

ROS

New client library and middleware features

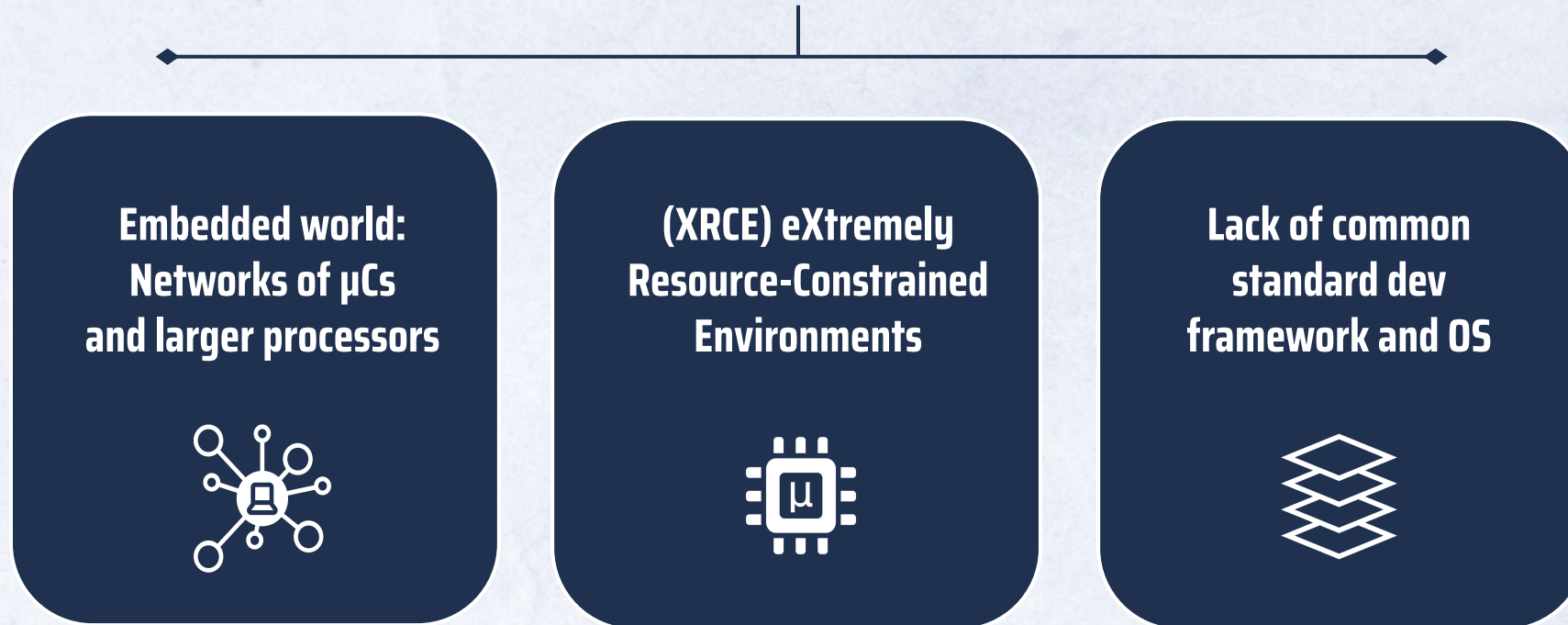


Ralph Lange (Robert Bosch GmbH, Corporate Research)

Maria Merlan (eProsima)

October 2021

Overview and Mission



micro-ROS bridges this gap for ROS 2

FULL PORTABILITY

Any RTOS and Bare metal Library Generator!

Any low-mid range MCU!

Typical features:

~ 150 KB of flash memory

> 25 KB of RAM memory

General purpose input/output pins

Peripherals: GPIO, USB, Ethernet, SPI, UART, I2C, CAN, etc



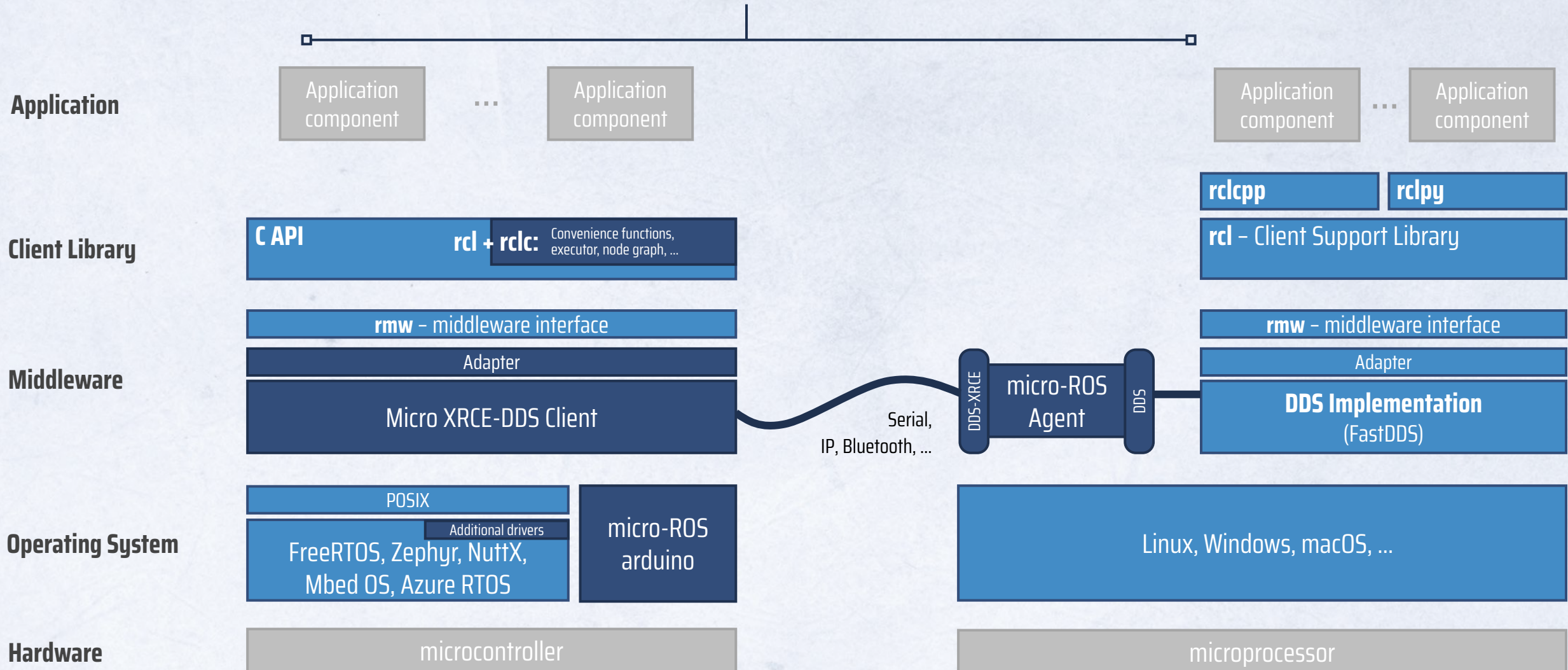
REFERENCE HW

- **Renesas RA6M5**
- **Arduino**
- **Raspberry Pi Pico**
- **Arduino Nano RP2040 Connect** 1st Arduino with Raspberry Pi silicon
- **ESP-IDF v4.3 & ESP32-S2/C3**
- **Teensy 3.2 / 3.5 / 4.1 / 4.2**
- **OpenCR support**
- **STM32CubeMX & STM32CubeIDE**
- **Crazyflie 2.1 drone, ...**

REFERENCE RTOS

- **Mbed RTOS 6.8 / 6.9 / 6.10**
- **FreeRTOS**
- **NuttX 10.0 / 10.1**
- **Zephyr RTOS 2.4 / 2.5**
- **Azure RTOS ThreadX**

Architecture



Build Systems



For your ROS tooling:

```
$ git clone github.com/micro-ROS/micro_ros_setup.git
$ colcon build

$ ros2 run micro_ros_setup
  ↳ create_firmware_ws.sh [RTOS] [MCU]
  ↳ configure_firmware.sh [MY_APP] -t [TRANSPORT]
  ↳ build_firmware.sh
  ↳ flash_firmware.sh

$ ros2 run micro_ros_setup
  ↳ create_agent_ws.sh
  ↳ build_agent.sh
  ↳ micro_ros_agent [TRANSPORT]
```

For your favorite tool:

- **Renesas e2 studio**
- **STM32CubeMX/IDE**
- **RasPi Pico SDK**
- **Arduino**
- **ESP32 IDF**
- **Zephyr**



NEW FEATURES

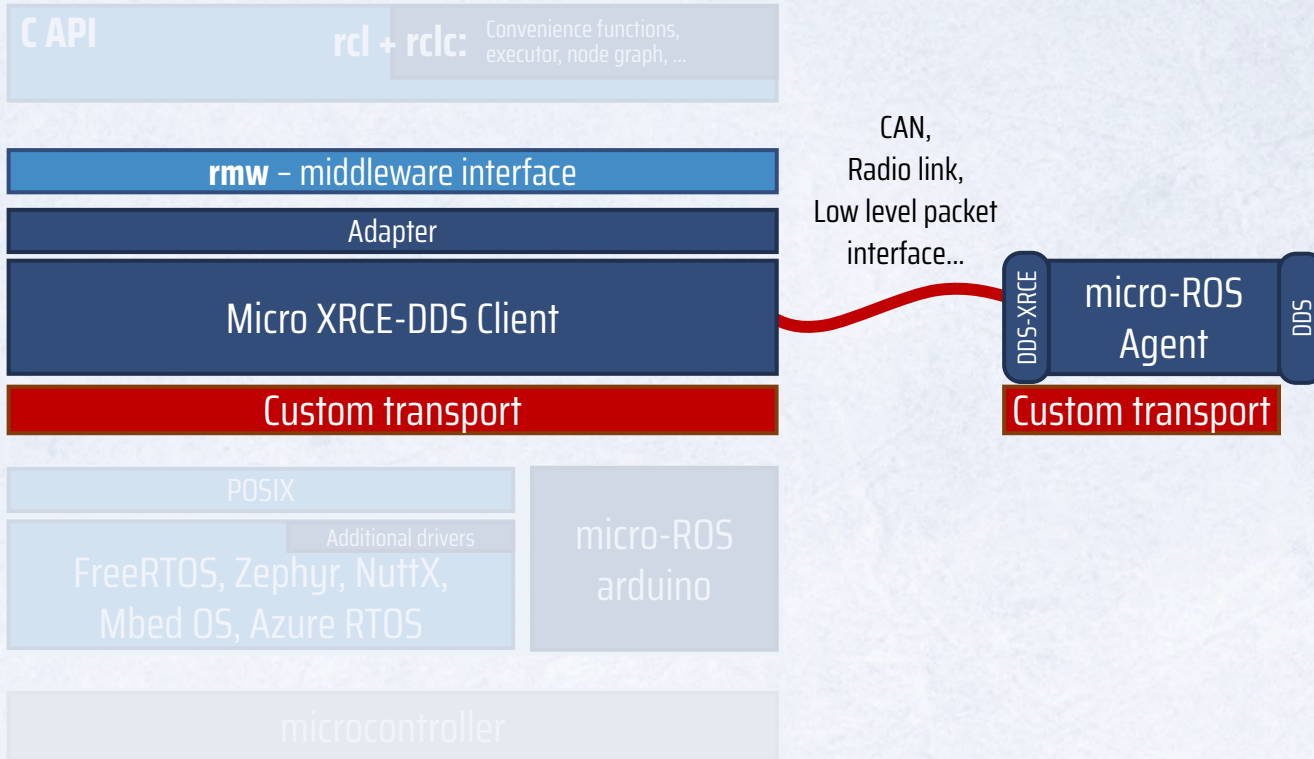
New API for custom transports **01**

Executor worker concept **02**

Diagnostics **03**

More new features **04**

Custom Transports

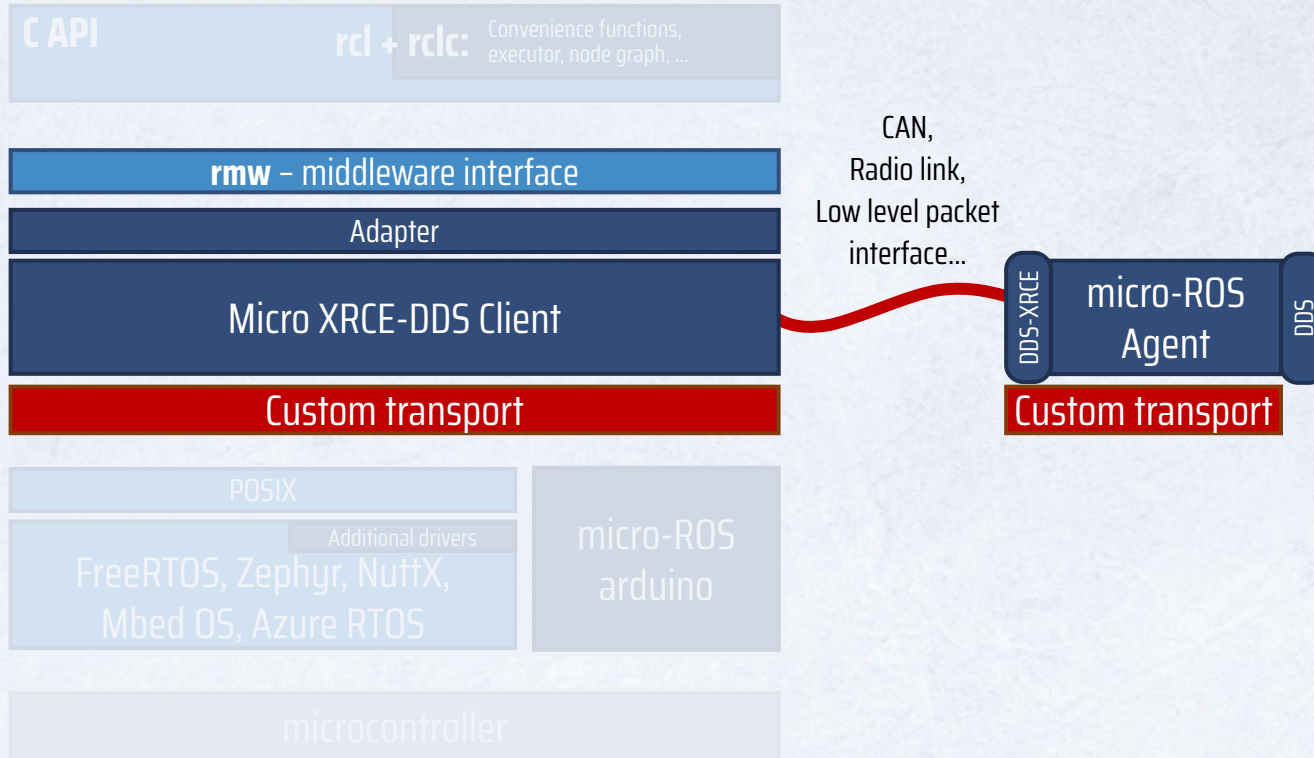


API for micro-ROS Agent and Client based on just four callbacks:

```
rmw_uros_set_custom_transport(  
    true, // Framing enabled here.  
    (void *) &args,  
    my_custom_transport_open,  
    my_custom_transport_close,  
    my_custom_transport_write,  
    my_custom_transport_read  
);
```

- Packet or stream mode based on HDLC
- Custom arguments

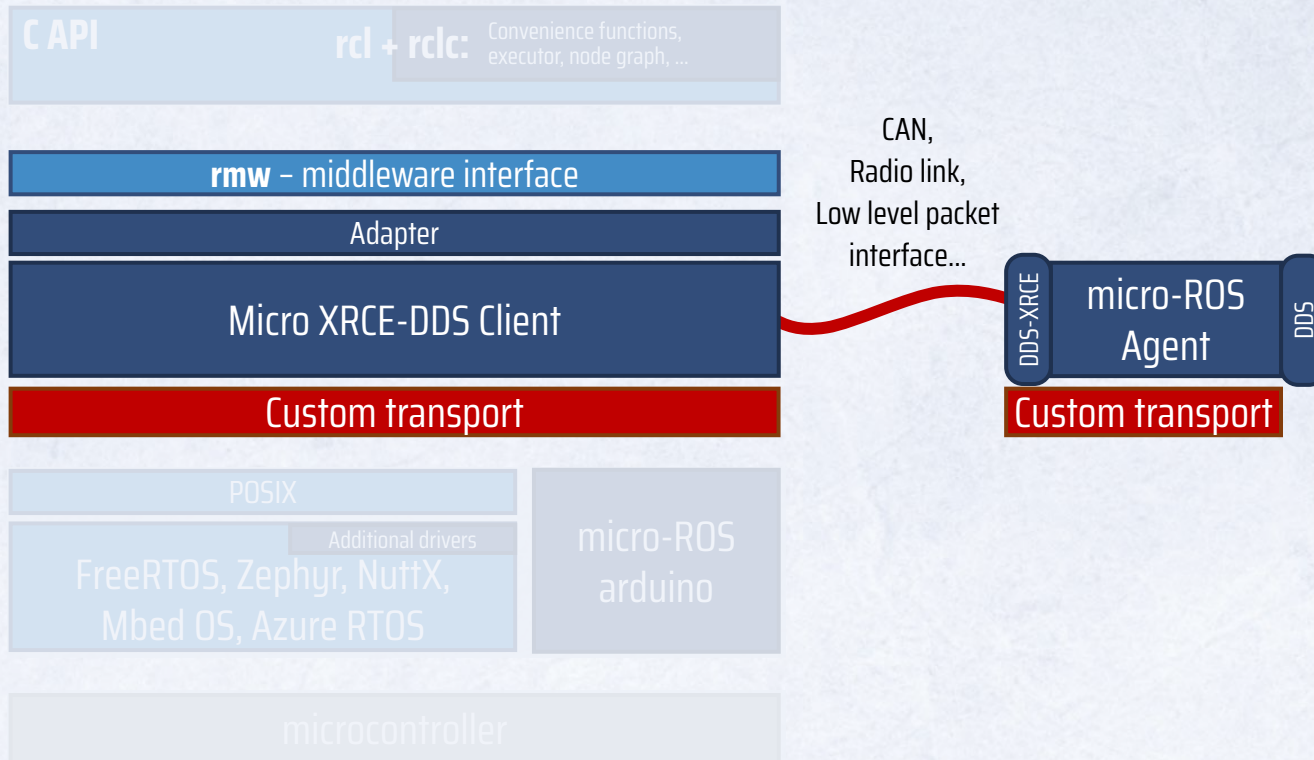
Custom Transports



Micro XRCE-DDS cares for error handling:

- Custom transport does not need to provide (1) reliability, i.e., messages may be dropped (2) ordering, i.e., messages may arrive out of order (3) notification of dropped messages
- Details in Section 11.1 of DDS-XRCE standard at www.omg.org/spec/DDS-XRCE/

Custom Transports



Example in Crazyflie demo

- github.com/micro-ROS/freertos_apps/microros_crazyflie21_extensions/src/microros_transports.c

Documentation at micro.ros.org

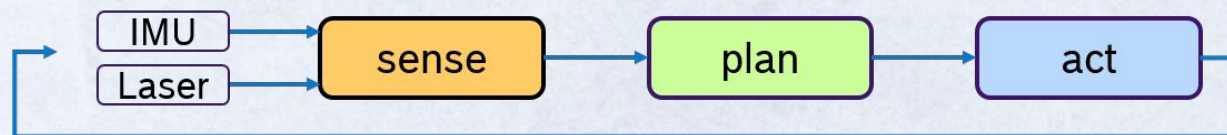
- Tutorials → Advanced → Creating custom micro-ROS transports

Executor Worker Concept

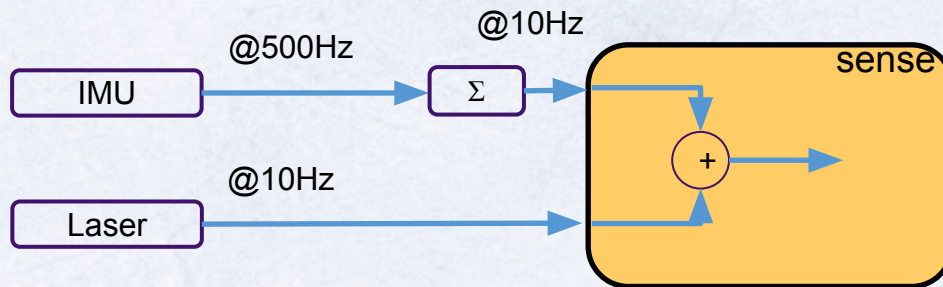


Extended API for typical patterns

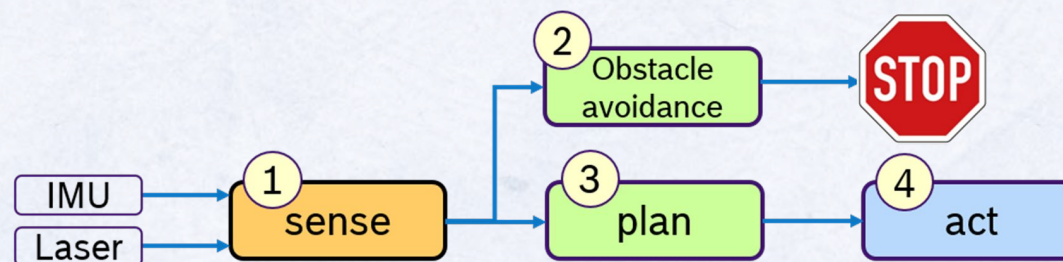
- Control loops



- Data fusion



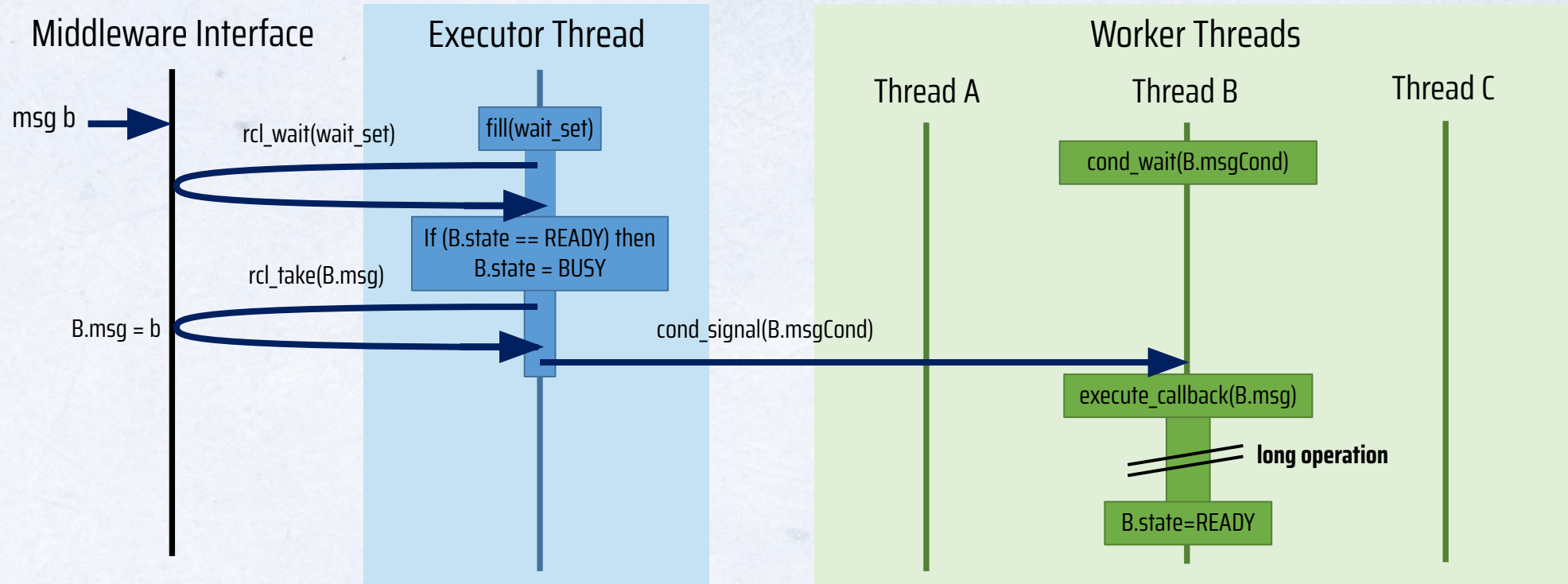
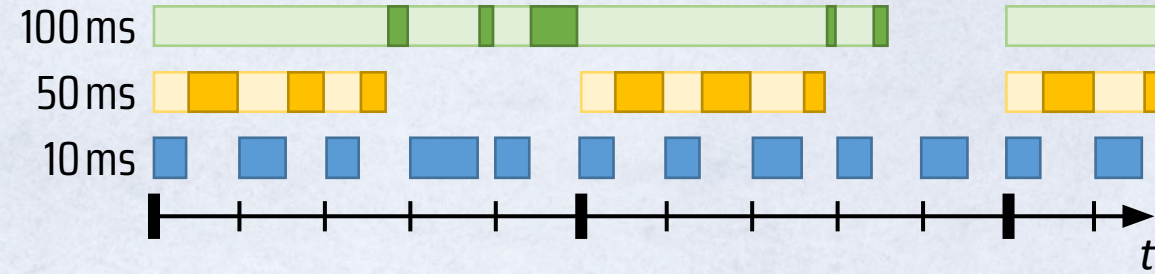
- Prioritized paths



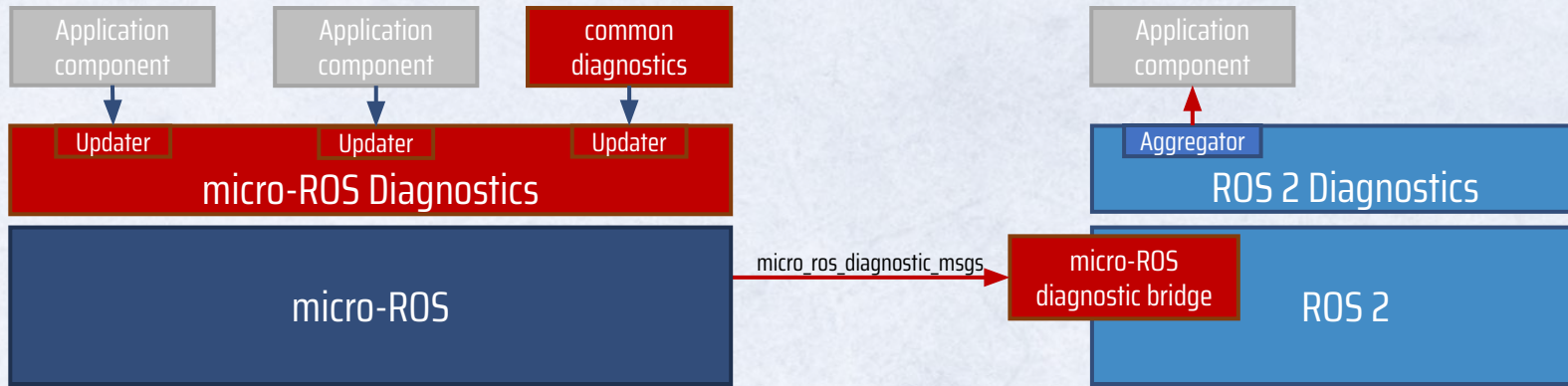
Executor Worker Concept



Flexible concept for use with rate-monotonic or reservation-based scheduling



Diagnostics



Important features

- Optimized message types
- Updater implementation for rcl+rcl

Source code at github.com/micro-ROS/micro_ros_diagnostics

- Contributions welcome. Reach out to us!

More New Features



Hardware support: **RasPi Pico** • **Teensy** • **Renesas EK RA6M5**

RTOS: **bare metal (Arduino)** • **Mbed OS** • **Azure RTOS ThreadX**

Middleware: **static memory pools** • **thread-safety** • **embedded RTPS RMW beta support**

Client library: **parameters** • **node lifecycle** • **ROS graph** • **services**

micro.ros.org

[micro-ROS at GitHub](#)

[Features](#)

[SW and HW support](#)

[Advanced tutorials with examples](#)

[Articles](#)

[Demo videos](#)

+ Join the Embedded WG!

The screenshot shows the homepage of the micro-ROS website. The browser address bar displays 'https://micro.ros.org'. The navigation menu includes 'Overview', 'Concepts', 'Tutorials', 'API', and 'Blog'. The main heading is 'micro-ROS puts ROS 2 onto microcontrollers'. Below this, there are three columns of content: 'Mission', 'Key Features', and 'Getting Started'. The 'Mission' section describes bridging the gap between resource-constrained microcontrollers and larger processors. The 'Key Features' section lists several benefits, including microcontroller-optimized APIs, seamless integration with ROS 2, and multi-RTOS support. The 'Getting Started' section encourages users to explore tutorials and demos. At the bottom, there is an 'Architecture' section with a diagram showing the stack from the microcontroller up to application components. The diagram includes layers for ROS 2 stack (ROS API, ROS Middleware Interface, Micro XRCE-DDS Client), ROS 2 Agent, and RTOS (Zephyr, FreeRTOS, NuttX, micro-ROS arduino). A 'Benchmarking' bar is also visible on the right side of the architecture diagram.

micro-ROS | ROS 2 for microco X +

https://micro.ros.org

micro-ROS Overview Concepts Tutorials API Blog

Search via Lunr.js

micro-ROS

micro-ROS puts ROS 2 onto microcontrollers

Mission

Bridging the gap between resource-constrained microcontrollers and larger processors in robotic applications that are based on the Robot Operating System.

Key Features

- ✓ Microcontroller-optimized client API supporting all major ROS concepts
- ✓ Seamless integration with ROS 2
- ✓ Extremely resource-constrained but flexible middleware
- ✓ Multi-RTOS support with generic build system
- ✓ Permissive license
- ✓ Vibrant community and ecosystem
- ✓ Long-term maintainability and interoperability
- ✓ Much more...

Getting Started

Our [tutorials](#) and [demos](#) give you a quick start with micro-ROS. The [basic tutorials](#) can even be completed without a microcontroller.

Architecture

The architecture of the [micro-ROS stack](#) follows the ROS 2 architecture. Dark blue components are developed specifically for micro-ROS. Light blue components are taken from the standard ROS 2 stack.

Why Microcontrollers?

Microcontrollers are used in almost every robotic product. Typical reasons are:

- Hardware access
- Hard, low-latency real-time
- Power saving

Another important reason is safety, but note that micro-ROS is not developed according to any safety standard. Check our complete [supported hardware list](#).

Commercial support

eProsima provides [commercial support](#) to boost micro-ROS projects:

- ✓ Port micro-ROS to your platform (HW, RTOS, transport)
- ✓ Efficient & reliable communication layer between μ C and DDS Data Space (ROS 2)
- ✓ Customized features development
- ✓ Architectural studies

Source Code



The OFERA project is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement No 780785