

# Sistema Operativo para Robots (ROS 2)

Verónica E. Arriola-Rios

Robótica móvil

14 de agosto de 2025

# Componentes en ROS2

- 1 Componentes en ROS2
- 2 Instalar
- 3 Componentes
- 4 Comandos y conceptos
- 5 URDF

# Temas

- 1 Componentes en ROS2
  - Componentes básicos

# Nodos

- ROS 2 define una gráfica con **nodos** que pueden comunicarse con otros nodos:
  - dentro del mismo proceso.
  - en procesos distintos.
  - en otra máquina.
- Por diseño un nodo debe ser una *unidad de cómputo*; cada nodo debe realizar una sola tarea lógica.
- Las **conexiones** se establecen mediante un proceso distribuido de **descubrimiento**.

<https://docs.ros.org/en/jazzy/Concepts/Basic/About-Nodes.html>

# Componentes básicos de ROS2

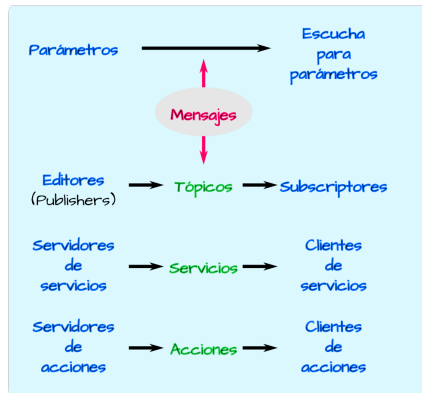
## Paquetes

Archivos de configuración

Lanzadores

Recursos

Nodos  
(Python o C++)



# Bibliotecas cliente

- Todas las bibliotecas cliente trabajan sobre la interfaz *ROS Client Library (RCL)* que implementa la **lógica** y **comportamiento** de ROS.
  - No es específica de ningún lenguaje.
  - Está implementada en C.
- Las dos bibliotecas cliente mantenidas por el equipo oficial son:
  - `rclcpp` En C++.
  - `rclpy` En Python. Al momento de enviar mensajes sus envolturas mandan llamar la versión en C, a menos que los publicadores y suscriptores se encuentren en el mismo proceso.
- Existen otras bibliotecas cliente mantenidas por la comunidad.

# Instalar

- 1 Componentes en ROS2
- 2 Instalar**
- 3 Componentes
- 4 Comandos y conceptos
- 5 URDF

# Verificar locale

```
$ locale # check for UTF-8

$ sudo apt update && sudo apt install locales
$ sudo locale-gen es_MX es_MX.UTF-8
$ sudo update-locale LC_ALL=es_MX.UTF-8 LANG=es_MX.UTF-8
$ export LANG=es_MX.UTF-8

$ locale # verify settings
```



# Instalación rápida

```
$ sudo apt install software-properties-common
$ sudo add-apt-repository universe
$ sudo apt update && sudo apt install curl -y
$ sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.
  ➔key -o /usr/share/keyrings/ros-archive-keyring.gpg
$ echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/
  ➔ros-archive-keyring.gpg] http://packages.ros.org/ros2/ubuntu $(. /etc/os
  ➔-release && echo $UBUNTU_CODENAME) main" | sudo tee /etc/apt/sources.
  ➔list.d/ros2.list > /dev/null
$ sudo apt update && sudo apt install ros-dev-tools
$ sudo apt update
$ sudo apt upgrade
$ sudo apt install ros-jazzy-desktop
$ sudo apt install ~nros-jazzy-rqt*
```

# Configuración

En `.bashrc`:

```
1 source /opt/ros/jazzy/setup.bash           # Activa por defecto el underlay
2
3 export ROS_DOMAIN_ID=5                     # Identificador para los dispositivos en la misma red.
4 export ROS_LOCALHOST_ONLY=1               # Usar cuando sólo se use una computadora
5
6
7 source /usr/share/colcon_cd/function/colcon_cd.sh
8 export _colcon_cd_root=/opt/ros/jazzy/
9 source /usr/share/colcon_argcomplete/hook/colcon-argcomplete.bash
10
11 export LC_NUMERIC="es_MX.UTF-8"           # Transmisión de msg con UTF
```

# Componentes

- 1 Componentes en ROS2
- 2 Instalar
- 3 Componentes**
- 4 Comandos y conceptos
- 5 URDF

# ROS vs ROS 2

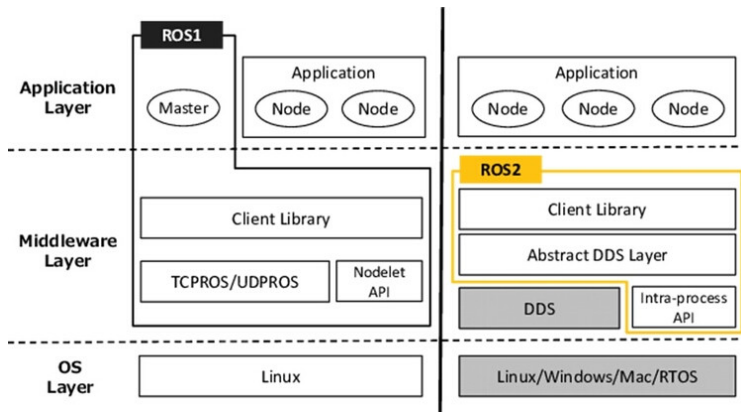


Figura: ROS 1 vs ROS 2 con la adición de la estandarización de la capa del protocolo de transporte en el estándar *Data Distribution Service* (DDS). (Robinson 2022)

# RMW, DDS, RTPS, QoS

ROS 2 depende de un nuevo sistema para transferir información, en búsqueda de hacer este proceso tan eficiente como lo requieren las aplicaciones industriales. Con esta decisión aparecen los siguientes acrónimos:

**RMW** *ROS Middleware Abstraction Interface*.

**DDS** *Data Distribution Service*. Hay diferentes vendedores y ROS ofrece soporte para varias alternativas, el incluido por defecto es *eProsima's Fast DDS*.

<https://docs.ros.org/en/humble/Installation/DDS-Implementations.html>

**RTPS** *Real-time Publish-Subscribe Protocol*, que es el protocolo de cable (*wire protocol*) del DDS, es decir, es el método para enviar datos de un punto a otro punto.

<https://www.omg.org/spec/DDS-RTSPS/2.3/Beta1/PDF>

QoS *Quality of Service* políticas para ajustar la comunicación entre nodos, depende del DDS subyacente.

<https://docs.ros.org/en/humble/Concepts/About-Quality-of-Service-Settings.html>

# RMW

La RMW define una interfaz de primitivas de software intermedio utilizadas por las APIs de ROS de nivel más alto. Consiste en los componentes siguientes:

- *Nodo* (*Node*)
- *Editor* (*Publisher*)
- *Subscripción* (*Subscription*)
- *Servidor de cliente* (*Client service*)
- *Servidor de servicio* (*Service server*)

También incluye funciones comunes, en combinación con *Tópicos* (*Topics*) y *Servicios* (*Services*); funciones para el manejo de componentes distribuidos, herramientas de introspección, entre otras funciones de utilidad.

<https://docs.ros2.org/bouncy/api/rmw/>

## Comandos y conceptos

- 1 Componentes en ROS2
- 2 Instalar
- 3 Componentes
- 4 Comandos y conceptos**
- 5 URDF



# Temas

## 4 Comandos y conceptos

- Conceptos
  - Interfaz de línea de comandos (*CLI*)
  - Desarrollo de bibliotecas

# Nodos, tópicos y servicios

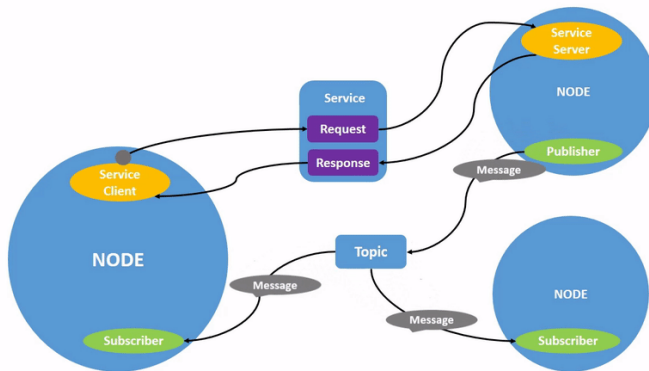


Figura: [https://docs.ros.org/en/humble/\\_images/Nodes-TopicandService.gif](https://docs.ros.org/en/humble/_images/Nodes-TopicandService.gif)

# Temas

- 4 Comandos y conceptos
  - Conceptos
  - Interfaz de línea de comandos (*CLI*)
  - Desarrollo de bibliotecas

# Comandos I

## Configuración del ambiente:

- Sólo el *underlay*:

```
$ source /opt/ros/<versión>/setup.bash
```

- *Underlay* y *overlay* (espacio de trabajo):

```
$ source /opt/ros/<versión>/setup.bash  
$ cd <workspace>  
$ . install/local_setup.bash
```

ó

```
$ cd <workspace>  
$ . install/setup.bash
```

# Comandos II

- [ROS2] ID de dominio para el grupo de agentes:

```
$ export ROS_DOMAIN_ID=<your_domain_id>
```

- Obtener información

```
$ ros2 -h
$ ros2 <command> -h
$ ros2 node list
$ ros2 node info
$ ros2 topic list
$ ros2 interface list [-m]
$ ros2 service list
$ ros2 action list
$ ros2 component types
```

# Paquetes

- Crear:

C++

```
$ ros2 pkg create --build-type ament_cmake <package_name>
```

Python

```
$ ros2 pkg create --build-type ament_python <package_name>
```

- Explorar:

```
$ colcon_cd <package>  
$ ros2 pkg executables <package>
```

# Nodos

- Ejecutar un nodo

```
$ ros2 run <package> <node_name>
```

# Remapeo

```
$ ros2 run <package> <node_name> --ros-args --remap <name>/<topic>:=<name>/<
  ↳topic>
```

Los argumentos para remapear se abrevian:

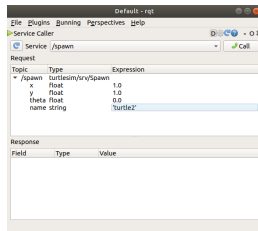
```
$ --ros-args -r <name>/<topic>:=<name>/<topic>
```



# RQT

- Obtener información

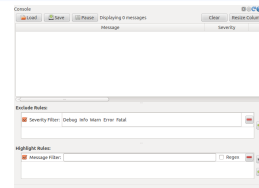
```
$ rqt
```



```
$ rqt_graph
```



```
$ ros2 run rqt_console rqt_console
```



# rosdep

- Instalar:

```
$ sudo apt install python3-rosdep2  
$ rosdep update
```

- Instalar dependencias de paquetes en src:

```
$ rosdep install -i --from-path src --rosdistro <versión> -y
```

# Temas

- 4 Comandos y conceptos
  - Conceptos
  - Interfaz de línea de comandos (*CLI*)
  - Desarrollo de bibliotecas

# Compilar el espacio de trabajo

- Instalar:

```
$ sudo apt install python3-colcon-common-extensions
```

- Compilar el espacio de trabajo:

```
$ colcon build
```

- Sólo el paquete deseado (y sus dependencias):

```
$ colcon build --packages-up-to <package>  
$ colcon build --packages-select <package>
```

- Para no recompilar python:

```
$ colcon build --symlink-install
```

- No hacer source del *overlay* en la misma terminal donde se compila.

# URDF

- 1 Componentes en ROS2
- 2 Instalar
- 3 Componentes
- 4 Comandos y conceptos
- 5 URDF**

# Prerrequisitos

```
$ sudo apt install ros-jazzy-urdf-tutorial
```

Este comando instala los paquetes:

```
ros-jazzy-joint-state-publisher  
ros-jazzy-joint-state-publisher-gui  
ros-jazzy-urdf-tutorial  
ros-jazzy-xacro
```

```
$ export LC_NUMERIC="es_MX.UTF-8"  
$ cd 'ros2 pkg prefix --share urdf_tutorial'
```

Es conveniente definir LC\_NUMERIC de modo permanente:

```
$ echo 'export LC_NUMERIC="es_MX.UTF-8"' >> ~/.bashrc
```

# URDF hola mundo

## Código: Mi primer robot

```
1 <?xml version="1.0"?>
2 <robot name="mi_primer_robot">
3   <link name="base_link">
4     <visual>
5       <geometry>
6         <cylinder length="0.6" radius="0.2"/>
7       </geometry>
8     </visual>
9   </link>
10 </robot>
```

<https://docs.ros.org/en/rolling/Tutorials/Intermediate/URDF/Building-a-Visual-Robot-Model-with-URDF-from-Scratch.html>


# Xacro

Instalar con:

```
sudo apt install ros-jazzy-xacro
```



# Referencias I

-  Robinson, Matt (ago. de 2022). *From academia to industry and beyond*. English. Southwest Research Institute. URL:  
<https://ifr.org/post/from-academia-to-industry-and-beyond>.