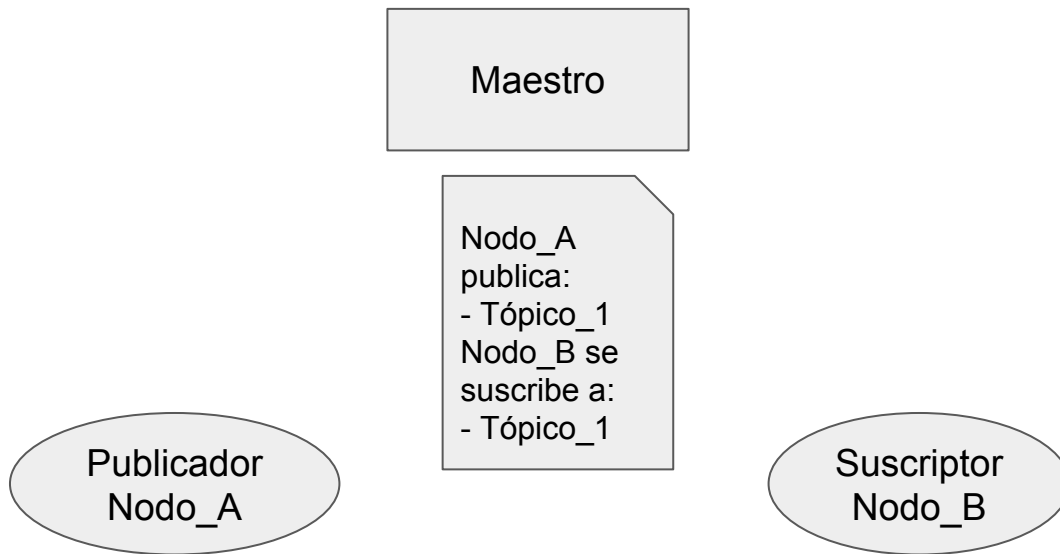
Arquitectura de un sistema ROS

Nodos, tópicos y mensajes



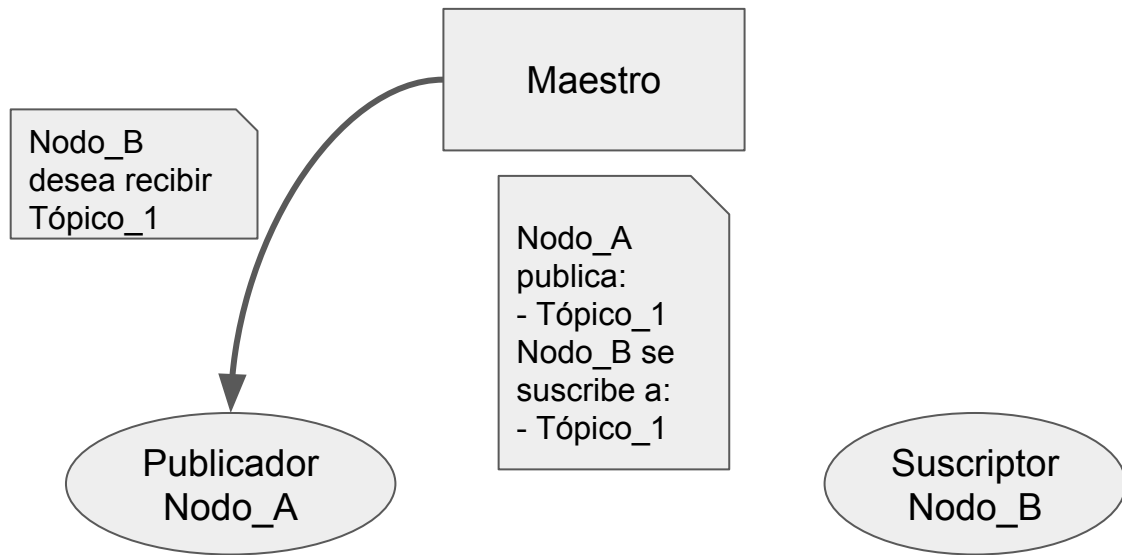
Arquitectura de un sistema ROS

Nodos, tópicos y mensajes



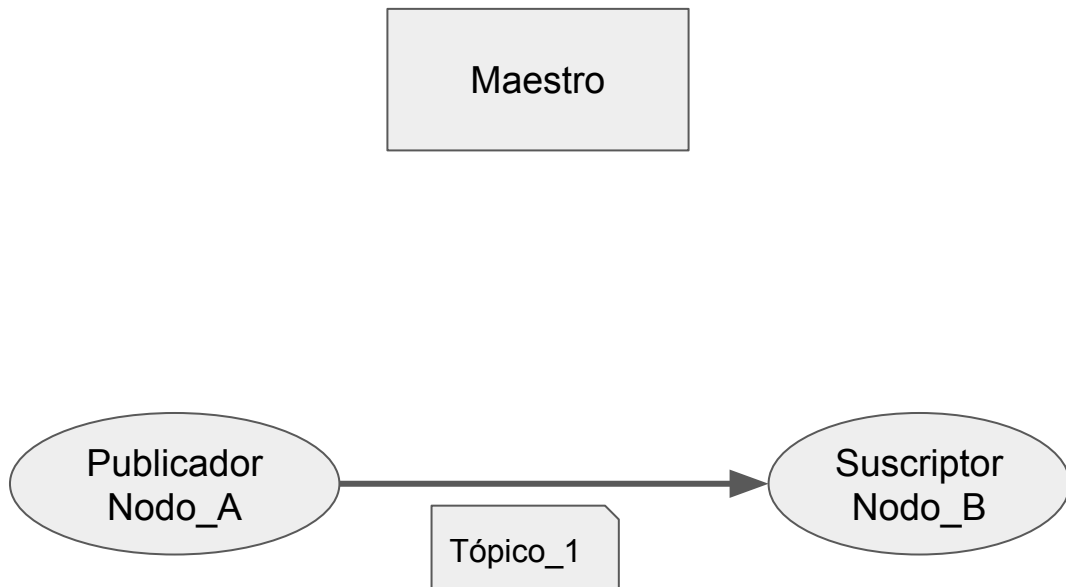
Arquitectura de un sistema ROS

Nodos, tópicos y mensajes



Arquitectura de un sistema ROS

Nodos, tópicos y mensajes



Arquitectura de un sistema ROS

Ejercicio 1: antes de empezar...

```
$ gedit ~/.bashrc
```

```
# Obtener el entorno de ROS  
source /opt/ros/indigo/setup.bash
```

Arquitectura de un sistema ROS

Ejercicio 1: lanzar el maestro y un nodo

```
$ roscore
```

```
$ rosrun turtlesim turtlesim_node
```

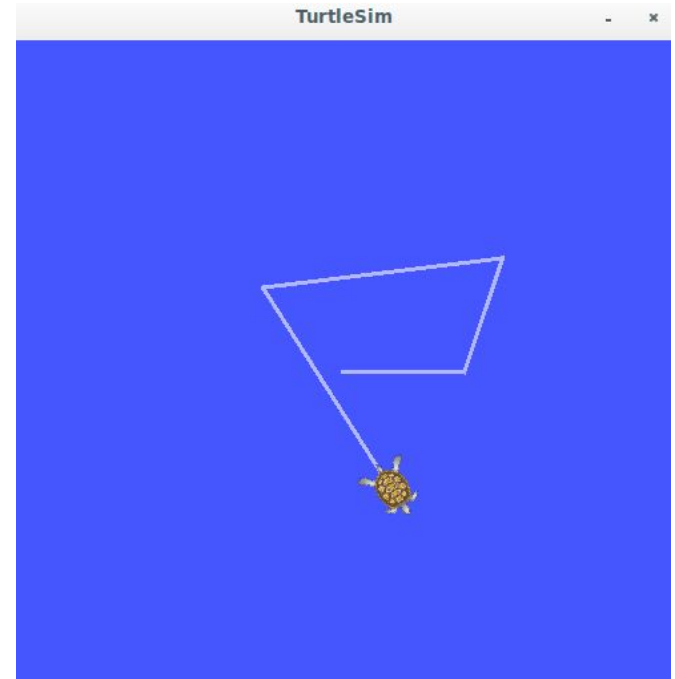
TurtleSim



Arquitectura de un sistema ROS

Ejercicio 1: lanzar otro nodo para controlar la tortuga

```
$ rosrun turtlesim turtle_teleop_key
```



Arquitectura de un sistema ROS

Ejercicio 1: enumerando nodos, tópicos y espiando mensajes

```
$ rosnode list  
  
$ rostopic list  
  
$ rostopic echo /turtle1/cmd_vel  
  
$ rqt_graph
```


Estructura de un mensaje

```
$ rostopic info /turtle1/cmd_vel  
  
$ rosmmsg show geometry_msgs/Twist  
  
$ rosmmsg show geometry_msgs/Vector3
```

Estructura de un mensaje

Twist.msg

This expresses velocity in free space broken into its linear and angular parts.

Vector3 linear
Vector3 angular

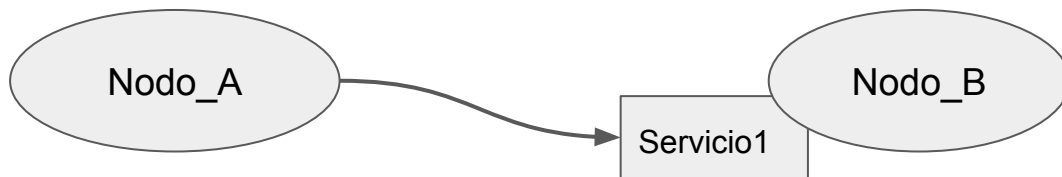
Vector3.msg

This represents a vector in free space.

float64 x
float64 y
float64 z

Servicios

Llamadas a funciones remotas



Servicios

Llamadas a funciones remotas

```
$ rosservice list
```

```
$ rosservice call /reset
```

Experimentando ROS

Ejercicio 2: crear un nodo ROS

```
import rospy, math, random, time
from geometry_msgs.msg import Twist

def talker():
    pub = rospy.Publisher('/turtle1/cmd_vel', Twist, queue_size=10)
    rospy.init_node('square_turtle', anonymous=True)
    while not rospy.is_shutdown():
        rospy.loginfo("Moviendo la tortuga...")
        mensaje = Twist()
        mensaje.linear.x = random.random()*2 + 0.1
        pub.publish(mensaje)
        rospy.sleep(2)
        mensaje = Twist()
        mensaje.angular.z = random.random()*3
        pub.publish(mensaje)
        rospy.sleep(2)

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

Experimentando ROS

Ejercicio 3: conectándose a un maestro externo

```
$ export ROS_MASTER_URI=http://192.168.0.2:11311
```

Experimentando ROS

Ejercicio 3: conectándose a un maestro externo

```
import rospy, cv2
import numpy as np

from sensor_msgs.msg import CompressedImage

class image_converter:
    def __init__(self):
        self.image_sub = rospy.Subscriber("/camera",
        CompressedImage, self.callback)

    def callback(self, data):
        ##### direct conversion to CV2 #####
        np_arr = np.fromstring(data.data, np.uint8)
        image_np = cv2.imdecode(np_arr, cv2.
        CV_LOAD_IMAGE_COLOR)
        cv2.imshow("ImageWindow", image_np)
        cv2.waitKey(1)
```

```
def main():
    cv2.namedWindow("ImageWindow")
    ic = image_converter()
    rospy.init_node('image_receiver', anonymous=True)
    try:
        rospy.spin()
    except KeyboardInterrupt:
        print("Shutting down")
    cv2.destroyAllWindows()

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

Referencias

- “A gentle introduction to ROS”, Jason M. O’Kane, <https://cse.sc.edu/~jokane/agitr/>
- “Programming Robots with ROS”, Morgan Quigley, Brian Gerkey, William D. Smart, O’Reilly
- <http://wiki.ros.org/>
- <http://answers.ros.org/>