HackerSystem's

GIT, ROS y Otros

28 de Marzo de 2018

Hora de las noticias

Amaru Ahumada (Ing Trainee)
&
Gerald Barnert (Ing Trainee)

Luces interactivas en vehículos autónomos

—

27 de Marzo de 2019

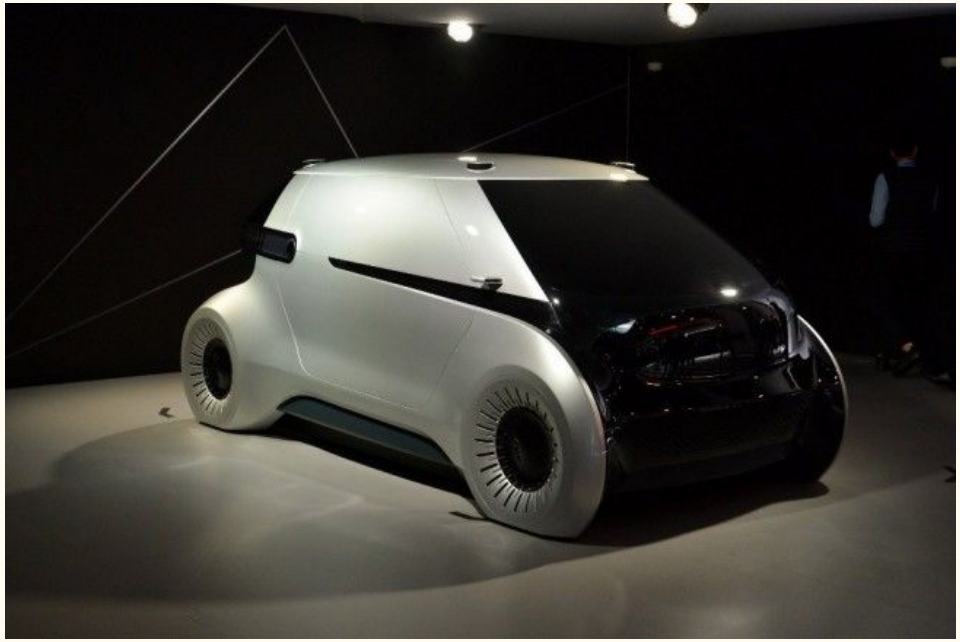
CES 2019 - Hyundai

Hyundai MOBIS presenta un sistema de luces interactivas en el CES 2019

Hyundai MOBIS, la división de componentes de Hyundai, ha presentado en el Consumer Electronics Show 2019 un prototipo que cuenta con un sistema de iluminación interactiva, destinado a que vehículo y usuario se comuniquen con los peatones y otros usuarios de la vía.

- Consumer Electronics Show celebrado en Enero de 2019 en Las Vegas
- Hyundai MOBIS, la división de componente de Hyundai
- Últimos avances de la marca en sistemas de iluminación.

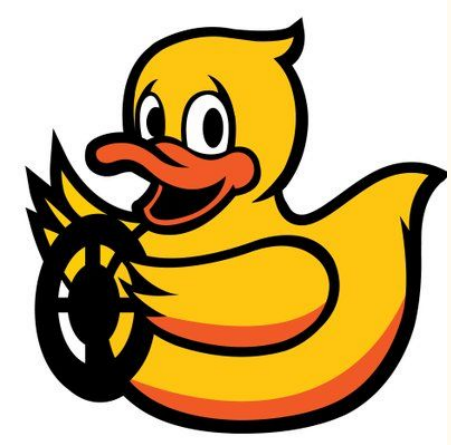
- Sistema de iluminación exterior sobre los paneles de su carrocería
- Sistema de comunicación interactivo
- Proyección de gráficas y mensajes sobre su carrocería



Hyundai MOBIS Communication Lighting concept.


2019 CES
Hyundai Mobis
Future Autonomous Driving
M.VISION

¿En qué afecta esto a Duckietown?



- Disminuye cantidad de accidentes
- Mayor organización
- Mayor bienestar

Reflexión y Discusión

- Reducción de probabilidad de accidentes
- Relación entre peatones y automóviles
- Inicio de nuevas tecnologías innovadoras en el país

Anuncios corporativos

M. Mattamala (CEO)

HackerSystem's

GIT, ROS y Otros

29 de Marzo de 2018

José Astorga

Chief Operations Officer (COO)

(ROS Evangelizer)

Sebastián Zapata

Cyber Security Ninja

Duckietown
Engineering
Chile

Capacitación de hoy

- SSH - FTP
- Byobu
- Git
- ROS
  - Nodos
  - Tópicos
  - Publicación/Subscripción
  - ROS master
  - Packages
- Desafío

SSH

- Protocolo de seguridad
- Permite conectar 2 computadores por red
- Permite iniciar procesos a través de otro computador

FTP

- Protocolo de intercambio de archivos
- Permite traspasar archivos de un computador a otro
- Con ayuda de software, permite editar los archivos de un computador usando otro.

Byobu

- Terminal++
- Emula múltiples terminales en una sola
- Ideal para trabajar con duckiebots

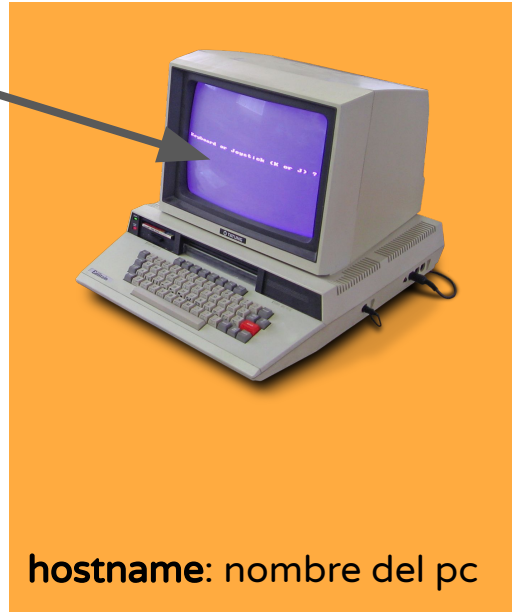
GIT

- Versionamiento
- Tracking de modificaciones
- Colaboración con pares
- Branchs, forks, gitstuffs.

Conexión al Robot...

Conexión SSH al duckiebot

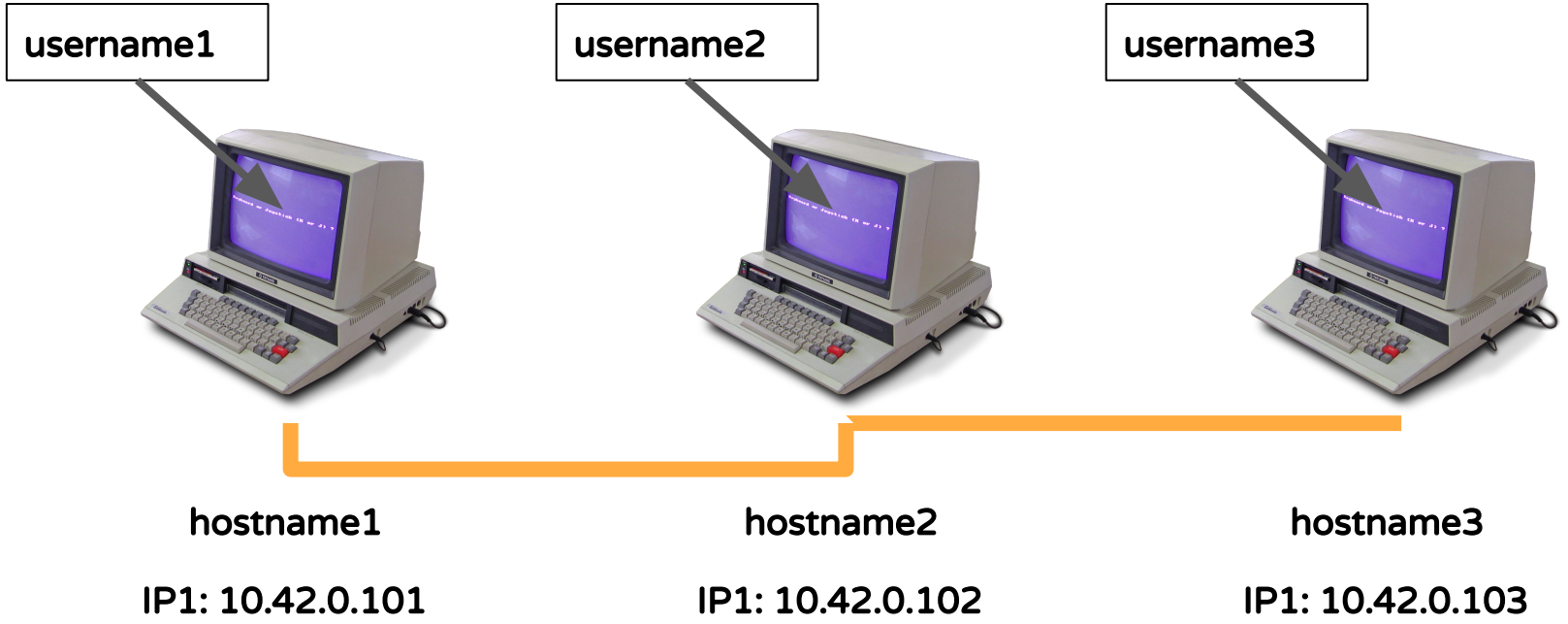
username: nombre del usuari@ ***en*** el pc



hostname: nombre del pc

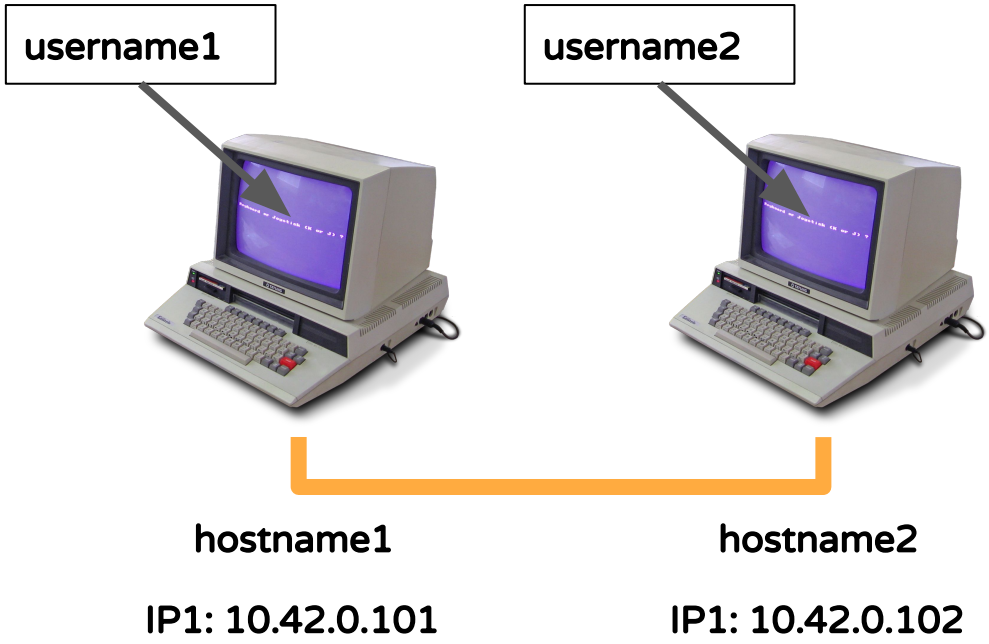
Duckietown Engineering Chile

Red de computadores



Duckietown
Engineering
Chile

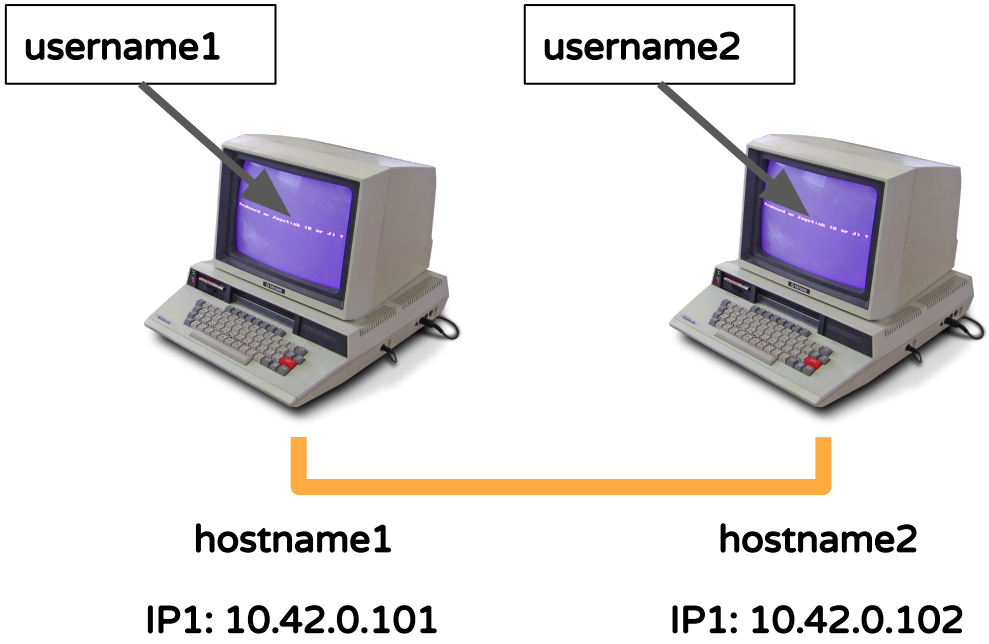
Asegurar que están conectados




```
$ ping 10.42.0.102
```

Duckietown Engineering Chile

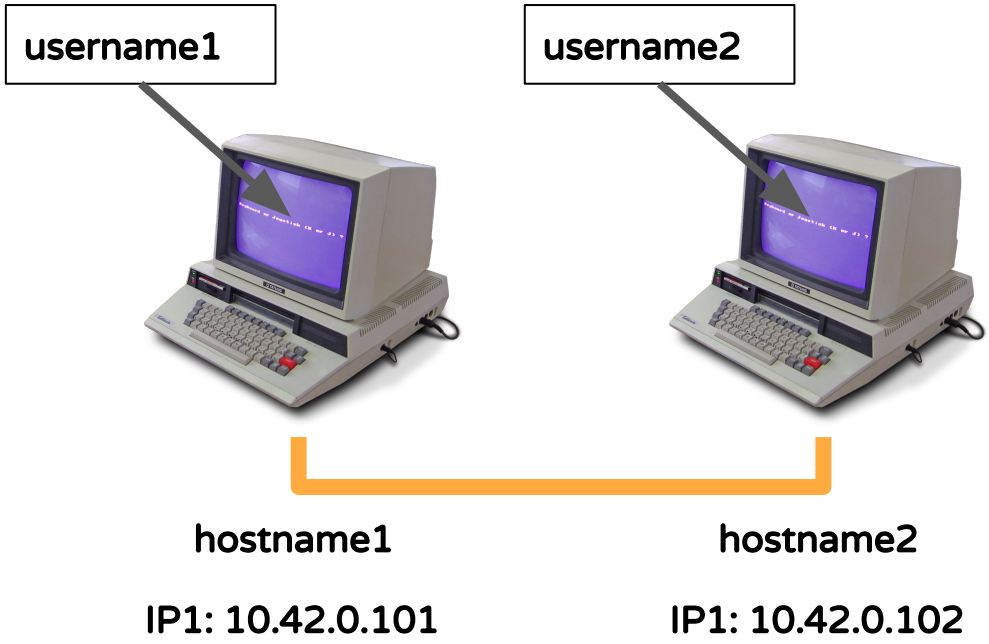
Asegurar que están conectados




```
$ ping hostname2
```

Duckietown Engineering Chile

Conexión SSH PC 1 a PC 2



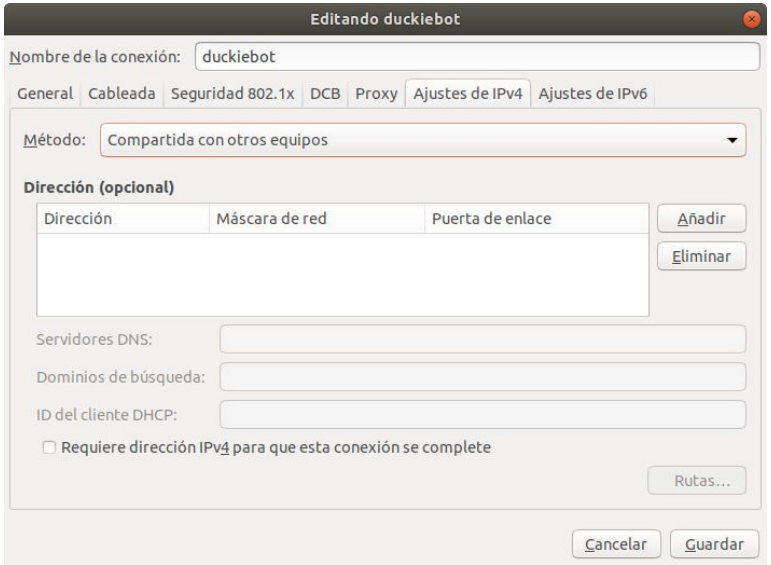

```
$ ssh username2@10.42.0.102

$ ssh username2@hostname2
```

Duckietown Engineering Chile

Primera conexión a Duckiebot

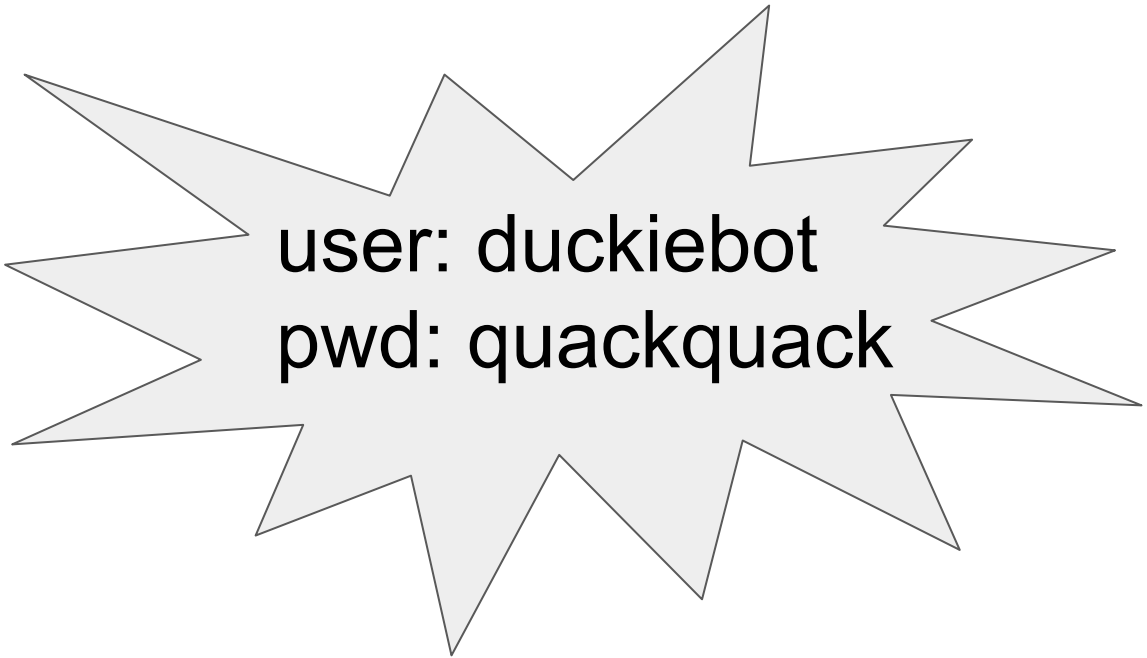
- Configurar conexión Ethernet
  - Ubuntu 18: $ nm-connection-editor



Duckietown Engineering Chile

Primera conexión a Duckiebot

```
$ ssh -X duckiebot@duckiebot.local
```



Ahora probamos Byobu

$ byobu

Ahora probamos Byobu

- Byobu Shortcuts:
  - <F2> : Crear nueva pestaña
  - <F3> : Mover a pestaña anterior
  - <F4> : Mover a pestaña siguiente
  - <Ctrl + D > o “exit” : Cerrar pestaña
  - <Ctrl + F2> : Dividir pestaña verticalmente
  - <Shift + F2> : Dividir pestaña horizontalmente
  - <Shift + F3> <Shift + F4> : Cambiar de división

Ahora probamos Byobu

- Comandos de Unix
  - $ ls
  - $ cd
  - $ pwd
  - $ cat
  - $ nano | vi

Quiero ver carpetas como en güindous

- Conexión sftp
  - Ir a nautilus
  - Panel izquierdo -> Otras ubicaciones
  - Conectar al servidor:
    - **sftp://duckiebot@duckiebot.local**

pwd: quackquack

Duckietown Engineering Chile

Y ahora ...

... ¿Cómo se programan los robots?

self driving cars aren't even hard to make lol

just program it not to hit stuff

```
if(goingToHitStuff) {
dont();
}
```

ROS

Robot Operating System

Team MAXed Out

Learning Algorithms and Systems Laboratory (LASA) | EPFL

Duckietown Engineering Chile

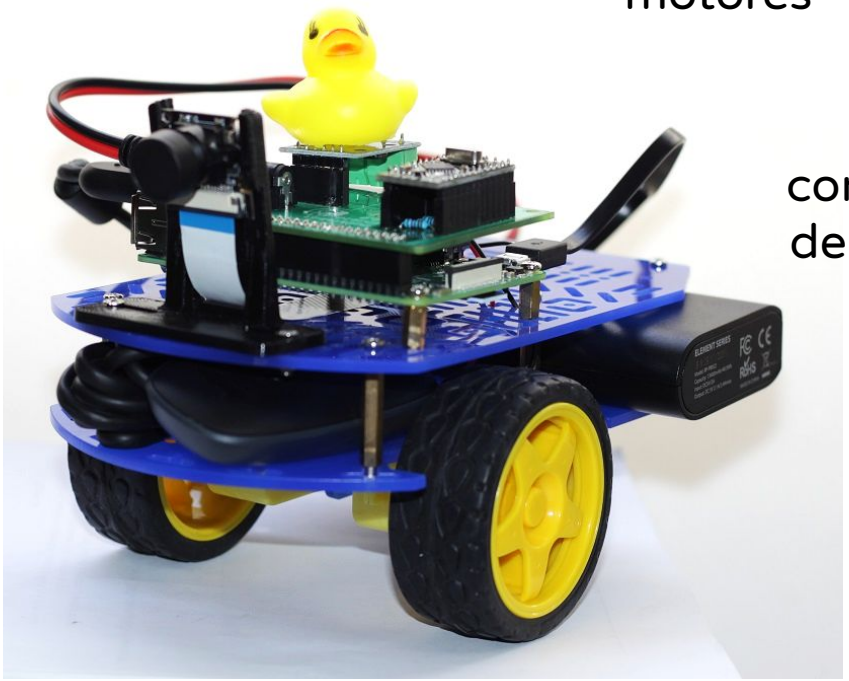
El problema

Motor controllers
Inverse Kinematics
GUI
Collision Detection
Trajectory Planner
PID Controller
Error Monitoring
Teaching Mode

El problema

motores

cámara



controlador
de motores

joystick

visión
computacional

planificación

cinemática

control automático

Duckietown
Engineering
Chile

ROS is an open-source, meta-operating system

pointcloudlibrary

Arquitectura: Nodos

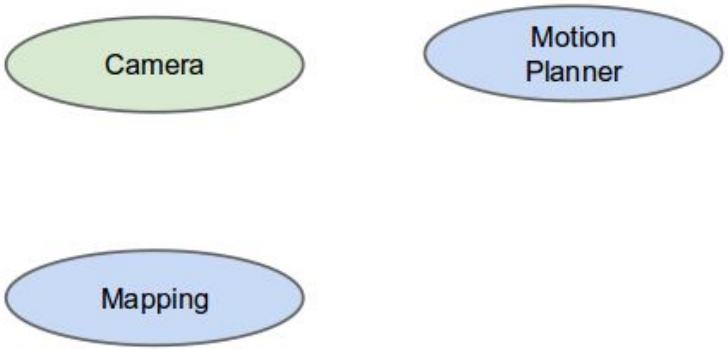
-Equivalente a un módulo e independiente en ejecución

-En general, se programan en C++ o Python

-Puede:

-interactuar con el **hardware** (driver)

-mandar y recibir **mensajes**

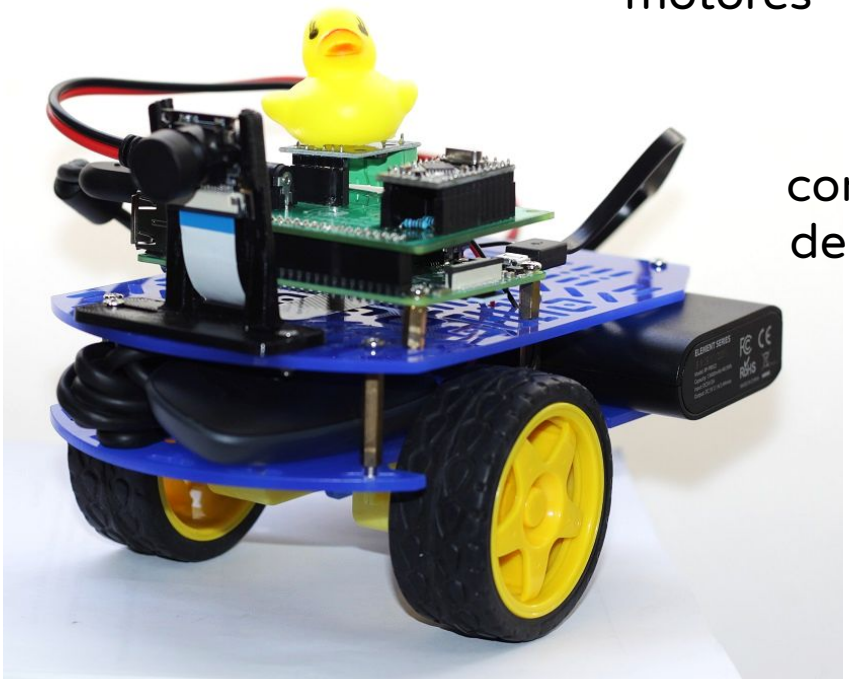


Duckietown Engineering Chile

El problema

motores

cámara



controlador
de motores

joystick

visión
computacional

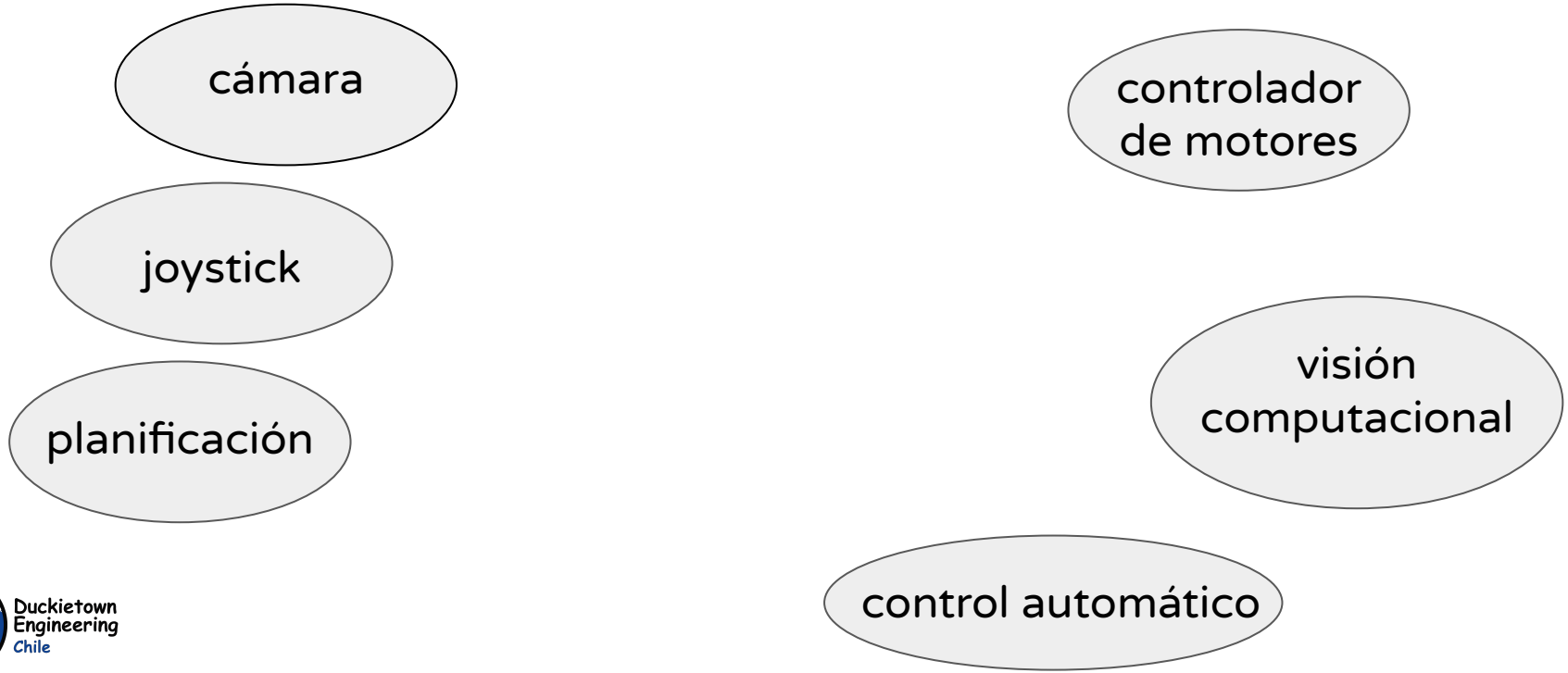
planificación

cinemática

control automático

Duckietown
Engineering
Chile

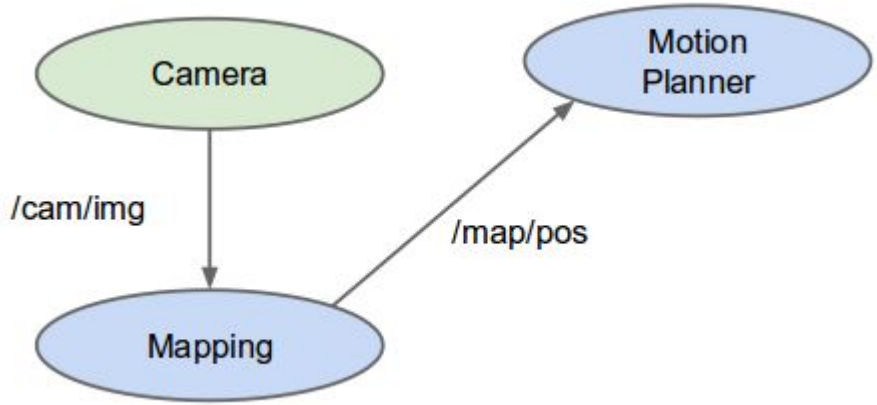
Arquitectura: Nodos



Arquitectura: Tópicos

-Flujos de datos con publicación/suscripción semántica.

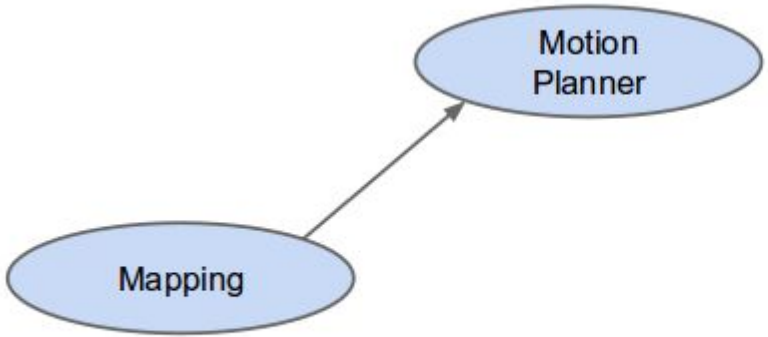
-Ellos son los únicos que pueden ser identificados por su nombre.



Arquitectura: Mensajes

-Un mensaje es simplemente una estructura de datos, que comprende los tipos de campos.

-Lenguaje de representación de datos agnóstico. C++ puede hablar con Python.



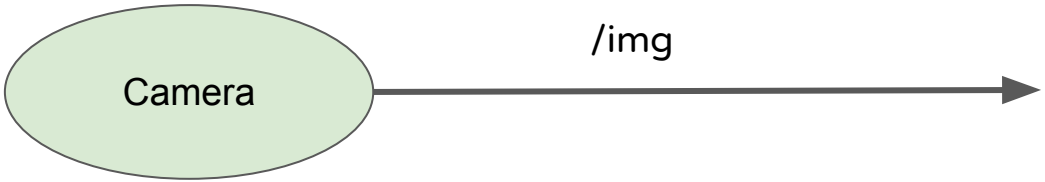
File: pos.msg

```
string robotName
uint32 posX
uint32 posY
uint32 goalX
uint32 goalY
```

Duckietown Engineering Chile

Arquitectura: Publisher

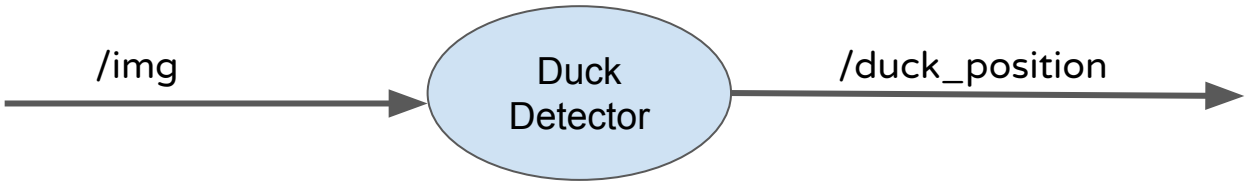
- Un publisher es un tipo de nodo que publica en uno o más tópicos.



Arquitectura: Subscriber

- Un subscriber es un tipo de nodo que escucha mensajes de uno o más tópicos.

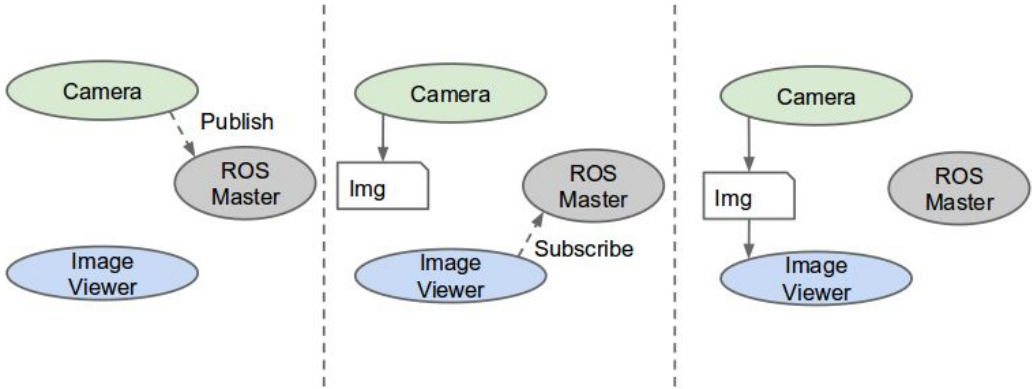
¿ Y Si quiero Publicar y Leer ?



Arquitectura: ROS Master

Proporciona el registro de nombres y la búsqueda de nodos. Sin el Master, los nodos no serían capaces de encontrar e intercambiar mensajes, o invocar servicios.

```
$ roscore
```



Duckietown Engineering Chile

Ejemplo : Publicación

```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) # 10hz
    while not rospy.is_shutdown():
        hello_str = "hello world %s" % rospy.get_time()
        rospy.loginfo(hello_str)
        pub.publish(hello_str)
        rate.sleep()

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

Ejemplo : Subscripción

```python
#!/usr/bin/env python
import rospy
from std_msgs.msg import String

def callback(data):
    rospy.loginfo(rospy.get_caller_id() + "I heard %s", data.data)

def listener():

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

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

    # spin() simply keeps python from exiting until this node is stopped
    rospy.spin()

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

Duckietown Engineering Chile

DEMO!!!

Duckietown
Engineering
Chile

Resumen: Comunicación Inter-Nodos

-Los nodos pueden publicar y suscribirse a tópicos

-Los tópicos se identifican por su nombre absoluto. Ej: “/chatter”

-El master se encarga de que los mensajes enviados a un tópico lleguen a sus suscriptores

Actividad de hoy

Implementando funcionalidades en el duckiebot

Ahora probamos Byobu

- Conexión SSH
- $ cd duckietown
- $ source environment.sh
- $ roscore

Ahora probamos Byobu

- Abrir nueva terminal con byobu <F2>

- `$ source environment.sh`

- `$ roslaunch ros_cap duckie_core.launch veh:=duckiebot`

Duckietown Engineering Chile

Checklist

- Ingresar al duckiebot ssh -X duckiebot@duckiebot.local

- Abrir **byobu**

- `cd duckietown`

- `source environment.sh`

- `roscore`

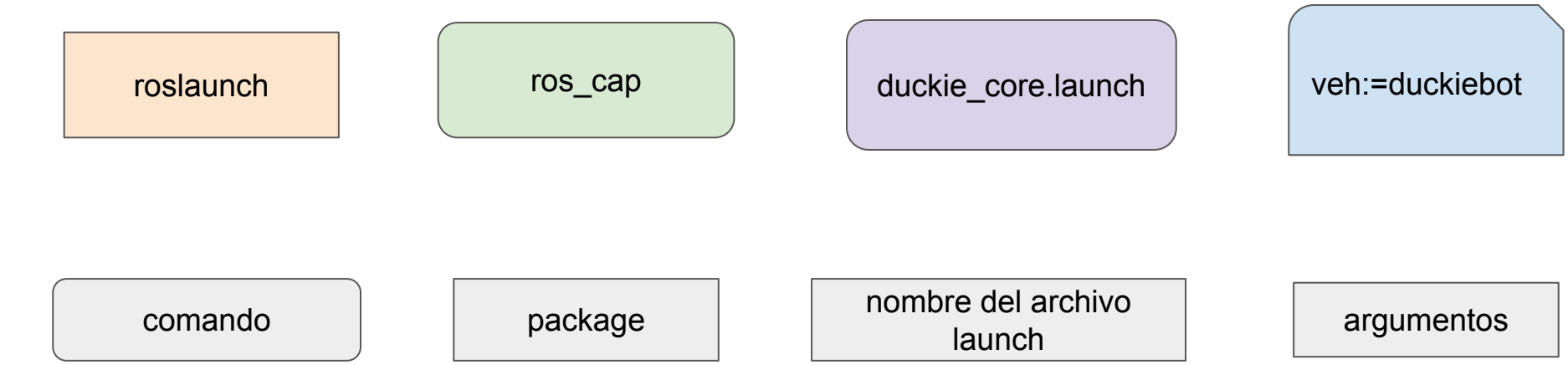
user: duckiebot
pwd: quackquack

Duckietown
Engineering
Chile

- Abrir nueva terminal con byobu (F2)

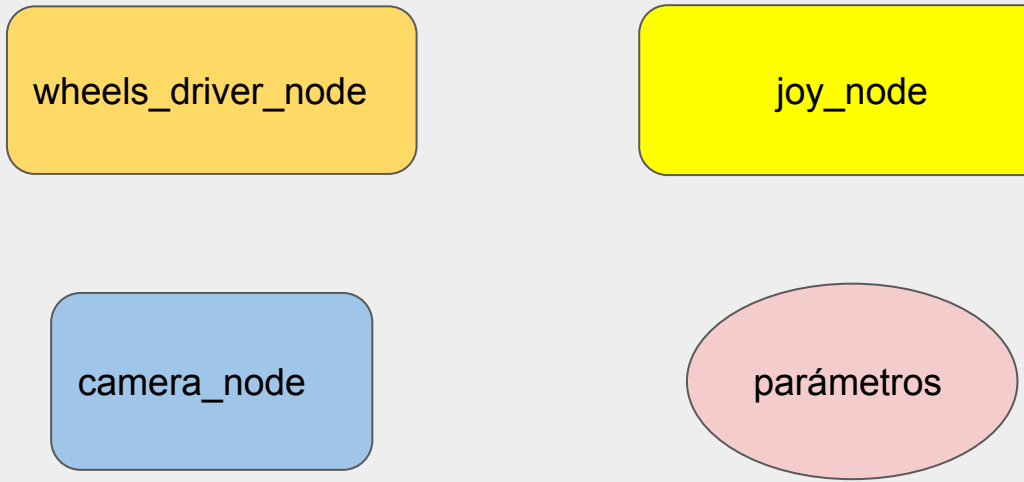
- `source environment.sh`

- `roslaunch ros_cap duckie_core.launch veh:=duckiebot`



Duckietown Engineering Chile

roslaunch ros_cap duckie_core.launch veh:=duckiebot



std_msgs/Joy

std_msgs/Header header

float32[] axes

int32[] buttons

duckietown_msgs/Twist2DStamped

Header header

float32 v

float32 omega

Desafío para la casa

> Correr turtlesim_node del package turtlesim

$ rosrun turtlesim turtlesim_node

> Correr turtlesim_teleop_key del package turtlesim

$ rosrun turtlesim turtlesim_teleop_key

> Utilizar las flechas del teclado para mover a la tortuguita

> Utilizar rostopic list / echo … etc para hacerse una idea de cómo funciona.

HackerSystem's

GIT, ROS y Otros