micro-ROS Market

micro-ROS is contributing to the faster growth of a competitive industry of small robots and robot components manufacturers, enabling European companies to rapidly deliver robotic products integrating highly resource-constrained devices.

Bringing all the benefits from the ROS technology into robotics product development

Accelerate the adoption of robotics in multiple domains

Increase development efficiency

Reduce costs

Service Robots_
Smart homes_
Transport Robots_
Agriculture Robots_
ROS2
Smart Cities_
Industrial Robots_
Healthcare Robots_

Why adopting it?

ROS interoperability
micro-ROS is joining forces with all the capabilities of ROS by bringing the ROS APIs to microcontrollers. Porting of advanced application-level software (e.g., for self-localization, obstacle avoidance) is simplified.

Reduce development costs and risks
By integrating microcontrollers into ROS, the developer can benefit from all ROS tools and advanced introspection, diagnostics, runtime configuration and monitoring features.

Faster time to market
micro-ROS is enabling rapid delivery of robotic products that integrate highly resource-constrained devices.

Wide community support
micro-ROS is enjoying a broad support from the ROS community. It is accessing a large base of users already working with the platform and bringing to Europe a key feature of this big robotic initiative.

Widening verticals adoption
micro-ROS redefines the boundaries of the ROS ecosystem by extending the range of applications (Industry 4.0, IoT, ...).

µROS

- *Global open-source project under permissive licenses*
- *Receives contributions from the world-wide ROS community*
- *Well-aligned with the on-going development of ROS 2*

micro-ros.github.io
Getting started • Tutorials • Concepts • Blog

github.com/micro-ROS
Source code • Issue tracking • Developers

discourse.ros.org and **answers.ros.org**
Tag your micro-ROS questions/discussions with #embedded

SPAIN	GERMANY	SWITZERLAND	POLAND
EPROSIMA The Middleware Experts	BOSCH Invented for life FIWARE FOUNDATION	ACUTRONIC ROBOTICS	PIAP SIEĆ BADAWCZA ŁUKASIEWICZ

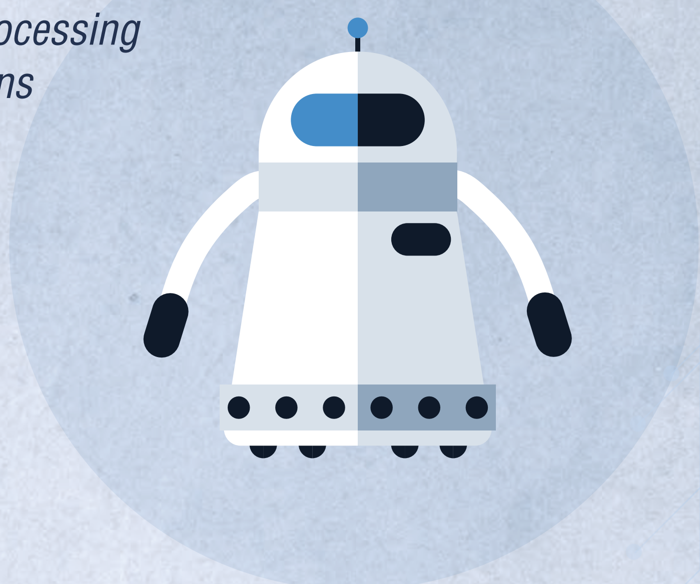
From 2018 to 2020, micro-ROS is backed by the EU research project OFERA (Open Framework for Embedded Robot Applications). The OFERA partners (see above) initiated the development of micro-ROS and maintain the core software packages.

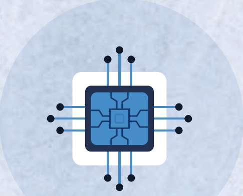
European Commission

This project has received funding from the European Union´s Horizon 2020 research and innovation programme under grant agreement No 780785.

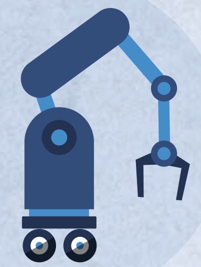
micro-ROS: a robotic framework

bridging the gap between resource constrained and larger processing units in robotic applications



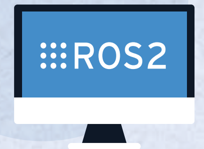


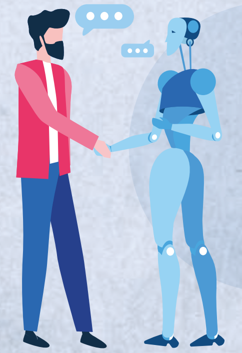
Microcontrollers no longer being inflexible black boxes



Brings ROS programming interface to resource-constrained devices

Seamless integration into ROS 2 based systems





Lowers market entry barriers

µROS

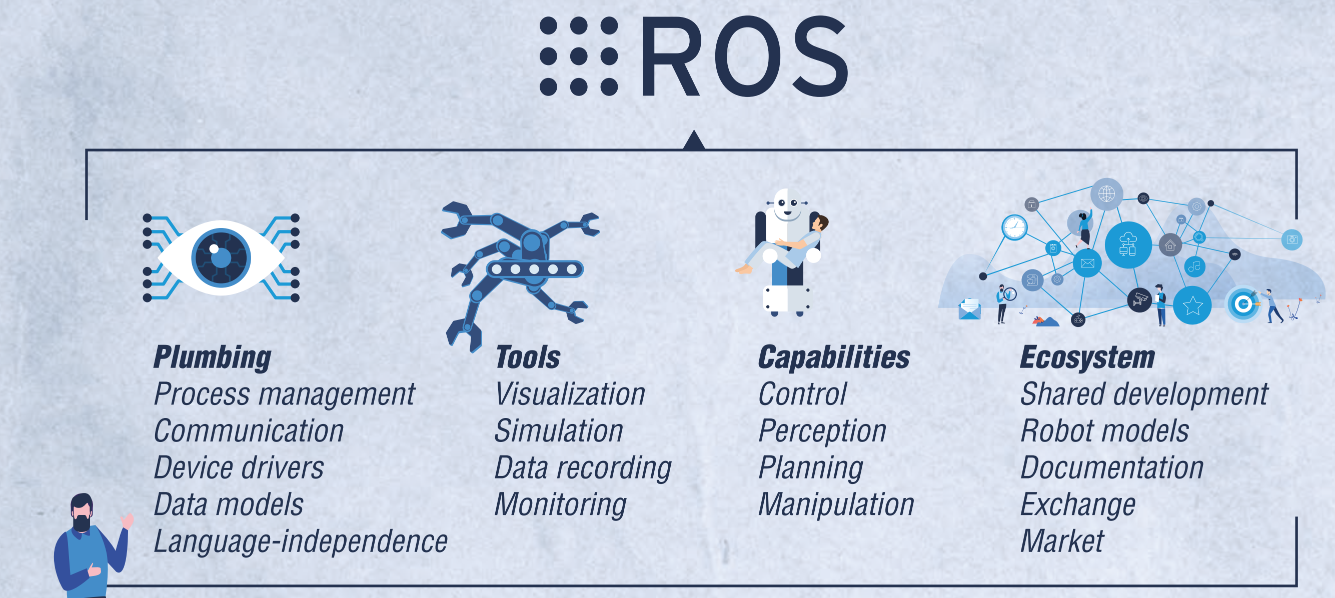
micro-ROS: at a glance

1. micro-ROS is the robotic framework that bridges the gap between resource constrained and larger processing units in robotic applications.

2. micro-ROS is compatible with the Robot Operating System (ROS 2), the de facto standard for robot application development.

3. micro-ROS empowers resource constrained devices and brings ROS 2 programming interfaces into them. This makes resource constrained devices first class participants of the ROS ecosystem, reducing the cost and size of robots.

4. micro-ROS enables the interoperability that distributed robotic systems demand to exploit the increasing overlap between robotics, embedded devices and IoT.



Plumbing
Process management
Communication
Device drivers
Data models
Language-independence

Tools
Visualization
Simulation
Data recording
Monitoring

Capabilities
Control
Perception
Planning
Manipulation

Ecosystem
Shared development
Robot models
Documentation
Exchange
Market

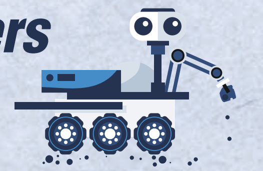
What is the Robot Operating System?

ROS is a middleware, development framework and toolbox for robotics software development. It has a huge, continuously growing and fast-paced community behind it and has probably become the largest open-source initiative undertaking in robotics. Its main contributions are:

- A service oriented architecture and communication mechanisms to support the assembly and orchestration of robotic software components as well as their interoperability with hardware drivers.
- A rich set of tools to develop, visualize, operate and maintain robot applications.
- Multi-language support: C++, Python, Java, C#, JavaScript, Ruby, ...
- Ready-to-use software components with functional capabilities for robot perception, control, planning, navigation and manipulation.

New to ROS? Start your journey at www.ros.org

Microcontrollers in robotics



Most robots are networks of microcontrollers and larger microprocessors. There are many reasons for the use of microcontrollers in robotics:

Hardware access
Microcontrollers provide rich input/output capabilities including GPIOs, AD converters, and PWM generators. They feature hardware support for communication buses such as CAN, UART, SPI, or I²C.

Hard, low-latency real-time
Real-time operating systems (RTOS) for microcontrollers allow context switching in less than 100 cycles - a magnitude less than with common desktop operating systems. Most RTOS require only few milliseconds to boot.

Power saving
Microcontrollers consume 10 to 100x less power than single-board computers for desktop operating systems. Many microcontrollers feature several low-power sleep modes.

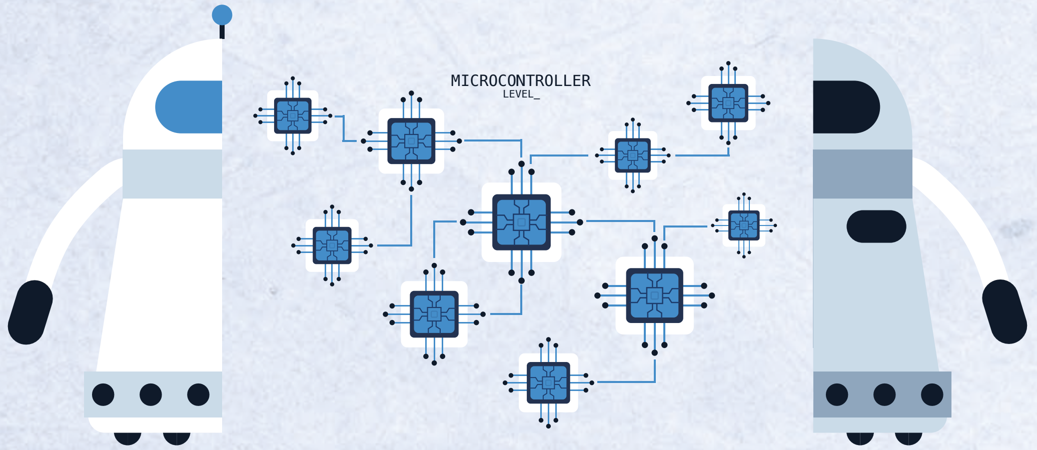
Safety
There exists a rich variety of microcontrollers for safety-critical applications. Similarly, a number of safety-certified RTOSs are available.

Integrated motor drive

Packaged sensor component / Safety module

Power-efficient infrastructure sensor

Ultra-light motion controller

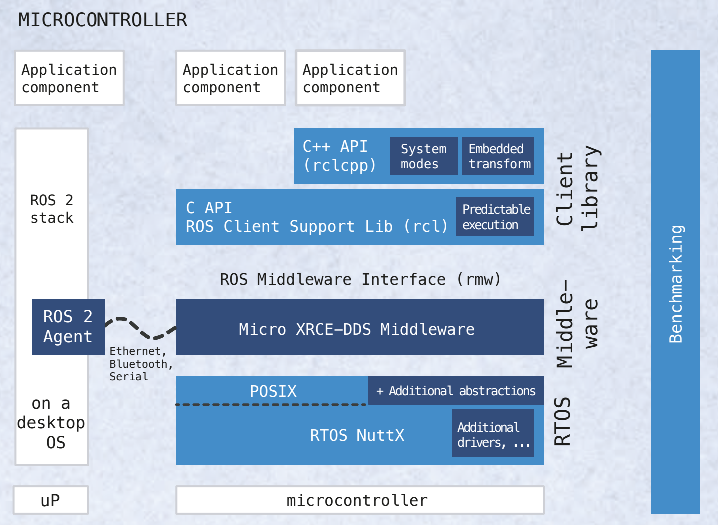


Microcontrollers and ROS

The ROS community has tried to support microcontrollers in the past and as part of the redesigned upcoming release ROS 2. These attempts unveiled various design choices in ROS and ROS 2 that render such porting impossible. These choices include the use of the DDS middleware in ROS 2, which is not intended for highly resource-constrained devices, the non-consideration of power efficiency requirements and the lack of advanced real-time scheduling capabilities, amongst others.

This is, where micro-ROS comes into play ...

micro-ROS features and architecture

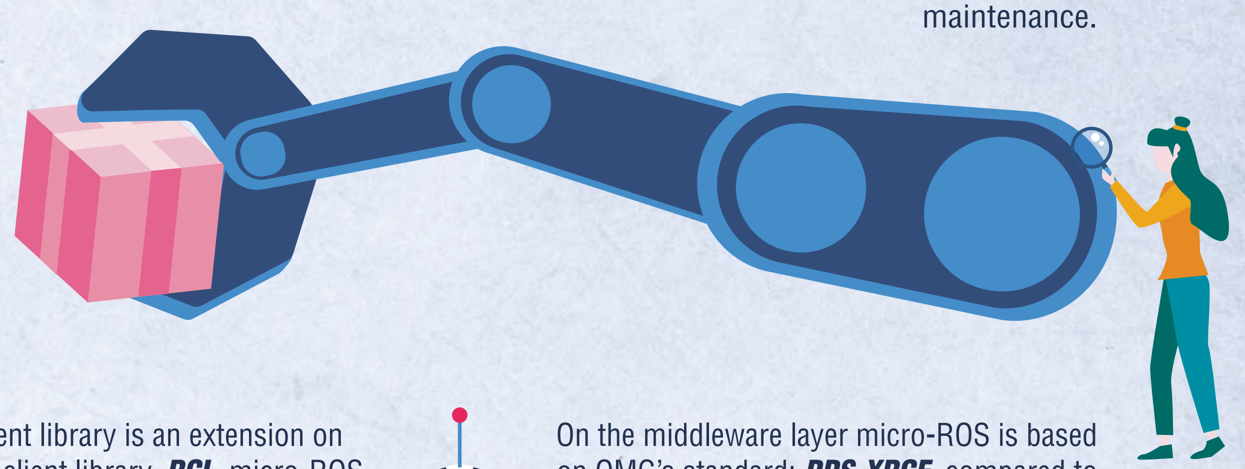


The most important differences of the ***micro-ROS*** stack compared to standard ROS 2 are on the lower layers:

micro-ROS uses a real-time operating system (***RTOS***) and not a desktop operating system like Linux. Currently, micro-ROS supports NuttX and FreeRTOS.

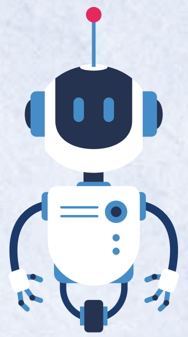
DDS-XRCE instead of standard DDS.

Apart of those layers, micro-ROS uses ROS 2 core stack layers, granting compatibility and ensuring long-term maintenance.

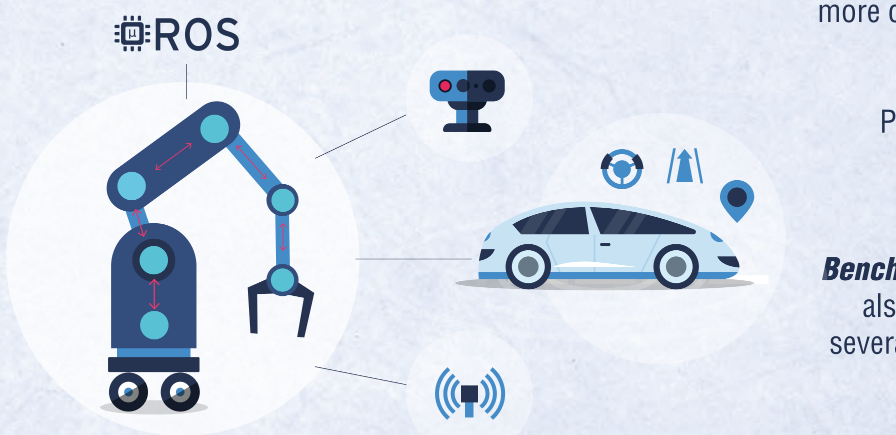


micro-ROS client library is an extension on existing ROS 2 client library, ***RCL***. micro-ROS will add dedicated modules to the existing approach adding concepts appealing to microcontrollers such as:

- A new predictable execution model.
- A model-based approach for runtime system configuration:



On the middleware layer micro-ROS is based on OMG's standard: ***DDS-XRCE***, compared to ROS 2 DDS base layer, micro-ROS underlying middleware layer is:
Designed and created to bridge DDS and embedded devices
Client-Server architecture.
Focused on low resource consumption.



Micro-ROS is focused on embedded devices and more concrete, microcontrollers used with an RTOS. The use of an ***RTOS***:
Provide HW abstraction.
Provide already known APIs, like POSIX.
Use of tools to configure-build-deploy firmwares.

Benchmarking of embedded environments is also a point of interest for micro-ROS and several benchmarking tools will be provided.