ROS 2 on Embedded Devices: overview and alternatives

www.eProsima.com

eProsima products

eProsima Services

- Architecture Study
- Feature acceleration
- Customized solutions
- Technical Support

eProsima license

- SW privative license

Fast DDS

- Publish-Subscribe DDS middleware for real-time distributed systems
- Adopted by ROS 2 **(leader MW 2017-2025)**

XRCE DDS

- Wire protocol for eXtremely **R**esource **C**onstrained **E**nvironments (MCUs)
- Adopted by micro-ROS

Monthly Users > 50k

New!

Safe DDS

- DDS middleware ISO 26262
- Automotive compliant middleware

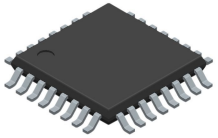
Introduction

1. Robots are made of **sensors and actuators**
2. Sensors and actuators are controlled by **low level interfaces**
3. Sensors and actuators feed **real time control loops**
4. **Microcontrollers** provide low level interfaces and hard real time capabilities

But ...

1. Robotic engineers use **ROS 2**
2. **ROS 2** runs on general purpose CPUs
3. General purpose CPUs **does not have** low level interfaces nor hard real time capabilities

ROS 2 Embedded



?

ROS 2 Embedded

Why

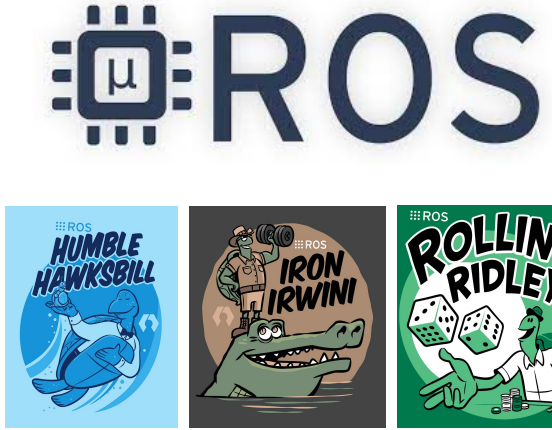
- **Ubiquity**: ROS 2 nodes close to the metal
- **Simplicity**: Direct control over low level sensors / actuators
- **Abstraction**: Reuse of available packages and common interfaces

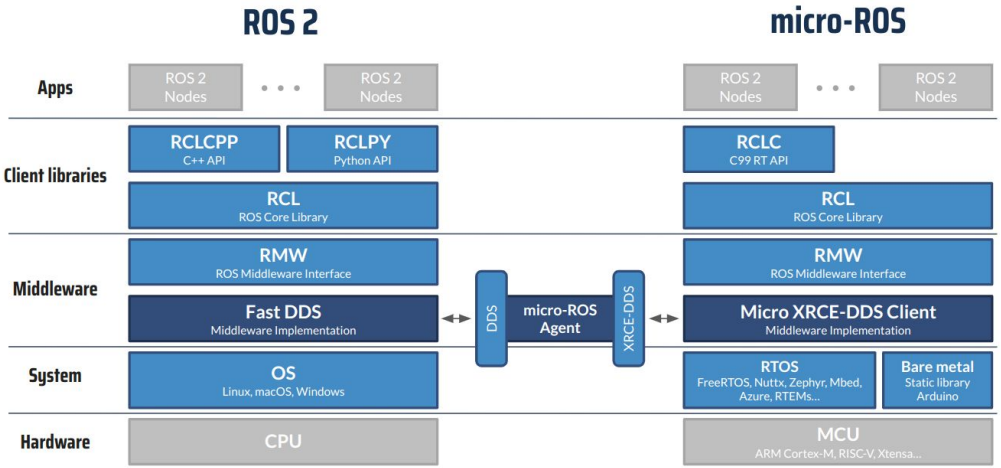
Problems

- **Heterogeneity**: Vendor-specific, lack of standardization, multiple IDE/toolchains, etc.
- **Low resources:** Has low resources (< 512 kB RAM)
- **Lack of portability:** May have no C++ support, STD, libc, etc.
- **Low bandwidth:** May have low bandwidth communication (UART, CAN, etc)

micro-ROS

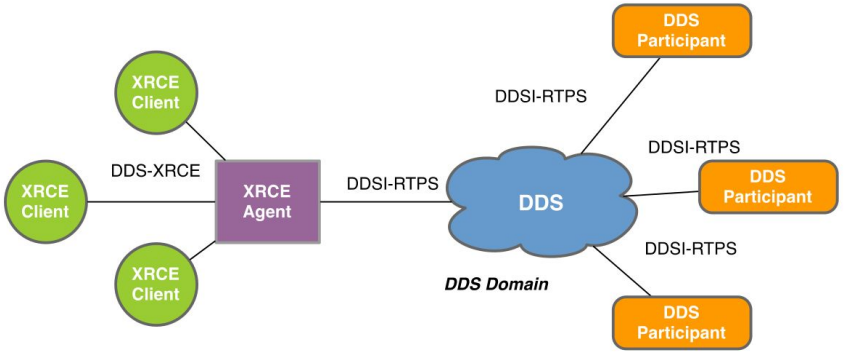
- **Port** of the ROS 2 stack to embedded devices
- **Tools for integration** on most embedded platforms, vendors, RTOS and frameworks
- **Optimized and compatible** ROS 2 packages: RCLC, micro_ros_utils, etc.
- **Supported:** pub/sub, services, actions, parameters, etc.



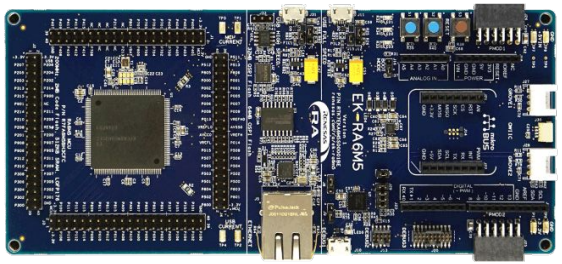


eProsima Micro XRCE-DDS

- **Embedded library:** deeply embedded library in C99 with extreme low resource usage
- **Micro XRCE-DDS Agent:** bridge between DDS and embedded worlds
- **Brokered architecture:** a Micro XRCE-DDS Client communicates with an Agent
- **DDS Compatible:** exposes most of the DDS features with a embedded friendly API



micro-ROS use cases



CAPRA ROBOTICS

GARDIN

RENESAS

New generation

Embedded silicon industry has evolved in last 5 years:

- **Unification**: ARM Cortex-M/A & RISC-V are *de facto* standards → GCC based compilers
- **Resources**: Huge amounts of RAM memory. From < 512 kB to ~ 2 MB in MCUs.
- **Peripherals**: 10/100 Ethernet interfaces, advanced networking libraries.

Next generation ROS 2 / DDS middlewares are possible:

- **Brokerless**: MCUs are first class citizens in ROS 2 / DDS dataspaces.
- **Advanced features:** Complex behaviours and QoS for communication channels
- **Complex ROS 2 packages**: More computational power → More complex architectures

eProsima Safe DDS

Safe DDS

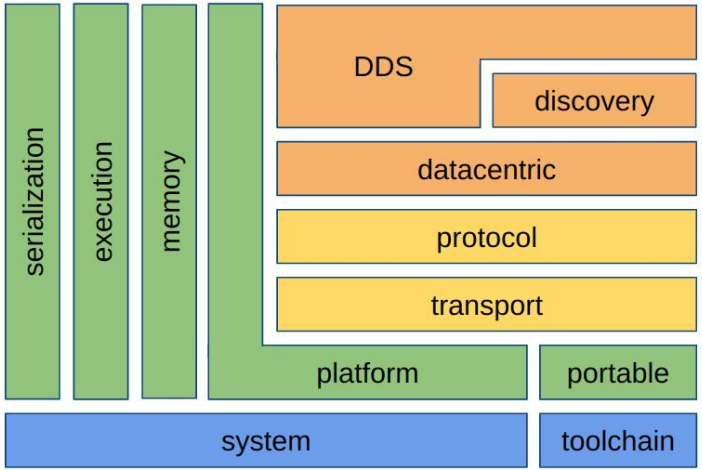
The safety certified (ISO 26262) DDS compliant middleware library

It targets hard real-time safety critical systems based on mid-low range MCUs and CPUs, including automotive electronics control units (ECUs).
It enables developers to create communication systems certifiable for ISO 26262 (ASILD).

eProsima Safe DDS

- **Embedded library:** embedded library in C++14 with low resource usage
- **ROS 2 / DDS compatible:** fully compatible DDS APIs and compatible with Fast DDS
- **Performance:** same latency / throughput as Fast DDS
- **ISO 26262:** Functional Safety certification for automotive

Safe DDS



Conclusions

	Safe DDS	micro-ROS / XRCE-DDS
Architecture	DDS Compatible	Brokered architecture *Client - Agent - DDS*
OS	General OS / RTOS / Bare Metal *Linux / FreeRTOS / QNX / …*	RTOS and Bare Metal *Azure RTOS / FreeRTOS / Zephyr /…*
Interfaces	Network transport agnostic *Eth / WiFi*	Transport agnostic *UART / Eth / WiFi / USB / CAN-FD*
Real time capabilities	Hard real time *Non blocking API & Zero heap memory*	Hard real time *Non blocking API & Zero heap memory*
Memory Usage	Very low memory usage *< 64 kB per DDS Participant + pub + sub*	Extremely low memory usage *< 32 kB per DDS Participant + pub + sub*
Target	Mid-low range MCU / CPU / ECU	Low range MCU

Open documentation

Visit online documentation for knowing more about eProsima products:

safe-dds.docs.eprosima.com

micro-xrce-dds.docs.eprosima.com

micro.ros.org

eProsima

The Middleware Experts

www.eProsima.com

Linkedin.com/company/eProsima

Twitter.com/EProsima