D3.10

Interoperability Report Y1

Grant agreement no.	780785
Project acronym	OFERA (micro-ROS)
Project full title	Open Framework for Embedded Robot Applications
Deliverable number	D3.10
Deliverable name	Interoperability Report
Date	December 2018
Dissemination level	Public
Workpackage and task	3.3
Author	Borja Outerelo Gamarra (eProsima)
Contributors	Ingo Lütkebohle (Bosch), Iñigo Muguruza (Acutronic)
Keywords	Micro-ROS
Abstract	This document provides a report regarding the interoperability of micro-ROS with existing ROS2, FIWARE and H-ROS.

Contents

1 Acronyms

Acronym	Explanation
CDR	Common Data Representation
DDS	Data Distribution Service
DDS-XRCE	DDS For Extremely Resource Constrained Environments
GA	Grant Agreement
OFERA	Open Framework for Embedded Robotic Applications
ROS	The Robot Operating System
RTPS	Real Time Publish Subscribe
WP	Work package

2 Executive summary

This report, **D3.10 Interoperability Report - Y1**, provides an overview of the status of the interoperability of micro-ROS, on the first year of the project.

OFERA consortium focused on three micro-ROS interoperability points during the reporting period:

- Interoperability with ROS 2 and ROS systems.
- Interoperability with FIWARE.
- Hardware interoperability.

The first interoperability point addressed by OFERA consortium was ROS 2 and ROS interoperability. micro-ROS interoperability with ROS2 has been achieved at a great extent thanks to the defined architecture.

In micro-ROS architecture definition, the interoperability with current ROS 2 middleware protocol, DDS/RTPS, was taken into account. The middleware protocol that consortium chose for micro-ROS, DDS-XRCE, provides this DDS interoperability.

OMG creates DDS-XRCE protocol with this interoperability as one of its primary objectives. DDS-XRCE grants interoperability between its clients and existing DDS global data spaces. DDS-XRCE achieves this interoperability thanks to the use of a DDS-XRCE Agent which acts as a bridge between DDS-XRCE networks and a DDS global data space.

For all the previous, using DDS-XRCE and granting DDS interoperability allows the interoperability between micro-ROS and ROS 2 systems, in the current status of ROS 2 implementations. One benefit of interoperating with ROS 2 is that micro-ROS can interoperate with ROS 1 using the existing ROS 2-ROS 1 Bridge. WP3 work and more precisely task 3.3: ROS bridging & Interoperability achieved this ROS2 and ROS interoperability.

As a second interoperability point, micro-ROS targeted to interoperate with FIWARE. This interoperation has been achieved during the reporting period thanks to adaptation and usage of eProsimas FIWARE integration services, in this concrete case, using FIROS 2. In this first year, the consortium achieved interoperability between FIWARE and micro-ROS getting used of the DDS network from the micro-ROS Agent (DDS-XRCE Agent) side. micro-ROS - FIWARE Interoperation works the same way standard ROS

2 interoperates with FIWARE, using FIROS 2. For each ROS 2 interface that the user wants to integrate with FIWARE, a ROS 2 FIROS 2 node needs to be created with the dedicated integration services working for that interface. These integration services make use of micro-ROS rmw layer to be able to serialise-deserialise the ROS2 interfaces types. Using rmw implementation simplify the required tooling and development from the user side. WP3 involving task 3.4: Fiware interoperability achieved this FIWARE interoperability.

The third micro-ROS interoperability is at a hardware level, thanks to the use of HRIM. HRIM will provide micro-ROS with the capability to interoperate microcontrollers using it with other robot parts as they share common interfaces and definitions. In the reporting period designing tasks has been started as well as the development of generation tools by the WP 2 in task 2.3: Hardware Bridge.

3 Introduction

3.1 Purpose of document

The purpose of this document is to provide an overview of the interoperability of the Horizon 2020 Innovation Action shortly named OFERA - Grant Agreement Number 780785 o the European Commission. It covers the first period of the project, from 1st January 2018 to 31st December 2018 (twelve months, from M1 to M12). This interoperability report targets three different interoperations: ROS 2 (ROS), FIWARE and hardware. This report follows the following structure:

- Part 1 is the executive summary with a quick review of the advances done.
- Part 2 is the present introduction.
- Part 3 is the report itself. It ...
  - Explains ROS 2 interoperability
  - Explains FIWARE interoperability
  - Introduces hardware interoperability using HRIM.
- Part 5 contains conclusions.

This document is the first version of a series of periodic reports. OFERA has two more releases scheduled for the project as shown below:

- Interoperability Report Y2 - 31.12.2019
- Interoperability Report Y3 - 30.06.2020

4 Interoperability report

In this section, the report defines and explains the interoperability with ROS 2, FIWARE and HRIM regarding their status at the end of this first year of OFERA project.

4.1 ROS 2 Interoperability

4.1.1 Introduction

micro-ROS interoperability with ROS2 was one of the technical objectives of micro-ROS as mentioned in the OFERA GA. Also, it could be seen as the must-have interoperability as it builds one of the foundations of micro-ROS, bringing ROS 2 to embedded devices.

ROS 2 interoperability and portability has been a critical piece on the design and development on the current micro-ROS version. The current version of micro-ROS pursues to resemble as much as possible the usage of ROS 2. For that reason, micro-ROS reuses the same concepts of ROS 2, and even It reuses a significant amount of the tooling surrounding ROS 2.

This reuse of existing concepts and tools allows micro-ROS to achieve a high level of integration with current ROS 2 systems. In this first version, apart of taking much of its architecture concepts, micro-ROS reuses ROS 2 build system and types definitions. These two concrete cases ease in great extent the interoperability between micro-ROS and ROS 2:

a. The reuse of build system allows having the same process for generating interfaces and their associated and required libraries.
b. The reuse of existing types definitions ensures that the generation of code using them as the base, uses the same source as in ROS 2, avoiding types mismatching issues.

Apart from that high-level integrations and interoperation, micro-ROS propose to use standard middleware protocols, natively interoperable with the one used in ROS 2, DDS.

The protocol chosen for micro-ROS is the OMG DDS-XRCE as it is a perfect match with DDS, both come from the same standardisation body as well as DDS-XRCE was designed and created with DDS interoperation as a foundational requirement. Being interoperable with OMG standard DDS means that micro-ROS theoretically could be interoperable with any DDS vendor. Even, interoperation between DDS-XRCE client and agent is possible between different vendors as DDS-XRCE standardise that communication between Agent and Client.

Interoperation of DDS different vendors is out of the scope of this project, but OMG standardisation was designed with that in mind and provides multiple levels of interoperation, starting in the DDS interface, through the interoperable wire protocol used, RTPS, to the data representation, CDR. DDS vendors and users continuously test this DDS vendor interoperability in different ways, two of them are worthy to mention here:

1. OMG official GitHub repository [1] with interoperability tests. DDS vendors manually run these tests.
2. ROS 2 official CI site with automated tests [2], using different rmw implementations
    - They run those tests on a CI tool, Jenkins. Sample results of an execution: Jenkins test run

The use of such standardised and DDS interoperable protocols, as this report explains later eases the interoperability between ROS 2 and micro-ROS at the middleware layer.

4.1.2 micro-ROS current status

As introduced above, OFERA achieved micro-ROS and ROS 2 interoperability from the early stages of the project. This interoperability was eased thanks to the use of interoperable middleware protocols, DDS and DDS-XRCE on ROS 2 and micro-ROS side respectively.

As already mentioned, current ROS 2 implementation uses OMG DDS as underlying middleware protocol. Using DDS middleware allows any DDS implementation to be interoperable, almost out of the box, with ROS 2. Consortium partner eProsima provides one of the DDS implementations, Fast RTPS, and even it has been chosen to be the ROS 2 default middleware implementation.

The eProsima situation as the ROS 2 default middleware providers and its participation as an OMG member on the standardisation of these protocols, provides micro-ROS with a good advantage position. eProsima, apart from DDS implementation, also develops and maintains the current micro-ROS middleware implementation, Micro XRCE-DDS which implements DDS-XRCE. Being implementer of both ends in the interoperability, Fast RTPS and Micro XRCE-DDS, ensures that both products will keep interoperable from early development stages. Users and eProsima proved this interoperability in different ways:

a. eProsima uses CI with integration tests [3]. These tests use Micro XRCE-DDS Agents and Clients along with Fast RTPS testing communication between all the pieces.
b. Third parties: Renesas and Robotis are actively using micro XRCE-DDS. They mainly use Micro XRCE-DDS to communicate with ROS 2 (DDS).
    a. Renesas enables their platform GR-ROSE to be a ROS 2 participant using Micro XRCE-DDS [4].
    b. Robotis uses Micro XRCE-DDS as a bridge to communicate their XEL Network with ROS 2 [5].

Even though, this middleware interoperability is the base to a ROS2 - micro-ROS interoperability is not the only interoperability needed for achieving full integration.

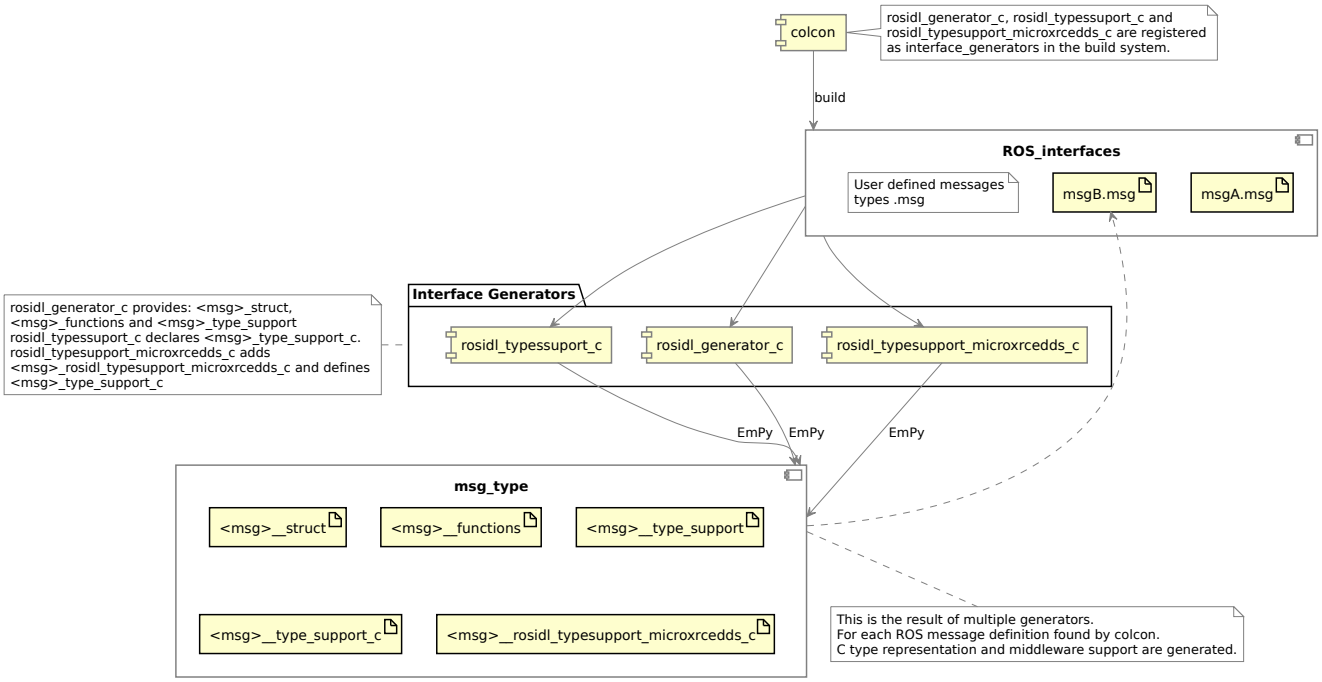
One of the critical aspects of ROS 2 is the standardisation of their interfaces, simplifying, the interoperability between applications. ROS 2 interface plays a noticeable role in the interoperability, and this first year, the micro-ROS consortium has implemented publication and subscriptions interfaces. OFERA has planned to add ROS 2 services interface in following improvements of micro-ROS.

One notable achievement of this first year of micro-ROS was the support of ROS 2 types. micro-ROS currently support the same types and same type definition as ROS 2 does. OFERA pushed the micro-ROS specific type support in rosidl_typesupport_microxrcedds This type support allows any type from ROS 2 to be ported to micro-ROS providing interoperability of the interfaces using it. However, the consortium has identified some issues regarding vector-like types due to memory limitations. , but the consortium will address on a following update of the micro-ROS type support package library.

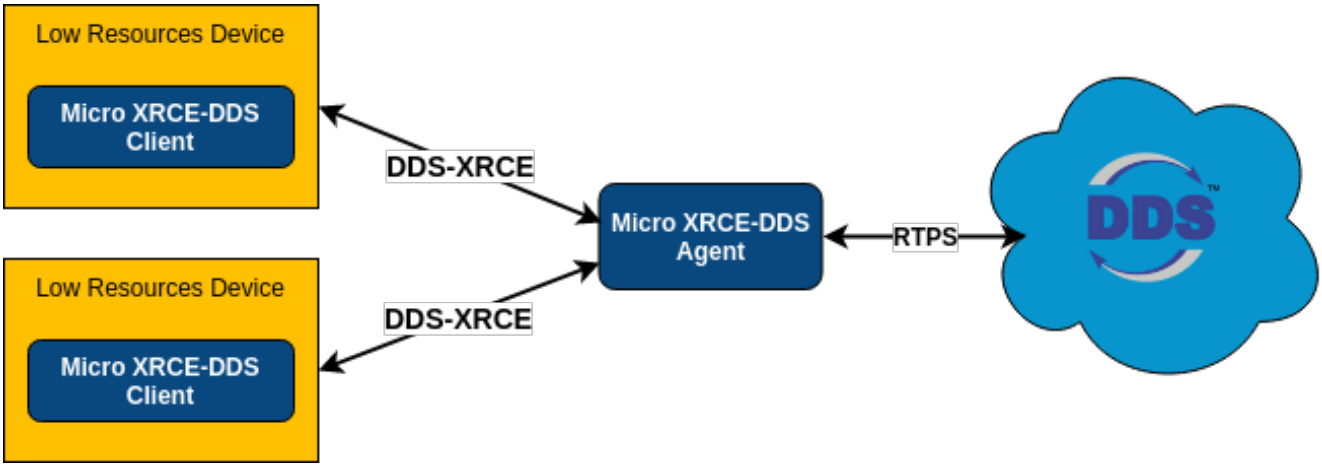
OFERA have provided demonstrations of ROS 2 messages from and to micro-ROS systems in the current micro-ROS version. The consortium pushed these demonstrations in micro-ROS-demos These demonstrations use the mechanism mentioned above where automatically generation of type support is achieved on both ends, ROS 2 and micro-ROS, using the same source type definition file. These demonstrations show the users how to include new types and how straightforward would be to integrate any types in both ends of the communication, in this case, micro-ROS and ROS2.

4.1.3 Architecture

ROS 2 interoperation is directly related with type support and middleware implementations. In the case of micro-ROS, consortium members have developed a type support mechanism similar to the one used in ROS 2. In micro-ROS, build system generates type support libraries in C language for each interface found in the workspace. The source of the generation is precisely the same as in ROS 2 that allows the reuse of types definition is easing the port of types from one end to the other end of the interoperable system. The following figure shows this interface generation process.



As stated above, another key point to allow micro-ROS - ROS 2 interoperability was the usage of interoperable middleware. On the ROS 2 end, the middleware used is based on DDS. On the other side, micro-ROS has been architecture and developed using a new OMG standard 100% interoperable with DDS. This new standard is DDS-XRCE, allowing to communicate embedded devices with DDS networks natively. DDS-XRCE provides the interoperability between micro-ROS devices with existing ROS 2 applications at middleware level.



This reuse of type definition together with the usage of interoperable middleware provides micro-ROS with native interoperation with ROS 2. From this point a wide range of interoperation for micro-ROS is opened, for example in the case of ROS 1, using the existing bridge from ROS 2 to ROS 1 will ensure that micro-ROS could interoperate with ROS 1 system hopping over ROS 2.

4.2 FIWARE Interoperability (FIROS 2)

4.2.1 Introduction

EPROS is the owner of FIROS2 FIWARE generic enabler, a component to bridge between several middleware components of FIWARE, including the FIWARE Context Broker and Fast RTPS, with ROS 2. The consortium will extend the bridge to support micro-ROS, with the goal of sharing the context data from micro-ROS devices to FIWARE components.

In other to achieve FIWARE interoperability, the consortium has improved and reused the existing interoperability between ROS2 and FIWARE. Despite that, FIROS2 will be extended to support micro-ROS natively. For this first release of micro-ROS no natively interoperability was attempted, but using already achieved micro-ROS - ROS 2 interoperability.

4.2.2 micro-ROS current status

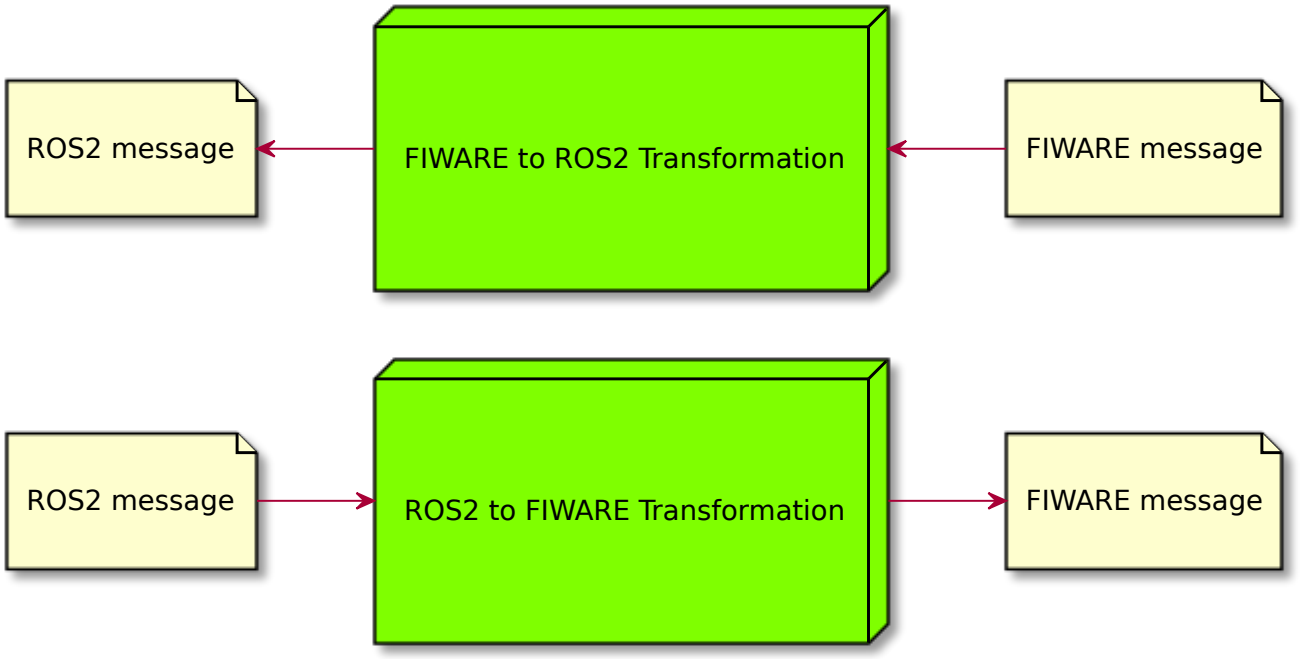
Interoperability with FIWARE in this first year of the project was achieved using the micro-ROS capabilities of being interoperable with ROS 2. micro-ROS consortium has adapted and modified the ROS 2 - FIWARE interoperability provider, FIROS 2. During this first part of the project, the consortium has created a demonstration where micro-ROS can interoperate with FIWARE context broker using ROS 2 and FIROS 2.

In the reporting period, OFERA did a micro-ROS - FIROS 2. This integration consists of the creation of custom transformation libraries for the ROS2 interfaces within FIROS 2. The transformation library from one side takes rmw implementation to deserialise/serialise the messages and on the other side uses FIWARE NGSI v2 messages. The current example can be found in the micro-ROS demos repository: demos

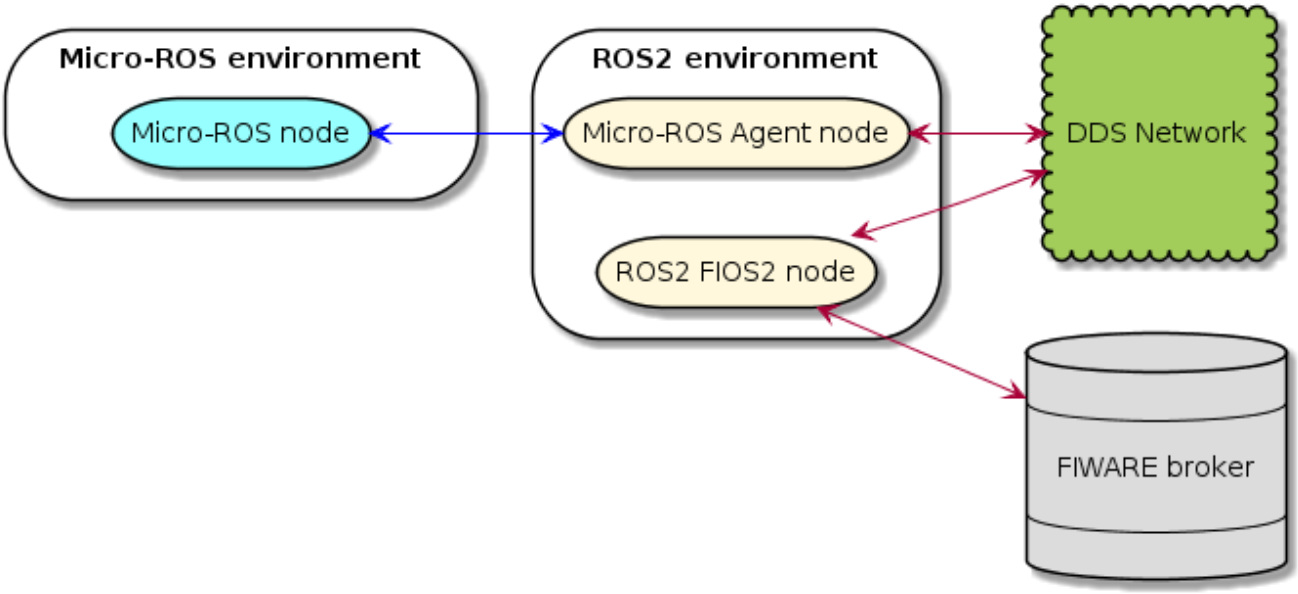
In the current architecture, there is no automatic generation, but the consortium has planned to dive in that matter and develop some automated generation tools. This automatization will not require the user to worry on the transformation library implementation. The automatization will be implemented similarly to type support generation, where each ROS 2 interface autogenerates its code. In this context, each ROS 2 interface found on the workspace will generate transformation code and configuration. This automatization will reduce type mismatch errors and keep ROS 2, micro-ROS and FIROS 2 transformation libraries synchronised.

4.2.3 Architecture

This interoperability is enabled thanks to FIROS 2. This eProsima integration service transforms ROS 2 messages to FIWARE NGSI v2 messages.



In the first version of interoperability with FIWARE. micro-ROS transformation library should be developed per each ROS 2 interface. This transformation library implementation should be provided along with a FIROS configuration to indicate which transformation library to use.



4.3 HRIM Information Model

Hardware Robotic Information Model (HRIM) is an information model for robots aiming to make use of predefined interfaces, data structures and parameters, focused at the robot component level. It makes use of model-driven engineering (MDE) approach to facilitate the development and integration of ro-

botic system at its design and deployment phases. The usage of such an information model allows component's self-description, and the usage of predefined interfaces avoiding manufacturer lock-in and enabling interoperability. This tool desires to empower robot end-users to fine grain their system, selecting the components that fit into their needs and/or use-cases.

HRIM proposes a set of rules each module has to meet to enable seamless interaction with other devices in the robot network. It does so by providing a meta-model that describes abstractly both individual robot modules as well as complete robots composed by a variety of them. Overall, HRIM facilitates the integration effort when building, maintaining or repurposing robots and simplifies their whole life cycle.

Comparing to current standards, such as OPC-UA, HRIM focuses mainly on the component rather than on the management of a complete system, for example, the management of a whole manufacturing facility. In terms of objectives, OPC-UA and HRIM diverge. On one hand, OPC-UA aims to provide a snapshot or description of the structure or state of a complete network composed by robots. On the other hand, HRIM's objective is defining how the robot components talk to each other. As we can see, even if at first glance both look to target the same needs, in practice, they do not share a focus, even though they could complement each other.

4.3.1 Status and next steps

Nowadays, ALR is actively designing its structure and developing the tooling for the generation of platform-specific implementations where a complete implementation of such an information model and its tools is expected. Thanks to this development and its integration process expected in micro-ROS, the use of HRIM under micro-ROS will ease microcontroller-based robot components to be integrated seamlessly in a robot, removing the integration hurdle that is required nowadays.

4.4 Conclusion

During the reporting period, essential interoperability with existing systems has been achieved.

ROS2 and FIWARE interoperability were achieved relatively easy thanks to the use of standard protocol (DDS and DDS-XRCE). One of the most significant efforts was on the integration of new tools to the existing ROS2 build system to automate the integration of micro-ROS with ROS 2.

ROS 2 integration needs further development to support all the data types. Also, development of interfaces needs to be done, more precisely, services and parameters are two ROS 2 concepts that will ensure better interoperability between micro-ROS and current ROS 2 systems.

From FIWARE point of view, interoperability has been achieved with the same extension as the one between ROS 2 and FIWARE. However, more in-depth interoperability will be required where micro-ROS Agent directly communicates with FIWARE without using DDS-FIWARE interoperability tools FIROS 2.

In the following steps of the project tooling and easing the configuration of interoperability parts (especially in FIWARE interoperability) will be one of the main objectives of the related tasks.

References

[1] O. (Object Management Group), 'Interoperability examples for the data-distribution service (dds) standard'. 2018 [Online]. Available: https://github.com/omg-dds

[2] O. (Open Source Robotics Foundation), 'Official ros 2 communication tests'. 2018 [Online]. Available: https://github.com/ros2/system_tests/tree/master/test_communication

[3] eProsima, 'Micro xrce-dds integration tests'. 2018 [Online]. Available: https://github.com/eProsima/Micro-XRCE-DDS-Integration-Tests

[4] godzilla-max, 'A dds-xrce implementation for rx65n'. 2018 [Online]. Available: https://github.com/godzilla-max/rose_sketch

[5] Robotis, 'Robotis xel network integration with micro xrce-dds'. 2018 [Online]. Available: https://xelnetwork.readthedocs.io/en/latest/