

Práctica ROS

Sandra Gómez Gálvez

Conducción cooperativa, conectada y autónoma
Universidad Politécnica de Madrid
2020/21

15 de enero de 2021

Índice general

1. Introducción	4
2. Desarrollo	5
2.1. Nodo (dummy) velocidad: Publisher	5
2.2. Nodo cámara: Publisher	11
2.3. Nodo ADAS: Subscriber	13
2.4. Nodo Decisiones: Servidor	16
2.5. Nodo ADAS: Cliente	19
3. Resultado	21
3.1. Launcher	23

Índice de figuras

2.1. Paso 1	5
2.2. Paso 2	6
2.3. Paso 3	6
2.4. Paso 4	6
2.5. Paso 5	6
2.6. Paso 6	7
2.7. Paso 7	7
2.8. Paso 8	7
2.9. Paso 9	7
2.10. Paso 10	8
2.11. Paso 11	8
2.12. Paso 12	8
2.13. Paso 13	8
2.14. Paso 14	8
2.15. Paso 15 - Nodo Velocidad Parte 1	9
2.16. Paso 16 - Nodo Velocidad Parte 2	9
2.17. Paso 17	9
2.18. Paso 18	10
2.19. Paso 19	10
2.20. Paso 20	10
2.21. Paso 21	11
2.22. Paso 22	11
2.23. Paso 23	11
2.24. Paso 24	12
2.25. Paso 25	12
2.26. Paso 26	12
2.27. Paso 27 - Nodo Cámara Parte 1	12
2.28. Paso 28 - Nodo Cámara Parte 2	13
2.29. Paso 29	13
2.30. Paso 30	13
2.31. Paso 31	13
2.32. Paso 32	14
2.33. Paso 33 - Nodo ADAS	14
2.34. Paso 34	14
2.35. Paso 35	14
2.36. Paso 36	15
2.37. Paso 37 - Resultado 1 publishers/subscriber	15

2.38. Paso 38 - Resultado 2 publishers/subscriber	16
2.39. Paso 39	16
2.40. Paso 40	17
2.41. Paso 41	17
2.42. Paso 42	17
2.43. Paso 43	17
2.44. Paso 44 - Nodo decisiones	18
2.45. Paso 45	18
2.46. Paso 46	18
2.47. Paso 47	19
2.48. Paso 48 - Nodo ADAS Parte 1	19
2.49. Paso 49 - Nodo ADAS Parte 2	20
3.1. Conexión nodos decisiones, adas y cámara	21
3.2. Conexión todos los nodos - Velocidad cámara = 200 y velocidad coche = 83. Respuesta: OK!	22
3.3. Conexión todos los nodos - Velocidad cámara = 20 y velocidad coche = 121. Respuesta: SPEEDING!! 101	23
3.4. Creación Launcher 1	23
3.5. Creación Launcher 2	24

Capítulo 1

Introducción

El objetivo de esta práctica es desarrollar una funcionalidad para el ADAS (*Advanced driver-assistance systems* o Sistemas avanzados de asistencia al conductor) de un vehículo.

La cámara del vehículo, al acercarse a una señal de límite de velocidad, indentificará la velocidad límite. Y si el vehículo circula a mayor velocidad que la indicadá, el ADAS deberá lanzar una alarma con la situación y el exceso de velocidad.

Para llevar a cabo este objetivo se tendrán que desarrollar nodos, mensajes, servicios y un launcher. Tendrá que componerse de lo siguiente:

- **Nodo (dummy) velocidad:** Nodo (dummy) que informa de la velocidad actual del vehículo a una frecuencia de 10Hz.
- **Nodo ADAS:** Nodo que representa el ADAS.
- **Nodo decisiones:** Nodo de decisiones del vehículo.
- **Envío mensajes:** Para la cámara se utilizará el envío de mensajes a través de la terminal.

Capítulo 2

Desarrollo

En este capítulo se desarrollará todo lo indicado en el capítulo 1.
Se propone lo siguiente:

- El nodo (dummy) de velocidad será un **publisher** que informe de la velocidad actual (será un random limitado por arriba) del vehículo a una frecuencia de 10Hz.
- El nodo ADAS un será un **subscriber** del nodo de velocidad y de la cámara. También será un **cliente** que enviará esta información al nodo de decisiones para obtener una respuesta y publicar por pantalla lo recibido: alarma con la situación y exceso de velocidad.
- El nodo de decisiones será un **servicio** e indicará lo que debe mostrar el nodo ADAS.
- El nodo de la cámara será un **publisher** que enviará las velocidades introducidas a través de la terminal.

2.1. Nodo (dummy) velocidad: Publisher

Se comienza con el nodo de velocidad ya que es el más fácil.

Para ello, vamos a la shell y creamos un workspace llamado `ros_prac_ws` (`cd ~ && mkdir ros_prac_ws`), el directorio `src` (`ros_prac_ws && mkdir src`), el espacio de trabajo (`colcon build`), y cargamos el fichero `setup.bash` (`source install/setup.bash`).

```
root@foxy:~# cd ~ && mkdir ros_prac_ws
root@foxy:~# cd ros_prac_ws && mkdir src
root@foxy:~/ros_prac_ws# colcon build

Summary: 0 packages finished [0.08s]
root@foxy:~/ros_prac_ws# ls
build  install  log  src
root@foxy:~/ros_prac_ws# source install/setup.bash
```

Figura 2.1: Paso 1

Para esta práctica, tendremos en cuenta que el fichero `.bashrc` ya está configurado.

Seguidamente, vamos a `src` (`cd src`) y creamos el paquete `super_pkg` (`ros2 pkg create super_pkg --build-type ament_python --dependencies rclpy`).

```

root@foxy:~/ros_prac_ws# cd src
root@foxy:~/ros_prac_ws/src# ls
root@foxy:~/ros_prac_ws/src# ros2 pkg create super_pkg --build-type ament_python --dependencies rclpy
going to create a new package
package name: super_pkg
destination directory: /root/ros_prac_ws/src
package format: 3
version: 0.0.0
description: TODO: Package description
maintainer: ['root <root@todo.todo>']
licenses: ['TODO: License declaration']

```

Figura 2.2: Paso 2

```

root@foxy:~/ros_prac_ws/src# ls
super_pkg
root@foxy:~/ros_prac_ws/src# cd super_pkg
root@foxy:~/ros_prac_ws/src/super_pkg# ls
package.xml resource setup.cfg setup.py super_pkg test

```

Figura 2.3: Paso 3

```

root@foxy:~/ros_prac_ws/src/super_pkg# cd ..
root@foxy:~/ros_prac_ws/src# cd ..
root@foxy:~/ros_prac_ws# colcon build
Starting >>> super_pkg
Finished <<< super_pkg [0.43s]

Summary: 1 package finished [0.52s]
root@foxy:~/ros_prac_ws# ls install -l
total 68
-rw-r--r-- 1 root root    0 ene 14 14:53 COLCON_IGNORE
-rw-r--r-- 1 root root 3331 ene 14 16:30 local_setup.bash
-rw-r--r-- 1 root root 2008 ene 14 16:30 local_setup.ps1
-rw-r--r-- 1 root root 3720 ene 14 16:30 local_setup.sh
-rw-r--r-- 1 root root 13537 ene 14 16:30 _local_setup_util_ps1.py
-rw-r--r-- 1 root root 13621 ene 14 16:30 _local_setup_util_sh.py
-rw-r--r-- 1 root root 3726 ene 14 16:30 local_setup.zsh
-rw-r--r-- 1 root root 1562 ene 14 16:30 setup.bash
-rw-r--r-- 1 root root 1337 ene 14 16:30 setup.ps1
-rw-r--r-- 1 root root 2338 ene 14 16:30 setup.sh
-rw-r--r-- 1 root root 1546 ene 14 16:30 setup.zsh
drwxr-xr-x 4 root root 4096 ene 14 16:30 super_pkg

```

Figura 2.4: Paso 4

Antes de crear el nodo velocidad, vamos a crear el mensaje que enviará este nodo velocidad. Para ello, tenemos que crear un nuevo paquete llamado interfaces del mismo modo que creamos el anterior. Eliminamos lo inservible y creamos el directorio msg.

```

root@foxy:~/ros_prac_ws/src# cd ..
root@foxy:~/ros_prac_ws# cd src
root@foxy:~/ros_prac_ws/src# ros2 pkg create interfaces
going to create a new package
package name: interfaces
destination directory: /root/ros_prac_ws/src
package format: 3
version: 0.0.0
description: TODO: Package description

```

Figura 2.5: Paso 5

```

root@foxy:~/ros_prac_ws/src# cd interfaces
root@foxy:~/ros_prac_ws/src/interfaces# ls
CMakeLists.txt  include  package.xml  src
root@foxy:~/ros_prac_ws/src/interfaces# rm -rf include/ src/
root@foxy:~/ros_prac_ws/src/interfaces# ls
CMakeLists.txt  package.xml
root@foxy:~/ros_prac_ws/src/interfaces# mkdir msg
root@foxy:~/ros_prac_ws/src/interfaces# ls
CMakeLists.txt  msg  package.xml
root@foxy:~/ros_prac_ws/src/interfaces# vi package.xml

```

Figura 2.6: Paso 6

Situados en el directorio interfaces abrimos package.xml y escribimos las siguientes dependencias para crear una interfaz:

```

root@foxy: ~/ros_prac_ws/src/interfaces# vi package.xml
<name>interfaces</name>
<version>0.0.0</version>
<description>TODO: Package description</description>
<maintainer email="root@todo.todo">root</maintainer>
<license>TODO: License declaration</license>

<buildtool_depend>ament_cmake</buildtool_depend>

<build_depend>rosidl_default_generators</build_depend>
<exec_depend>rosidl_default_runtime</exec_depend>
<member_of_group>rosidl_interface_packages</member_of_group>

<test_depend>ament_lint_auto</test_depend>
<test_depend>ament_lint_common</test_depend>

<export>
  <build_type>ament_cmake</build_type>
</export>
</package>
~

```

Figura 2.7: Paso 7

Y añadimos los siguientes paquetes al fichero CMakeLists.txt;

```

root@foxy:~/ros_prac_ws/src/interfaces# vi CMakeLists.txt

```

Figura 2.8: Paso 8

```

root@foxy: ~/ros_prac_ws/src/interfaces# vi CMakeLists.txt
if(CMAKE_COMPILER_IS_GNUCXX OR CMAKE_CXX_COMPILER_ID MATCHES "Clang")
  add_compile_options(-Wall -Wextra -Wpedantic)
endif()

# find dependencies
find_package(ament_cmake REQUIRED)
find_package(rosidl_default_generators REQUIRED)

rosidl_generate_interfaces(${PROJECT_NAME}
  "msg/HardwareStatus.msg"
)

```

Figura 2.9: Paso 9

Creamos el mensaje VelocityCar en el directorio msg.

```
root@foxy:~/ros_prac_ws/src/interfaces# cd msg
root@foxy:~/ros_prac_ws/src/interfaces/msg# touch VelocityCar.msg
root@foxy:~/ros_prac_ws/src/interfaces/msg# vi VelocityCar.msg
```

Figura 2.10: Paso 10

```
# Return velocity, if is_working and debug

int64 vel_car
bool is_working
string debug
```

Figura 2.11: Paso 11

Volvemos al archivo CMakeLists y añadimos el nuevo mensaje.

```
root@foxy:~/ros_prac_ws/src/interfaces/msg# cd ..
root@foxy:~/ros_prac_ws/src/interfaces# ls
CMakeLists.txt  msg  package.xml
root@foxy:~/ros_prac_ws/src/interfaces# vi CMakeLists.txt
```

Figura 2.12: Paso 12

```
add_compile_options(-Wall -Wextra -Wpedantic)
endif()

# find dependencies
find_package(ament_cmake REQUIRED)
find_package(rosidl_default_generators REQUIRED)

rosidl_generate_interfaces(${PROJECT_NAME}
  "msg/VelocityCar.msg"
)

# uncomment the following section in order to fill in
# further dependencies manually.
# find_package(<dependency> REQUIRED)
```

Figura 2.13: Paso 13

Ahora volvemos con cd al directorio ros_prac_ws y lo construimos con colcon build.

Ya está todo preparado para poder crear el nodo de velocidad. Nos dirigimos (con cd) a ros_prac_ws/src/super_pkg/super_pkg/ y aquí crearemos el nodo (vi velocidad.py).

```
root@foxy:~/ros_prac_ws# cd src/super_pkg/super_pkg
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# ls
__init__.py
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# vi velocity.py
```

Figura 2.14: Paso 14

Este nodo será un publisher que publicará una velocidad aleatoria en el que el minimo de ese rango irá aumentando con el paso del tiempo y el máximo será constante.

```
root@foxy: ~/ros_prac_ws/src/super_pkg/supe
root@foxy: ~/ros_prac_ws/src/super_pkg/super_p
#!/usr/bin/env python3

import rclpy
import random
from rclpy.node import Node

from interfaces.msg import VelocityCar

class VelocityNode(Node):
    def __init__(self):
        super().__init__('py_velocity')
        self.get_logger().info('Inititalizing py_velocity')
        self.vel = 0
        self.minVel=40
        self.maxVel=200
        self.create_timer(0.1, self.publish)
        self.publisher = self.create_publisher(VelocityCar, "velocity_topic", 10)

    def publish(self):
        self.vel = random.randint(self.minVel,self.maxVel)
        self.get_logger().warning(f'Velocidad Actual:{self.vel}')
        msg = VelocityCar()
        msg.vel_car = self.vel
        msg.is_working = True
        msg.debug = 'debugeando'
        self.publisher.publish(msg)
```

Figura 2.15: Paso 15 - Nodo Velocidad Parte 1

```
def main(args=None):
    try:
        rclpy.init(args=args)
        node = VelocityNode()
        rclpy.spin(node)
    finally:
        rclpy.shutdown()
if __name__ == '__main__':
    main()
```

Figura 2.16: Paso 16 - Nodo Velocidad Parte 2

Ahora, vamos a package.xml y añadimos la dependencia de interfaces.

```
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# cd ..
root@foxy:~/ros_prac_ws/src/super_pkg# ls
package.xml  resource  setup.cfg  setup.py  super_pkg  test
root@foxy:~/ros_prac_ws/src/super_pkg# vi package.xml
```

Figura 2.17: Paso 17

```

root@foxy:~$ cat package.xml
<description>TODO: Package description</description>
<maintainer email="root@todo.todo">root</maintainer>
Archivos e>TODO: License declaration</license>

<depend>roscpp</depend>
<depend>std_msgs</depend>

<test_depend>ament_copyright</test_depend>
<test_depend>ament_flake8</test_depend>
<test_depend>ament_pep257</test_depend>
<test_depend>python3-pytest</test_depend>

```

Figura 2.18: Paso 18

Y vamos a setup.py y añadimos el nodo.

```

root@foxy:~$ cd ~/ros2_ws/src/super_pkg
root@foxy:~/ros2_ws/src/super_pkg# ls
package.xml resource setup.cfg setup.py super_pkg test
root@foxy:~/ros2_ws/src/super_pkg# vi setup.py

```

Figura 2.19: Paso 19

```

root@foxy:~/ros2_ws/src/super_pkg# cat setup.py
from setuptools import setup

setup(
    name='super_pkg',
    version='0.0.0',
    install_requires=['setuptools'],
    zip_safe=True,
    maintainer='root',
    maintainer_email='root@todo.todo',
    description='TODO: Package description',
    license='TODO: License declaration',
    tests_require=['pytest'],
    entry_points={
        'console_scripts': [
            'velocity = super_pkg.velocity:main',
        ],
    },
)

```

Figura 2.20: Paso 20

Volvemos al directorio del workspace y hacemos colcon build, y probamos si funciona el publisher ejecutando `ros2 run super_pkg velocity`.

```

root@foxy:~/ros_prac_ws# source install/setup.bash
root@foxy:~/ros_prac_ws# colcon build
Starting >>> interfaces
Finished <<< interfaces [0.41s]
Starting >>> super_pkg
Finished <<< super_pkg [0.39s]

Summary: 2 packages finished [0.88s]
root@foxy:~/ros_prac_ws# ros2 run super_pkg velocity
[INFO] [1610643658.796635949] [py_velocity]: Initizalizing py_velocity
[WARN] [1610643658.897761390] [py_velocity]: Velocidad Actual:46
[WARN] [1610643659.001151774] [py_velocity]: Velocidad Actual:118
[WARN] [1610643659.099094076] [py_velocity]: Velocidad Actual:195
[WARN] [1610643659.200559204] [py_velocity]: Velocidad Actual:154
[WARN] [1610643659.299486083] [py_velocity]: Velocidad Actual:162
[WARN] [1610643659.398424570] [py_velocity]: Velocidad Actual:193
[WARN] [1610643659.499546806] [py_velocity]: Velocidad Actual:115
[WARN] [1610643659.600169755] [py_velocity]: Velocidad Actual:91
[WARN] [1610643659.699829861] [py_velocity]: Velocidad Actual:183

```

Figura 2.21: Paso 21

Se ejecuta correctamente.

2.2. Nodo cámara: Publisher

De manera parecida vamos a crear el nodo publisher de la cámara. Los 4 nodos se van a crear en el mismo paquete, aunque si fuera una aplicación más grande lo más correcto sería hacerlo en pequeños paquetes.

Primero vamos a crear el mensaje que utilizará este nodo. Vamos al directorio msg y lo creamos con nombre VelocityCamera.msg .

```

root@foxy:~/ros_prac_ws/src/interfaces# cd msg
root@foxy:~/ros_prac_ws/src/interfaces/msg# ls
VelocityCar.msg
root@foxy:~/ros_prac_ws/src/interfaces/msg# vi VelocityCamera.msg

```

Figura 2.22: Paso 22

```

# Return velocity, if is_working and debug

int64 vel_cam
bool is_working
string debug

```

Figura 2.23: Paso 23

Y al igual que antes, lo declaramos en CMakeLists.txt

```

root@foxy:~/ros_prac_ws/src/interfaces# ls
CMakeLists.txt  msg  package.xml
root@foxy:~/ros_prac_ws/src/interfaces# vi CMakeLists.txt

```

Figura 2.24: Paso 24

```

find_package(ament_cmake REQUIRED)
find_package(rosidl_default_generators REQUIRED)

rosidl_generate_interfaces(${PROJECT_NAME}
  "msg/VelocityCar.msg"
  "msg/VelocityCamera.msg"
)

# uncomment the following section in order to fill in
# further dependencies manually.
# find_package(<dependency> REQUIRED)

```

Figura 2.25: Paso 25

Y en el workspace hacemos colcon build.

Ahora ya podemos crear el nodo cámara. Se utiliza el nodo velocidad como plantilla, por tanto copiamos este nodo para crear el nodo cámara.

```

root@foxy:~/ros_prac_ws/src/super_pkg# cd super_pkg
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# ls
__init__.py  velocity.py
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# cp velocity.py camera.py
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# vi camera.py

```

Figura 2.26: Paso 26

Este nodo será un publisher que recogerá velocidades introducidas por el terminal y las enviará.

```

#!/usr/bin/env python3

import rclpy
import random
from rclpy.node import Node

from interfaces.msg import VelocityCamera

class VelocityCameraNode(Node):
    def __init__(self):
        super().__init__('py_velocity_cam')
        self.get_logger().info('Initizalizing py_velocity_cam')
        self.vel = 0
        self.create_timer(0.1, self.publish)
        self.publisher = self.create_publisher(VelocityCamera, "velocity_cam_topic", 10)

    def publish(self):
        self.get_logger().info('Enter a speed value')
        self.vel = int(input())
        self.get_logger().warning(f'Max Velocity:{self.vel}')
        msg = VelocityCamera()
        msg.vel_cam = self.vel
        msg.is_working = True
        msg.debug = 'debugando'
        self.publisher.publish(msg)

```

Figura 2.27: Paso 27 - Nodo Cámara Parte 1

```
def main(args=None):
    try:
        rclpy.init(args=args)
        node = VelocityCameraNode()
        rclpy.spin(node)
    finally:
        rclpy.shutdown()
if __name__ == '__main__':
    main()
```

Figura 2.28: Paso 28 - Nodo Cámara Parte 2

Y añadimos el nodo en el setup.py.

```
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# cd ..
root@foxy:~/ros_prac_ws/src/super_pkg# ls
package.xml resource setup.cfg setup.py super_pkg test
root@foxy:~/ros_prac_ws/src/super_pkg# vi setup.py
```

Figura 2.29: Paso 29

```
tests_require=['pytest'],
entry_points={
    'console_scripts': [
        'velocity = super_pkg.velocity:main',
        'camera = super_pkg.camera:main'
    ],
},
)
```

Figura 2.30: Paso 30

Finalmente hacemos colcon build y comprobamos si funciona.

```
root@foxy:~/ros_prac_ws# colcon build
Starting >>> interfaces
Finished <<< interfaces [0.45s]
Starting >>> super_pkg
Finished <<< super_pkg [0.42s]

Summary: 2 packages finished [0.95s]
root@foxy:~/ros_prac_ws# ros2 run super_pkg camera
[INFO] [1610646022.563532776] [py_velocity_cam]: Initalizing py_velocity_cam
[INFO] [1610646022.667297340] [py_velocity_cam]: Enter a speed value
40
[WARN] [1610646026.234121066] [py_velocity_cam]: Max Velocity:40
[INFO] [1610646026.235364743] [py_velocity_cam]: Enter a speed value
100
[WARN] [1610646031.683902565] [py_velocity_cam]: Max Velocity:100
[INFO] [1610646031.685149809] [py_velocity_cam]: Enter a speed value
```

Figura 2.31: Paso 31

2.3. Nodo ADAS: Subscriber

Ya creados los nodos publisher de velocidad y cámara, se pasará a crear el nodo ADAS del vehículo que en primer lugar será un subscriber que recogerá la información de los publisher. Y

más adelante se desarrollará como cliente para una comunicación entre el nodo de decisiones y este nodo.

Se comenzará creando el nodo como subscriber. Lo crearemos en el mismo paquete. Para ello, vamos a copiar el nodo dummy y lo modificaremos.

```
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# cp velocity.py adas.py
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# vi adas.py
```

Figura 2.32: Paso 32

```
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg 203x29
import rclpy
import random
from rclpy.node import Node

from interfaces.msg import VelocityCar
from interfaces.msg import VelocityCamera

class ADASNode(Node):
    def __init__(self):
        super().__init__('py_adas')
        self.get_logger().info('Initializing py_adas')
        self.subscriber_vel_car = self.create_subscription(VelocityCar, "velocity_topic", self.callbackVelocityCar, 10)
        self.subscriber_vel_cam = self.create_subscription(VelocityCamera, "velocity_cam_topic", self.callbackVelocityCamera, 10)
    def callbackVelocityCar(self, msg):
        self.get_logger().info(f'VeLOCITY car: {msg.vel_car}')
    def callbackVelocityCamera(self, msg):
        self.get_logger().warning(f'VeLOCITY camera: {msg.vel_cam}')
def main(args=None):
    try:
        rclpy.init(args=args)
        node = ADASNode()
        rclpy.spin(node)
    finally:
        rclpy.shutdown()
if __name__ == '__main__':
    main()
```

Figura 2.33: Paso 33 - Nodo ADAS

Una vez creado nos dirigimos a setup.py e introducimos el nodo.

```
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# cd ..
root@foxy:~/ros_prac_ws/src/super_pkg# ls
package.xml resource setup.cfg setup.py super_pkg test
root@foxy:~/ros_prac_ws/src/super_pkg# vi setup.py
```

Figura 2.34: Paso 34

```
maintainer_email='root@todo.todo',
description='TODO: Package description',
license='TODO: License declaration',
tests_require=['pytest'],
entry_points={
    'console_scripts': [
        'velocity = super_pkg.velocity:main',
        'camera = super_pkg.camera:main',
        'adas = super_pkg.adas:main',
    ],
}
```

Figura 2.35: Paso 35

Finalmente en el workspace realizamos colcon build y comprobamos que el subscriber funciona perfectamente. Para esto último, se hace run de los 3 nodos.

```

root@foxy:~/ros_prac_ws/src# cd ..
root@foxy:~/ros_prac_ws# colcon build
Starting >>> interfaces
Finished <<< interfaces [1.67s]
Starting >>> super_pkg
Finished <<< super_pkg [1.34s]

Summary: 2 packages finished [3.33s]

```

Figura 2.36: Paso 36

```

root@foxy: ~/ros_prac_ws
root@foxy:~/ros_prac_ws 203x20

root@foxy:~/ros_prac_ws# ros2 run super_pkg adas
[INFO] [1610702724.679611512] [py_adas]: Initializing py_adas
[WARN] [1610702742.268714784] [py_adas]: Velocity camera:100
[INFO] [1610702763.847948760] [py_adas]: Velocity car:46
[INFO] [1610702763.945874496] [py_adas]: Velocity car:167
[INFO] [1610702764.051700244] [py_adas]: Velocity car:195
[INFO] [1610702764.145668849] [py_adas]: Velocity car:81
[INFO] [1610702764.249327266] [py_adas]: Velocity car:194
[INFO] [1610702764.346371782] [py_adas]: Velocity car:78
[INFO] [1610702764.446617405] [py_adas]: Velocity car:159
[INFO] [1610702764.546871049] [py_adas]: Velocity car:68
[INFO] [1610702764.646888051] [py_adas]: Velocity car:170
[INFO] [1610702764.745641109] [py_adas]: Velocity car:108
[INFO] [1610702764.848255649] [py_adas]: Velocity car:193
[INFO] [1610702764.948415677] [py_adas]: Velocity car:152
[INFO] [1610702765.045960207] [py_adas]: Velocity car:103
[INFO] [1610702765.150239156] [py_adas]: Velocity car:80
[INFO] [1610702765.251482663] [py_adas]: Velocity car:134
[INFO] [1610702765.348399272] [py_adas]: Velocity car:195
[INFO] [1610702765.446916468] [py_adas]: Velocity car:84

root@foxy:~/ros_prac_ws 203x10

root@foxy:~/ros_prac_ws# ros2 run super_pkg camera
[INFO] [1610702739.039398722] [py_velocity_cam]: Initializing py_velocity_cam
[INFO] [1610702739.147572016] [py_velocity_cam]: Enter a speed value
100
[WARN] [1610702742.265859165] [py_velocity_cam]: Max Velocity:100
[INFO] [1610702742.267958641] [py_velocity_cam]: Enter a speed value
200
[WARN] [1610702768.351770728] [py_velocity_cam]: Max Velocity:200
[INFO] [1610702768.353445111] [py_velocity_cam]: Enter a speed value

root@foxy:~/ros_prac_ws 203x14

root@foxy:~/ros_prac_ws# ros2 run super_pkg velocity
[INFO] [1610702763.740840842] [py_velocity]: Initializing py_velocity
[WARN] [1610702763.844703101] [py_velocity]: Velocity:46
[WARN] [1610702763.943714082] [py_velocity]: Velocity:167
[WARN] [1610702764.047297065] [py_velocity]: Velocity:195
[WARN] [1610702764.143111907] [py_velocity]: Velocity:81
[WARN] [1610702764.246153441] [py_velocity]: Velocity:194
[WARN] [1610702764.343416052] [py_velocity]: Velocity:78
[WARN] [1610702764.443475093] [py_velocity]: Velocity:159
[WARN] [1610702764.544681729] [py_velocity]: Velocity:68
[WARN] [1610702764.644803018] [py_velocity]: Velocity:170
[WARN] [1610702764.743681711] [py_velocity]: Velocity:108
[WARN] [1610702764.845399052] [py_velocity]: Velocity:193
[WARN] [1610702764.945144970] [py_velocity]: Velocity:152
[WARN] [1610702765.042401022] [py_velocity]: Velocity:103

```

Figura 2.37: Paso 37 - Resultado 1 publishers/subscriber


```

root@foxy: ~/ros_prac_ws
root@foxy: ~/ros_prac_ws 203x20
[INFO] [1610702767.749372587] [py_adas]: Velocity car:191
[INFO] [1610702767.846265778] [py_adas]: Velocity car:168
[INFO] [1610702767.946675615] [py_adas]: Velocity car:55
[INFO] [1610702768.050338215] [py_adas]: Velocity car:137
[INFO] [1610702768.144393660] [py_adas]: Velocity car:76
[INFO] [1610702768.244691487] [py_adas]: Velocity car:150
[INFO] [1610702768.344771982] [py_adas]: Velocity car:165
[WARN] [1610702768.353639311] [py_adas]: Velocity camera:200
[INFO] [1610702768.446250011] [py_adas]: Velocity car:52
[INFO] [1610702768.546660116] [py_adas]: Velocity car:70
[INFO] [1610702768.645622225] [py_adas]: Velocity car:84
[INFO] [1610702768.746627641] [py_adas]: Velocity car:196
[INFO] [1610702768.846560584] [py_adas]: Velocity car:56
[INFO] [1610702768.945759173] [py_adas]: Velocity car:147
[INFO] [1610702769.047287384] [py_adas]: Velocity car:122
[INFO] [1610702769.145167047] [py_adas]: Velocity car:72
[INFO] [1610702769.246663106] [py_adas]: Velocity car:149
[INFO] [1610702769.344748745] [py_adas]: Velocity car:135
[INFO] [1610702769.444860060] [py_adas]: Velocity car:197
[INFO] [1610702769.546011159] [py_adas]: Velocity car:83

root@foxy: ~/ros_prac_ws 203x10
root@foxy:~/ros_prac_ws# ros2 run super_pkg camera
[INFO] [1610702739.039398722] [py_velocity_cam]: Initializing py_velocity_cam
[INFO] [1610702739.147572016] [py_velocity_cam]: Enter a speed value
100
[WARN] [1610702742.265859165] [py_velocity_cam]: Max Velocity:100
[INFO] [1610702742.267958641] [py_velocity_cam]: Enter a speed value
200
[WARN] [1610702768.351770728] [py_velocity_cam]: Max Velocity:200
[INFO] [1610702768.353445111] [py_velocity_cam]: Enter a speed value

root@foxy:~/ros_prac_ws 203x14
[WARN] [1610702767.444865719] [py_velocity]: Velocity:66
[WARN] [1610702767.542899455] [py_velocity]: Velocity:194
[WARN] [1610702767.643258090] [py_velocity]: Velocity:168
[WARN] [1610702767.745775647] [py_velocity]: Velocity:191
[WARN] [1610702767.844100495] [py_velocity]: Velocity:168
[WARN] [1610702767.943549165] [py_velocity]: Velocity:55
[WARN] [1610702768.045117953] [py_velocity]: Velocity:137
[WARN] [1610702768.142423503] [py_velocity]: Velocity:76
[WARN] [1610702768.242778769] [py_velocity]: Velocity:150
[WARN] [1610702768.343148127] [py_velocity]: Velocity:165
[WARN] [1610702768.443457635] [py_velocity]: Velocity:52
[WARN] [1610702768.544776837] [py_velocity]: Velocity:70
[WARN] [1610702768.643500525] [py_velocity]: Velocity:84
[WARN] [1610702768.744798221] [py_velocity]: Velocity:196
[WARN] [1610702768.844798221] [py_velocity]: Velocity:56

```

Figura 2.38: Paso 38 - Resultado 2 publishers/subscriber

Más adelante crearemos este nodo como cliente.

2.4. Nodo Decisiones: Servidor

Este nodo se creará como servidor. Recibirá la velocidad del coche y la velocidad de la cámara, y si la velocidad de la cámara es mayor enviará una alerta, sino enviará un mensaje *OK!*.

Como este nodo necesitará un mensaje srv, lo crearemos. Primero, vamos al directorio interfaces y creamos el directorio srv. Nos dirigimos a este nuevo directorio, y creamos Speeding.msg :

```

root@foxy:~/ros_prac_ws/src/interfaces# cd srv
root@foxy:~/ros_prac_ws/src/interfaces/srv# vi Speeding.srv

```

Figura 2.39: Paso 39

```

# Return a response

int64 vel_cam
int64 vel_car
---
string res
~
~
~
~

```

Figura 2.40: Paso 40

Ahora vamos a CMakeLists y añadimos el nuevo mensaje.

```

root@foxy:~/ros_prac_ws/src/interfaces/srv# ls
Speeding.srv
root@foxy:~/ros_prac_ws/src/interfaces/srv# cd ..
root@foxy:~/ros_prac_ws/src/interfaces# ls
CMakeLists.txt  msg  package.xml  srv
root@foxy:~/ros_prac_ws/src/interfaces# vi CMakeLists.txt

```

Figura 2.41: Paso 41

```

if(CMAKE_COMPILER_IS_GNUCXX OR CMAKE_CXX_COMPILER_ID MATCHES "Clang")
  add_compile_options(-Wall -Wextra -Wpedantic)
endif()

# find dependencies
find_package(ament_cmake REQUIRED)
find_package(rosidl_default_generators REQUIRED)

rosidl_generate_interfaces(${PROJECT_NAME}
  "msg/VelocityCar.msg"
  "msg/VelocityCamera.msg"
  "srv/Speeding.srv"
)

# uncomment the following section in order to fill in
# further dependencies manually.
# find_package(<dependency> REQUIRED)

if(BUILD_TESTING)

```

Figura 2.42: Paso 42

Ya se puede crear el nodo decisiones. Este nodo será creado en el mismo paquete. Para ello, vamos a copiar el nodo dummy y lo modificaremos.

```

root@foxy:~/ros_prac_ws/src/super_pkg# cd super_pkg
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# ls
camera.py  __init__.py  velocity.py
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# cp velocity.py decisions.py
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# ls
camera.py  decisions.py  __init__.py  velocity.py
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# vi decisions.py

```

Figura 2.43: Paso 43

```

root@foxy: ~/ros_prac_ws/src/super_pkg/super_pkg
root@foxy: ~/ros_prac_ws/src/super_pkg/super_pkg 203x47
#!/usr/bin/env python3

import rclpy
import random
from rclpy.node import Node

from interfaces.srv import Speeding

class DecisionsNode(Node):
    def __init__(self):
        super().__init__('py_decisions')
        self.get_logger().info('Inicializing py_decisions')
        self.server = self.create_service(Speeding, "velocity_response", self.callbackResponseVelocity)
        self.get_logger().info('Decisions server has been started')
    def callbackResponseVelocity(self, request, response):
        if request.vel_cam < request.vel_car:
            self.excess = request.vel_car - request.vel_cam
            self.get_logger().info(f'{request.vel_car} - {request.vel_cam} = {self.excess}')
            response.res = f'SPEEDING!!! {self.excess}'
        else:
            self.get_logger().info('Ok')
            response.res = 'OK!'
        return response
def main(args=None):
    try:
        rclpy.init(args=args)
        node = DecisionsNode()
        rclpy.spin(node)
    finally:
        rclpy.shutdown()
if __name__ == '__main__':
    main()

```

Figura 2.44: Paso 44 - Nodo decisiones

Una vez creado nos dirigimos a setup.py e introducimos el nodo.

```

root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# cd ..
root@foxy:~/ros_prac_ws/src/super_pkg# ls
package.xml  resource  setup.cfg  setup.py  super_pkg  test
root@foxy:~/ros_prac_ws/src/super_pkg# vi setup.py

```

Figura 2.45: Paso 45

```

],
install_requires=['setuptools'],
zip_safe=True,
maintainer='root',
maintainer_email='root@todo.todo',
description='TODO: Package description',
license='TODO: License declaration',
tests_require=['pytest'],
entry_points={
    'console_scripts': [
        'velocity = super_pkg.velocity:main',
        'camera = super_pkg.camera:main',
        'adas = super_pkg.adas:main',
        'decisions = super_pkg.decisions:main',
    ],
},
),

```

Figura 2.46: Paso 46

Finalmente en el workspace realizamos colcon build.

2.5. Nodo ADAS: Cliente

El nodo ADAS ya ha sido creado como subscriber del nodo cámara y velocidad, pero además este nodo será cliente de un servidor que es el nodo de decisiones. Para ello enviará al cliente la velocidad de la cámara y del coche, y el servidor le dirá si está todo OK o si hay un exceso de velocidad.

Para ello, modificaremos el nodo ADAS ya creado.

```
root@foxy:~/ros_prac_ws# cd src/super_pkg/super_pkg
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# ls
adas.py camera.py decisions.py __init__.py velocity.py
root@foxy:~/ros_prac_ws/src/super_pkg/super_pkg# vi adas.py
```

Figura 2.47: Paso 47

```
root@foxy: ~/ros_prac_ws
#! /usr/bin/env python3

import rclpy
import random
from rclpy.node import Node
from functools import partial

from interfaces.msg import VelocityCar
from interfaces.msg import VelocityCamera
from interfaces.srv import Speeding

class ADASNode(Node):
    def __init__(self):
        super().__init__('py_adas')
        self.get_logger().info('Initializing py_adas')
        self.velo_car = 0
        self.velo_cam = 0
        self.subscriber_vel_car = self.create_subscription(VelocityCar, "velocity_topic", self.callbackVelocityCar, 10)
        self.subscriber_vel_cam = self.create_subscription(VelocityCamera, "velocity_cam_topic", self.callbackVelocityCamera, 10)
        #self.call_speeding_server(self.velo_car, self.velo_cam)

    def callbackVelocityCar(self, msg):
        self.velo_car = msg.vel_car
        self.get_logger().info(f'VeLOCITY car:{msg.vel_car}')

    def callbackVelocityCamera(self, msg):
        self.velo_cam = msg.vel_cam
        self.get_logger().warning(f'VeLOCITY camera:{msg.vel_cam}')
        self.call_speeding_server(self.velo_car, self.velo_cam)

    def call_speeding_server(self, car, cam):
        client = self.create_client(
            Speeding, "velocity_response"
        )
        while not client.wait_for_service(1.0):
            self.get_logger().warn(
                'Waiting for server'
            )
        request = Speeding.Request()
        request.vel_cam = self.velo_cam
        request.vel_car = self.velo_car

        future = client.call_async(request)
        future.add_done_callback(partial(
            self.callback_speeding, car = self.velo_car, cam = self.velo_cam
        ))
```

Figura 2.48: Paso 48 - Nodo ADAS Parte 1

```

def callback_speeding(self, future, car, cam):
    try:
        response = future.result()
        if(response!='OK!'):
            self.get_logger().warning(response.res)
        else:
            self.get_logger().info(response.res)
    except Exception as e:
        self.get_logger().error('{e}')
def main(args=None):
    try:
        rclpy.init(args=args)
        node = ADASNode()
        rclpy.spin(node)
    finally:
        rclpy.shutdown()
if __name__ == '__main__':
    main()

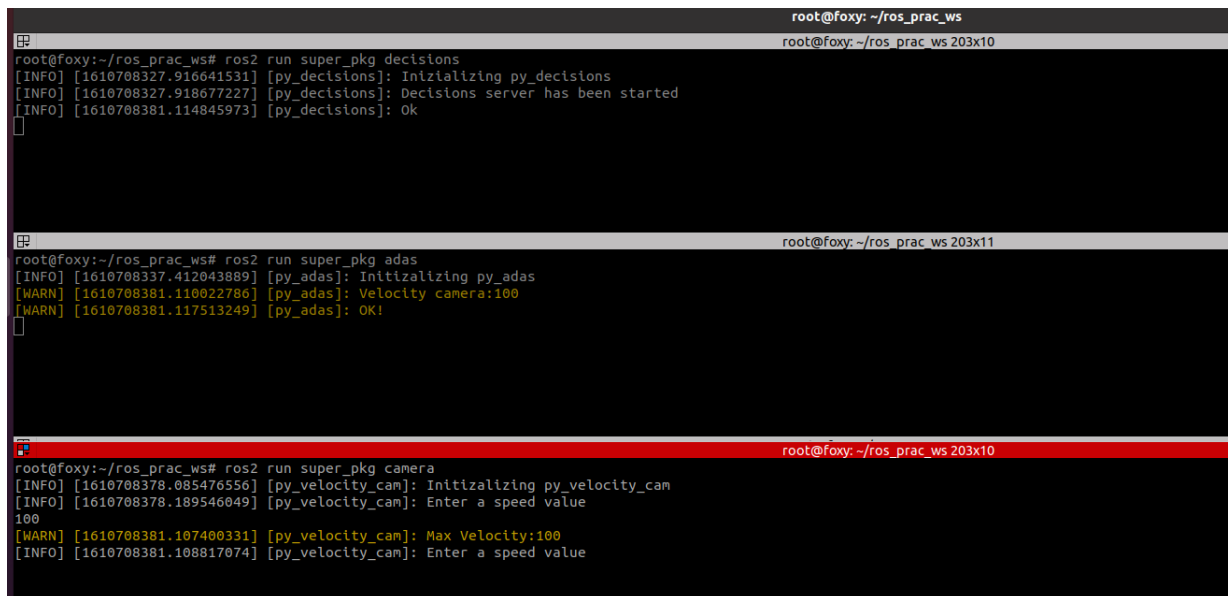
```

Figura 2.49: Paso 49 - Nodo ADAS Parte 2

Capítulo 3

Resultado

Ya con todos los nodos creados se verá el resultado. Para ello abriremos 4 shells distintas y ejecutaremos los respectivos runs de cada nodo. Comenzaremos ejecutando el nodo de decisiones, luego el ADAS, después el de la cámara, y finalmente el de velocidad.



```
root@foxy: ~/ros_prac_ws
root@foxy: ~/ros_prac_ws 203x10
root@foxy:~/ros_prac_ws# ros2 run super_pkg decisions
[INFO] [1610708327.916641531] [py_decisions]: Initializing py_decisions
[INFO] [1610708327.918677227] [py_decisions]: Decisions server has been started
[INFO] [1610708381.114845973] [py_decisions]: Ok
[]

root@foxy:~/ros_prac_ws# ros2 run super_pkg adas
root@foxy:~/ros_prac_ws 203x11
[INFO] [1610708337.412043889] [py_adas]: Initializing py_adas
[WARN] [1610708381.110022786] [py_adas]: Velocity camera:100
[WARN] [1610708381.117513249] [py_adas]: OK!
[]

root@foxy:~/ros_prac_ws# ros2 run super_pkg camera
root@foxy:~/ros_prac_ws 203x10
[INFO] [1610708378.085476556] [py_velocity_cam]: Initializing py_velocity_cam
[INFO] [1610708378.189546049] [py_velocity_cam]: Enter a speed value
100
[WARN] [1610708381.107400331] [py_velocity_cam]: Max Velocity:100
[INFO] [1610708381.108817074] [py_velocity_cam]: Enter a speed value
```

Figura 3.1: Conexión nodos decisiones, adas y cámara

```
root@foxy: ~/ros_prac_ws
[INFO] [1610708327.916641531] [py_decisions]: Inizializing py_decisions
[INFO] [1610708327.918677227] [py_decisions]: Decisions server has been started
[INFO] [1610708381.114845973] [py_decisions]: Ok
[INFO] [1610708432.670572256] [py_decisions]: Ok
root@foxy: ~/ros_prac_ws 203x17
[INFO] [1610708431.527156738] [py_adas]: Velocity car:41
[INFO] [1610708431.628739276] [py_adas]: Velocity car:178
[INFO] [1610708431.729228058] [py_adas]: Velocity car:150
[INFO] [1610708431.827661316] [py_adas]: Velocity car:154
[INFO] [1610708431.926968892] [py_adas]: Velocity car:61
[INFO] [1610708432.029300368] [py_adas]: Velocity car:106
[INFO] [1610708432.126240473] [py_adas]: Velocity car:198
[INFO] [1610708432.226748397] [py_adas]: Velocity car:84
[INFO] [1610708432.328234331] [py_adas]: Velocity car:116
[INFO] [1610708432.426991561] [py_adas]: Velocity car:172
[INFO] [1610708432.529099827] [py_adas]: Velocity car:193
[INFO] [1610708432.628973989] [py_adas]: Velocity car:83
[WARN] [1610708432.667181905] [py_adas]: Velocity camera:200
[WARN] [1610708432.672347204] [py_adas]: OK!
[INFO] [1610708432.725929152] [py_adas]: Velocity car:92
[INFO] [1610708432.828436803] [py_adas]: Velocity car:46
[INFO] [1610708432.927953630] [py_adas]: Velocity car:95
[INFO] [1610708432.927953630] [py_adas]: Velocity car:80
root@foxy: ~/ros_prac_ws 203x19
root@foxy:~/ros_prac_ws# ros2 run super_pkg camera
[INFO] [1610708378.085476556] [py_velocity_cam]: Inizializing py_velocity_cam
[INFO] [1610708378.189546049] [py_velocity_cam]: Enter a speed value
100
[WARN] [1610708381.107400331] [py_velocity_cam]: Max Velocity:100
[INFO] [1610708381.108817074] [py_velocity_cam]: Enter a speed value
200
[WARN] [1610708432.664619750] [py_velocity_cam]: Max Velocity:200
[INFO] [1610708432.665829869] [py_velocity_cam]: Enter a speed value
root@foxy: ~/ros_prac_ws 203x12
[WARN] [1610708432.027698434] [py_velocity]: Velocity:106
[WARN] [1610708432.124875899] [py_velocity]: Velocity:198
[WARN] [1610708432.225120816] [py_velocity]: Velocity:84
[WARN] [1610708432.326519708] [py_velocity]: Velocity:116
[WARN] [1610708432.425259740] [py_velocity]: Velocity:172
[WARN] [1610708432.527455524] [py_velocity]: Velocity:193
[WARN] [1610708432.627303427] [py_velocity]: Velocity:83
[WARN] [1610708432.724710465] [py_velocity]: Velocity:92
[WARN] [1610708432.826819025] [py_velocity]: Velocity:46
[WARN] [1610708432.926292872] [py_velocity]: Velocity:95
[WARN] [1610708433.026187223] [py_velocity]: Velocity:80
[WARN] [1610708433.125358381] [py_velocity]: Velocity:45
[WARN] [1610708433.225046481] [py_velocity]: Velocity:122
```

Figura 3.2: Conexión todos los nodos - Velocidad cámara = 200 y velocidad coche = 83. Respuesta: OK!

```
root@foxy: ~/ros_prac_ws
root@foxy: ~/ros_prac_ws 203x4
[INFO] [1610708435.465449243] [py_decisions]: Ok
[INFO] [1610708438.147805420] [py_decisions]: Ok
[INFO] [1610708440.231954639] [py_decisions]: 121 - 20 = 101
[INFO] [1610708442.089961939] [py_decisions]: 81 - 10 = 71
root@foxy: ~/ros_prac_ws 203x17
[INFO] [1610708439.327029719] [py_adas]: Velocity car:166
[INFO] [1610708439.427052681] [py_adas]: Velocity car:149
[INFO] [1610708439.527730030] [py_adas]: Velocity car:165
[INFO] [1610708439.627953486] [py_adas]: Velocity car:102
[INFO] [1610708439.727086232] [py_adas]: Velocity car:100
[INFO] [1610708439.827879064] [py_adas]: Velocity car:71
[INFO] [1610708439.927034224] [py_adas]: Velocity car:105
[INFO] [1610708440.027948986] [py_adas]: Velocity car:151
[INFO] [1610708440.127641583] [py_adas]: Velocity car:121
[WARN] [1610708440.223108557] [py_adas]: Velocity camera:20
[INFO] [1610708440.230922960] [py_adas]: Velocity car:48
[WARN] [1610708440.235317468] [py_adas]: SPEEDING!!! 101
[INFO] [1610708440.326553760] [py_adas]: Velocity car:55
[INFO] [1610708440.426915977] [py_adas]: Velocity car:66
[INFO] [1610708440.527202837] [py_adas]: Velocity car:92
[INFO] [1610708440.626963219] [py_adas]: Velocity car:170
[INFO] [1610708440.728275615] [py_adas]: Velocity car:129
[INFO] [1610708440.827652705] [py_adas]: Velocity car:133
root@foxy: ~/ros_prac_ws 203x9
20
[WARN] [1610708440.221566114] [py_velocity_cam]: Max Velocity:20
10
[INFO] [1610708440.222835123] [py_velocity_cam]: Enter a speed value
10
[WARN] [1610708442.082242797] [py_velocity_cam]: Max Velocity:10
6
[INFO] [1610708442.083623557] [py_velocity_cam]: Enter a speed value
6
[WARN] [1610708445.448629081] [py_velocity_cam]: Max Velocity:0
[INFO] [1610708445.449891873] [py_velocity_cam]: Enter a speed value
root@foxy: ~/ros_prac_ws 203x12
[WARN] [1610708444.626688154] [py_velocity]: Velocity:103
[WARN] [1610708444.724824764] [py_velocity]: Velocity:184
[WARN] [1610708444.825403794] [py_velocity]: Velocity:130
[WARN] [1610708444.926061276] [py_velocity]: Velocity:85
[WARN] [1610708445.027249604] [py_velocity]: Velocity:174
[WARN] [1610708445.126413528] [py_velocity]: Velocity:130
[WARN] [1610708445.225741700] [py_velocity]: Velocity:107
[WARN] [1610708445.325384355] [py_velocity]: Velocity:160
[WARN] [1610708445.426148185] [py_velocity]: Velocity:197
[WARN] [1610708445.524788860] [py_velocity]: Velocity:80
[WARN] [1610708445.625271031] [py_velocity]: Velocity:106
[WARN] [1610708445.726566407] [py_velocity]: Velocity:103
[WARN] [1610708445.826480918] [py_velocity]: Velocity:144
```

Figura 3.3: Conexión todos los nodos - Velocidad cámara = 20 y velocidad coche = 121. Respuesta: SPEEDING!! 101

3.1. Launcher

Finalmente, se creará un launcher para agrupar toda la configuración.

```
root@foxy:~/ros_prac_ws# vi super_launch.launch.py
```

Figura 3.4: Creación Launcher 1


```
root@foxy: ~/ros_prac_ws 203x46
from launch import LaunchDescription
from launch_ros.actions import Node

def generate_launch_description():
    ld = LaunchDescription()

    decisionsCar = Node(
        package = 'super_pkg',
        executable = 'decisions',
    )

    adasCar = Node(
        package = 'super_pkg',
        executable = 'ada',
    )

    cameraCar = Node(
        package = 'super_pkg',
        executable = 'camera',
    )

    velocityCar = Node(
        package = 'super_pkg',
        executable = 'velocity',
    )

    ld.add_action(decisionsCar)
    ld.add_action(adasCar)
    ld.add_action(cameraCar)
    ld.add_action(velocityCar)
    return ld
```

Figura 3.5: Creación Launcher 2