D3.14

FIROS2
Software new release Y2

Grant agreement no.	780785
Project acronym	OFERA (micro-ROS)
Project full title	Open Framework for Embedded Robot Applications
Deliverable number	D3.14
Deliverable name	FIROS2
Date	December 2019
Dissemination level	public
Workpackage and task	3.4
Author	Borja Outerelo Gamarra (eProsima)
Contributors	
Keywords	micro-ROS, ROS2, FIWARE, intercommunication, NGSIv2 protocol, communication bridges, context broker, SOSS, System Handler
Abstract	This document provides links to the released software and documentation for deliverable D3.14 *FIROS2_new_Release* of the Task 3.4 *FIWARE Interoperability*.

Contents

1 Summary

As explained in *D3.11 Interoperability Report - Y2* and in *D1.2 Annual Project Report - Y2* this year effort in the task 3.4 has been focused in aligning FIROS2 with SOSS development. SOSS is a systems integration platform developed as a joint effort between EPROS and OSRF and with help of a FTP from the EU funded project: ROS-Industrial.

I this document we present the software released which able to communicate FIWARE Context broker and micro-ROS. However SOSS does not limit its possibilities to FIWARE, SOSS allows micro-ROS applications to connect to any other system that is integrated with SOSS in the form of SOSS System Handler (See SOSS repository with a list of available System Handlers).

2 Acronyms and keywords

Term	Definition
IS	Integration service
FIROS2	FIWARE - ROS2 bridge
FTP	Focused Technical Project
OSRF	Robot Operating System 2
ROS2	Robot Operating System 2

3 Overview to Results

This document provides links to the released software and documentation for deliverable D3.13 *FIROS2_new_Release* of the Task 3.4 *FIWARE Interoperability*.

4 Links to Software Repositories

For communicate micro-ROS with FIWARE using SOSS, the user will need, FIWARE System handler to connect SOSS and FIWARE:

- Git repository: https://github.com/eProsima/SOSS-FIWARE
  Branch: feature/xtypes-support
  Commits: a4ba23e

User will also need SOSS and the ROS 2 System Handler, both hosted in:

- Git repository: https://github.com/eProsima/soss_v2/
  Branch: feature/xtypes-support
  commits: 840c73a
  ROS 2 System handler: /packages/ros2

5 Annex 1: FIWARE System Handler

content of https://github.com/eProsima/SOSS-FIWARE/blob/feature/xtypes-support/README.md from 23th December 2019

soss-commit 660078e

System handle to connect *SOSS* to *FIWARE*

5.1 Installation

To install this package into a workspace already containing SOSS, just clone this repository into the sources directory and build it:

```
$ cd <soss workspace folder>
$ git clone https://github.com/eProsima/SOSS-FIWARE.git src/soss-fiware
$ colcon build --packages-up-to soss-fiware
```

5.2 Use case - Connecting with ROS2

0. Prerequisites: curlpp and asio installed

1. Create a colcon workspace.

```
$ mkdir -p soss_wp/src
$ cd soss_wp
```

2. Clone the soss project into the source subfolder.

```
$ git clone https://github.com/osrf/soss_v2.git src/soss
```

3. Clone this project into the subfolder.

```
$ git clone https://github.com/eProsima/SOSS-FIWARE.git src/soss-fiware
```

The workspace layout should look like this:

```
soss_wp
...src
......soss
.........(other soss project subfolders)
.........packages
...........soss-ros2 (ROS2 system handle)
......soss-fiware (repo)
...........fiware (soss-fiware colcon pkg)
...........fiware-test (soss-fiware-test colcon pkg)
```

4. In the workspace folder, execute colcon:

```
$ colcon build --packages-up-to soss-fiware
```

5. Source the current environment:

```
$ source install/local_setup.bash
```

5.3 Usage

This system handle can be used to connect FIWARE with other systems.

5.3.1 Configuration

SOSS must be configured with a YAML file, which tells the program everything it needs to know in order to establish the connection between two or more systems that the user wants. For example, if a simple string message wants to be exchanged between FIWARE and ROS2, the configuration file for SOSS should look as follows.

```
systems:
    ros2: { type: ros2 }
    fiware: { type: fiware, host: CONTEXT_BROKER_IP, port: 1026}

routes:
    fiware_to_ros2: { from: fiware, to: ros2 }
    ros2_to_fiware: { from: ros2, to: fiware }

topics:
    hello_fiware: { type: "std_msgs/String", route: ros2_to_fiware }
    hello_ros2: { type: "std_msgs/String", route: fiware_to_ros2 }
```

To see how general SOSS systems, users and topics are configured, please refer to SOSS' documentation.

For the FIWARE system handle, the user must give two extra YAML key-value pairs which are the host and port in which this system handle will try to connect to an instance of FIWARE's Orion context broker.

FIWARE does not allow certain characters in its entities names. For this reason, if a type defined in the topics section of the configuration file has in its name a `/`, the FIWARE system handle will map that character into two underscores. This is something important to notice when connecting to ROS2, because in ROS2 most of the types have a `/` in their names. Also, notice that in FIWARE the type will be published as an entity with the same name but with every slash substituted with two underscores.

With that, in the YAML file the type under the `topics` section can have a `/` (and to connect to ROS2, normally it will HAVE to), just remember that in FIWARE the entity created must have the same name but with two underscores instead of a slash.

Notice that this system handle maps soss messages directly to a JSON compatible with FIWARE. As FIWARE doesn't allow nested types, neither does this system handle.

5.3.2 More information

- For more information, you can see the demo steps and the related video

- Also, you can have a look at the internal design
- For a fast configuration, you can use the dockerfile. **NOTICE**: Fiware Orion context broker may not be able to reach the docker IP, as is not accessible outside the host computer by default. You can run (or build if necessary) the docker with –network=host to share the network interface of the host with Docker, and make it accessible in your LAN.

5.3.3 Changelog

5.3.3.1 v0.2.0

- Added dockerfiles
- Added integration tests
- Removed asio as local thirdparty
- Subscription host and port automatically generated if they are not specified.

5.3.3.2 v0.1.1

- Fiware communication take into account the topic type
- Added thread protection

5.3.3.3 v0.1.0

- Fiware communication in both directions based on topic

Supported by ROSIN - ROS-Industrial Quality-Assured Robot Software Components. More information:
* http://rosin-project.eu