Topic	ICT-09-2019-2020 (H2020)
Acronym	ATLANTIS
Title	The Atlantic Testing Platform for Maritime Robotics: New Frontiers for Inspection and Maintenance of Offshore Energy Infrastructures.
Project number	871571
Delivery date	01.04.2022
Deliverable number	D2.6 Communication links and interfaces for interoperability
Dissemination level	PUBLIC
Lead Beneficiary	SPACEAPPS

Communication links and interfaces for interoperability

SPACEAPPS, VTT, INESC, IQUA, UdG, ECA

Actions

	Action	Organisation	Date	Deadline
Technical Manager	Requested deliverable from the Deliverable Responsible.	VTT	21.03.2022	28.03.2023
Deliverable Responsible	Prepared draft of the deliverable.	SPACEAPPS	28.03.2022	28.03.2022
Technical Manager	Approved the updated draft as the first version.	VTT	28.03.2022	28.03.2022
Quality Manager	Approved the updated first version as the second version.	UdG	29.03.2022	29.03.2022
Project Coordinator	Approved the updated second version as the final version and sent to the European Commission.	INESC TEC	31.03.2022	31.03.2022

Disclaimer

This document does not represent the opinion of the European Union nor the European Commission is responsible for any use that might be made of its content. The ATLANTIS consortium cannot warrant that information contained in this document is free from risk and, neither the European Commission nor the ATLANTIS consortium parties are responsible for any use that may be made of the information contained therein.

This document may contain material, which is the copyright of certain ATLANTIS consortium parties, and may not be reproduced or copied without permission. The commercial use of any information contained in this document may require a license from the proprietor.

The sole responsibility for the content of this publication lies with the authors and all ATLANTIS consortium parties have agreed to full publication of this document.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Table of Contents

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Table of Figures

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

List of tables

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

List of Abbreviations

API: Application Programming Interface
ASV: Autonomous Surface Vehicle
AUV: Autonomous Unmanned Vehicle
CLI: Command Line Interface
CPE: Customer-Provided Equipment
DB: Database
DDS: Data Distribution Service
GPS: Global Positioning System
GSM: Global System for Mobile Communication
HMI: Human Machine Interface
IMR: Inspection, Maintenance and Repair
NA: Not Applicable
NOS: Portugal Telecom Company
OGC: Open Geographical Consortium
O&M: Operations and Maintenance
OMG: Object Management Group
PoE: Power over Ethernet
QoS: Quality of Service
RF RC: Radio Frequency Remote Control
ROS: Robot Operating System
ROV: Remote Operated Vehicle
SCC: Supervisory Control Centre
SDR: Software Defined Ratio
TBD: To Be Defined
UAV: Unmanned Aerial Vehicle
UI: User Interface
USBL: Ultra Short BaseLine
UV: Unmanned Vehicle
UxV: Unmanned Vehicle of type X
VPN: Virtual Private Network
WP: Work Package

1. Introduction

1.1. Scope of Work

This report describes the hardware and software systems that allow the ATLANTIS assets (robots, control centres, sensors etc.) to interact. Once a vehicle is compatible with these systems, it will be **interoperable** with all the other systems used in the IMR tests in Viana do Castelo. The work included the selection and development of a middleware that will be used onboard all the robots tested in the pilot site. It also describes the communication links that are made available (in both Coastal and Offshore Testbed). This document details how the functional interoperability is assured between the infrastructure of ATLANTIS and other future robots to be validated for IMR activities.

Both interoperability specification and the availability of the communication links are made public. They will be available for the companies that want to test their robots inside the test pilot. The interoperability specification was designed by looking at the ATLANTIS scenarios and the robots that will demonstrate them. The consortium covers a wide range of unmanned vehicles, with various levels of autonomy including underwater, surface and aerial robots. The principles that guided the work done between partners for designing the interoperability specification are:

Decentralised interoperability: each robot and asset present in the communication system will be able to connect and disconnect from the network without influencing or impairing the well functioning of the other components. The system is fully decentralised, which means that there is no master or server that orchestrates the communication between robots and other connected devices (sensors, power generators etc.).

Heterogeneity of assets: Connecting to the interoperability network is done in the same way regardless of the type of hardware connected. All robots connected in the network should be able to discover each other and dynamically understand their capabilities.

Compatibility: The hardware and software technology chosen to ensure interoperability should be compatible with the most used middlewares and communication protocols on the market.

1.2. System Architecture Requirements

Design of the interoperability is part of the overall architecture of the ATLANTIS testbed and it makes the link between vehicles, control centre and the rest of the pilot. Looking at the proposed architecture (Figure 1), we can identify that the communication, O&M data processing and the Supervisory Control Centre (SCC) will be stable systems during various phases of the testbed exploitation, while the sensors, robots and robot HMIs are elements specific for different use cases being tested on the platform.

In the offshore energy sector, the end user goal is to improve the efficiency of IMR operations for the whole infrastructure. The SCC has the role to provide constant monitoring and control to the robotics operations and the capacity to constantly deliver the operations result in a consistent way, regardless of the type of robot performing the operation. The SCC provides the tools specific for a control centre, which include:

- Monitoring of environmental data,
- Monitoring and control of the UxVs,

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

- Deployment of operations,
- Planning and data visualisation.

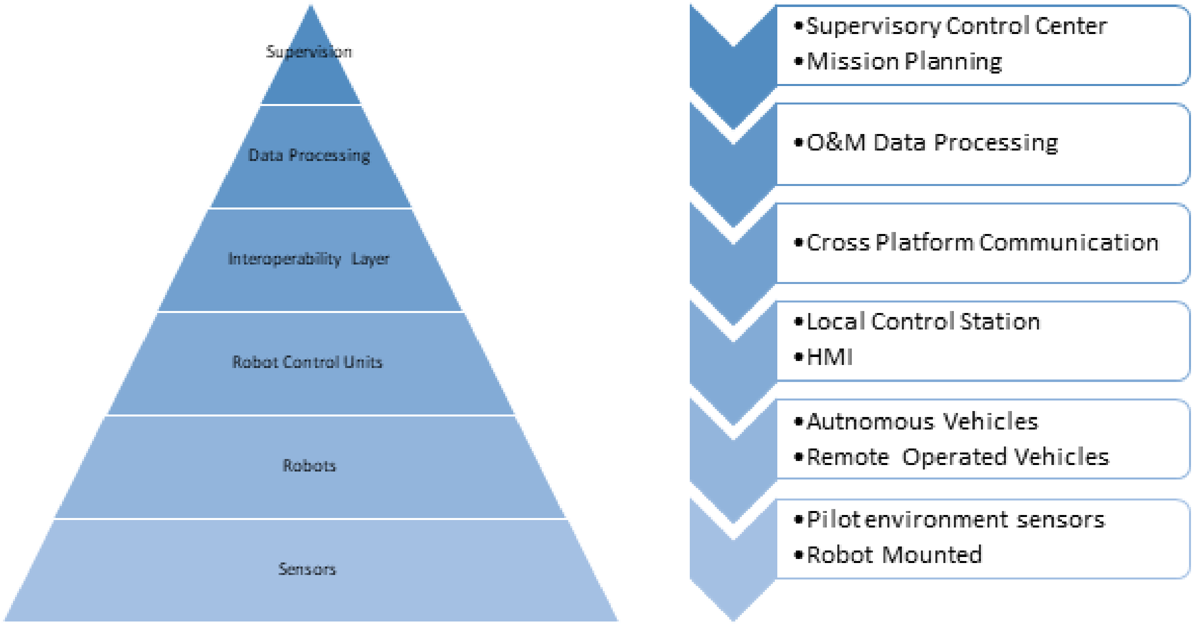


Figure 1. Technological layers in ATLANTIS System

From an abstract architectural perspective, the layer that links the robots with each other and with the onshore control centre is the *Interoperability Layer*. This layer has the main role of unifying the data types coming and going to the deployed robots. The higher layers will be able to have a unified access point to the data exchanged between robots named “data gateway”. The connection with the supervisory control centre will act just like another node in the network of assets.

Looking downward from the interoperability communication layer, there are four main data categories that are supported between assets:

- Info: each asset publishes the information about itself to all peers, thus being discoverable. The other nodes can exchange data with this asset based on this information.
- Commanding: a robot can receive commands from a control node.
- Monitoring: timestamped data used for generating situational awareness and the full history of the operation.
- Data retrieval: large amounts of data after performing a survey (images, point clouds, raw or processed data).

Each one of the robots has a specific set of commands that can be engaged by the control centre. The interoperability layer has the role to advertise towards the control centre which are the capabilities of each robot and to accept from the supervisory control centre a command format that is generic for all the robots. Inside the interoperability, the commands received from the control centre are translated into each robot's command specifications.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

To translate the generic command structure to the specific format, the interoperability layer must understand each of the robot's capabilities. This will be done in an automatic way the first time a robot is connected to the ATLANTIS Testbed system. The configuration of the robot will be passed to the interoperability layer through a configuration file. The configuration file will be stored on the robot on-board computer and it will contain the list of supported commands, telemetry and data payloads, along with the robot namespace. The interoperability software can communicate with an instance of the SCC launched in the cloud. This is possible by using access tokens and secure web protocols.

From a deployment view, the interoperability layer software will run on both the onshore facility and the offshore unmanned vehicles. The module running on shore will act as a data gateway between the SCC/Data Processing units and the rest of the system. The physical link between the onshore and offshore parts of the interoperability layer can be a satellite link, 4G, or the existing WindFloat communication infrastructure.

On the robot side, the module is deployed as a library. The library can be a dependency for the robot or the local robot control centre. From the Interoperability layer each robot control unit or robot will be linked through an API.

1.3. Main Objectives

The scope of the work performed in **Task 2.5 Communication Links and Interoperability** was related to:

1. Design and development of a **data exchange convention** which will be used to ensure the interoperability between heterogeneous systems coming from different providers. This convention will be used across all the vehicles in the project and serve as a link with the floating structures and the SCC.
2. Establish the **communication architecture** and perform the tradeoff between different hardware solutions that are compliant with the functional and performance specification of the project. The proposed architecture should support all the communication mediums that must be used in marine IMR and be compatible with the existing standards.
3. Validate the **communication systems** and **architecture** selected during the task. All the nodes shall be tested in a table top configuration to validate the feasibility of the architecture. The installation will be performed during WP3.
4. Provide a **public document** that serves as a reference for the future end users of the ATLANTIS Pilot. This document includes the reference to the communication links that can be used, the communication protocols and gives reference to an interoperability software that will connect the robots to the rest of the system.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

2. Hardware Communication Systems

2.1. Geographical Considerations

Before deciding on the network topology and the types of communication that can be successfully used for exchanging data in the pilot, an analysis of the constraints that emerge from the chosen test site and available infrastructure must be done. Distance between assets is relevant both on the coastal and offshore sites. Besides the positioning in space, other constraints like costs or safety requirements are also relevant.

2.1.1. Coastal Testbed Location

The coastal testbed is going to be installed in the Viana do Castelo port. The location of the floating structure is described on Figure 2 and and will be approximately 150m away from the building where the local SCC centre will be set up.

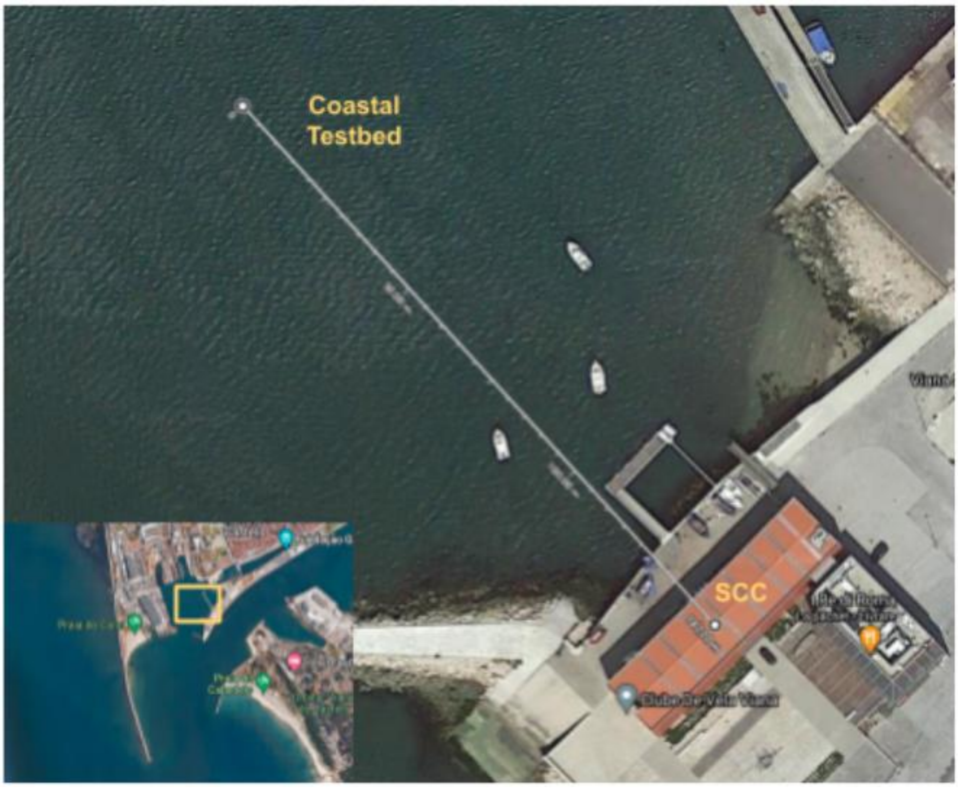


Figure 2. Line of sight distance between SCC Building and the Coastal Testbed

This choice of location for the floating mock and the SCC provides a short-range direct line of sight between the two. On the other hand, the SCC room will not have windows directed to the coastal pilot as it is shown in Figure 3. The partners testing the robots on site will benefit from an observation area that is adjacent to the test area where they can mount gazebos and will be able to service their robots. The observation zone is highlighted in Figure 3.

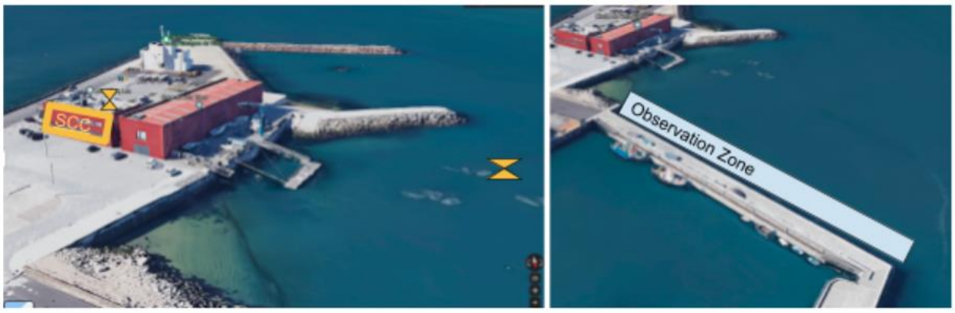


Figure 3. Supervisory Control Centre, observation zone and coastal testbed geography

Even if the coastal testbed is in close proximity to the onshore facilities, the objective of the pilot is to create a test site analogous with the offshore conditions. The communication system that will be used to validate the robots will be the same as the one used in production, namely satellite communication links and GSM. Nevertheless, for testing purposes we will also set up a directed Wi-Fi point-to-point between the testbed and the SCC. On the local networks, all the robots can communicate using the interoperability conventions. These local networks will consist of a mix between WiFi and acoustic. The illustration of the connections on the coastal site can be seen in Figure 4.

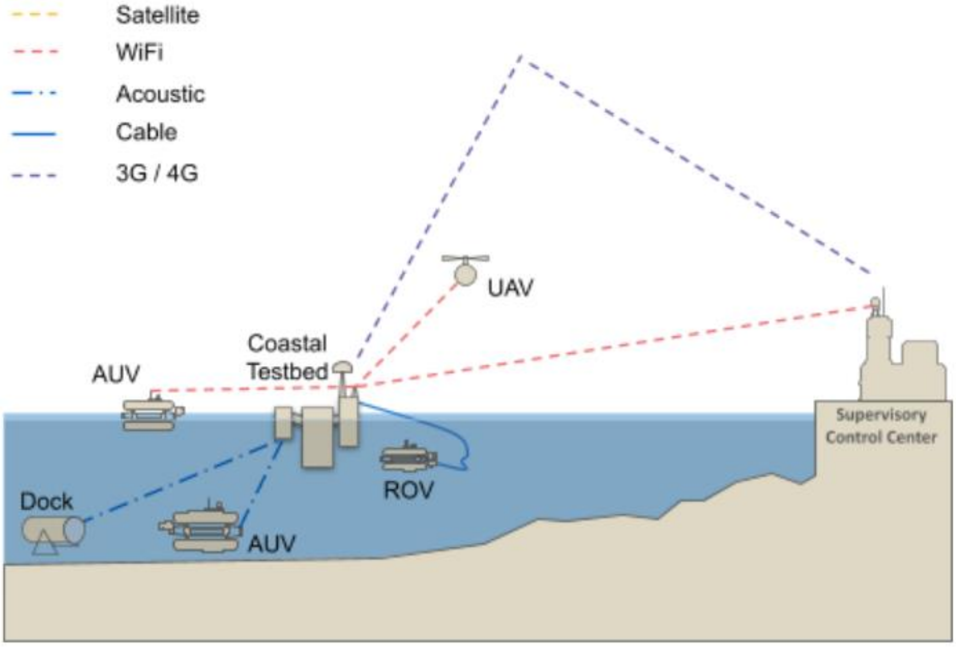


Figure 4. Coastal Testbed Communication

2.1.2. Offshore Testbed Location

In the case of the offshore test site, the structure of the communication is similar with the one designed for the coastal testbed and can be observed in figure 5. The main links between the test site and the onshore centre consists of satellite communication and 3G/4G.

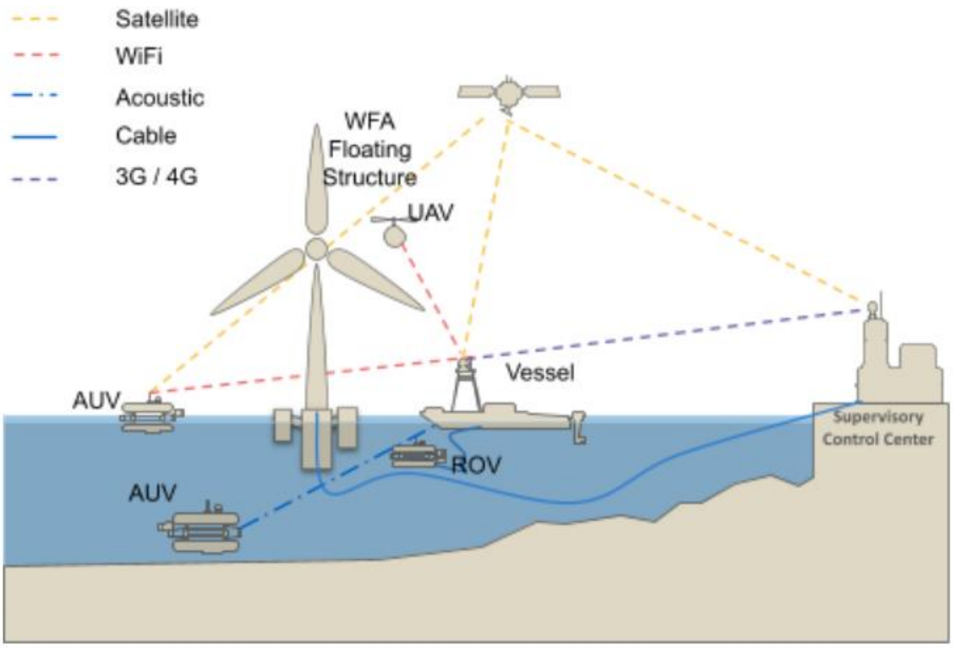
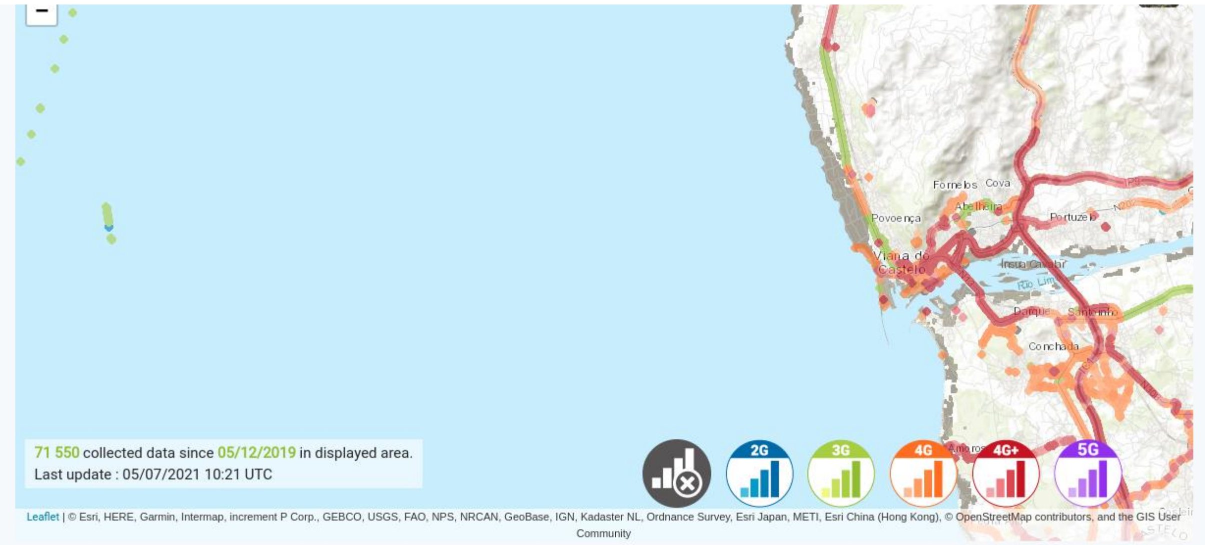


Figure 5. Offshore Testbed Communication Structure

For safety and efficiency purposes, the consortium has decided to install the remote communication hardware on the support vessel that will carry the testing crew and will serve as the launching platform for the robots. The area where WindFloat Atlantis is installed has a partial 3G coverage (Figure 6) that can be leveraged during tests. The satellite communication link can be used as an option to the gsm one. The advantage of having two options is related to testing data exchanges at different rates and also optimising the costs.


71 550 collected data since 05/12/2019 in displayed area.
Last update : 05/07/2021 10:21 UTC

Leaflet | © Esri, HERE, Garmin, Intermap, increment P Corp., GEBCO, USGS, FAO, NPS, NRCAN, GeoBase, IGN, Kadaster NL, Ordnance Survey, Esri Japan, METI, Esri China (Hong Kong), © OpenStreetMap contributors, and the GIS User Community

Figure 6. GSM Coverage in Viana do Castello

2.2. Communication Architecture

ATLANTIS Pilot communication architecture shall support the testing of IMR vehicles beyond the scope of the project. Having a large set of scenarios and a heterogeneous fleet of robots involved in this stage will facilitate building a generic architecture that will be compatible with a large variety of use cases. Different scenarios involve different vehicles that can be autonomous or remotely operated (D2.1). In the case of autonomous vehicle deployment, there are also cases in which two or more robots collaborate for achieving a common goal. In the current document, we are addressing all cases by describing the necessary interfaces that have to be established between various systems in order to successfully demonstrate all the planned scenarios.

Table 1 lists the robots and support assets used for each of the scenarios and what are the communication protocols that are supported for this purpose.

Table 1. Robots used for scenario demonstration:

		Robots and Assets	
Scenario	**Description**	**Coastal Test**	**Offshore**
1	Blade Tower Inspection	-	ASV: Zarco (INESC) UAV: Crow (INESC)
2	IMR of Floating Structure	AUV: Girona (UdG) AUV: Girona (IQUA)	AUV: Girona (IQUA)
3	Repair of Cable Protection System	Simulation (TBD)	
4	Underwater monitoring over large period of time	AUV: Sparus (UdG) Docking Station (UdG)	-
5	Maintenance of Foundations	AUV: Girona (UdG)	-
6	Maintenance of Scour Protection	ROV (ECA)	ROV (ECA)
7	O&M Operations Supported by Unmanned Vessels	Simulation (TBD)	
8	Optimization of Robotic Based Operations	NA	NA

There are three main operation locations that interact with each other: The onshore/cloud SCC, the Coastal Testbed and the Offshore Testbed. As the SCC is the point of access for the end user, we must

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

establish the best possible communication connection with the other two locations. In the case of the Coastal Testbed directional WiFi, 4G. 5G or Software Defined Radio (SDR) could be used, while for offshore communication we can either establish a satellite link or make use of existing communication lines if available.

Apart from the communication from the SCC, each of the pilot sites have to have a means of accessing the other devices and a mechanism to send that data through the SCC established connection. The two cases are solved in the same way by first setting a data gateway that bridges the local and remote network. As for the local communication, WiFi, SDR, USBL (for underwater) and cable (Ethernet or fibre optics) are potential solutions to, depending on the specific hardware configurations and operational requirements.

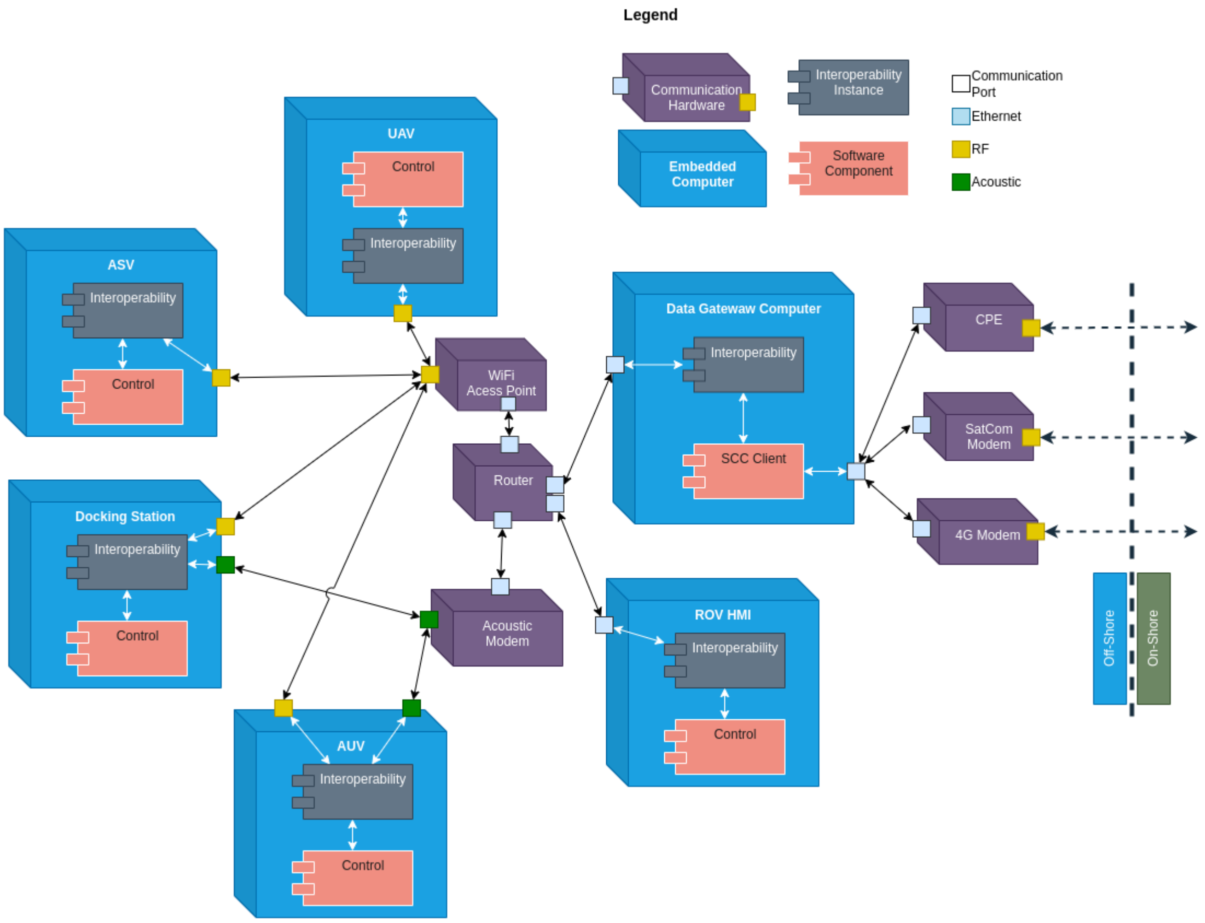


Figure 7. Deployment Diagram of the Atlantis Testbed Interoperability Links

The deployment view from Figure 7 depicts the distribution of communication and links with computers on the offshore or on the analog coastal testbed. The connection between the software and hardware are shown by identifying where each software component is installed and launched. Using the pilot for testing and validating technologies implies a generic approach that will replicate analog conditions regardless of the configuration of the test. The targeted operations are robotics IMR for offshore windfarms. This implies that the systems have two locations of deployment that are significantly separated: offshore and onshore. In the scope of this document we are focusing on the connections between robots and the pilot from both software and hardware perspective.

2.2.1. Offshore - Onshore Link

Communication between the testbed and the control centre will be done using the 4G network, satellite communication or the wind flow communication infrastructure. The communication architecture from the Coastal Testbed is similar to the one foreseen on the Offshore Testbed. In both cases, data will be received through a remote communication endpoint. The testbed has to accommodate the necessary hardware for the remote communication endpoint and for a local communication network.

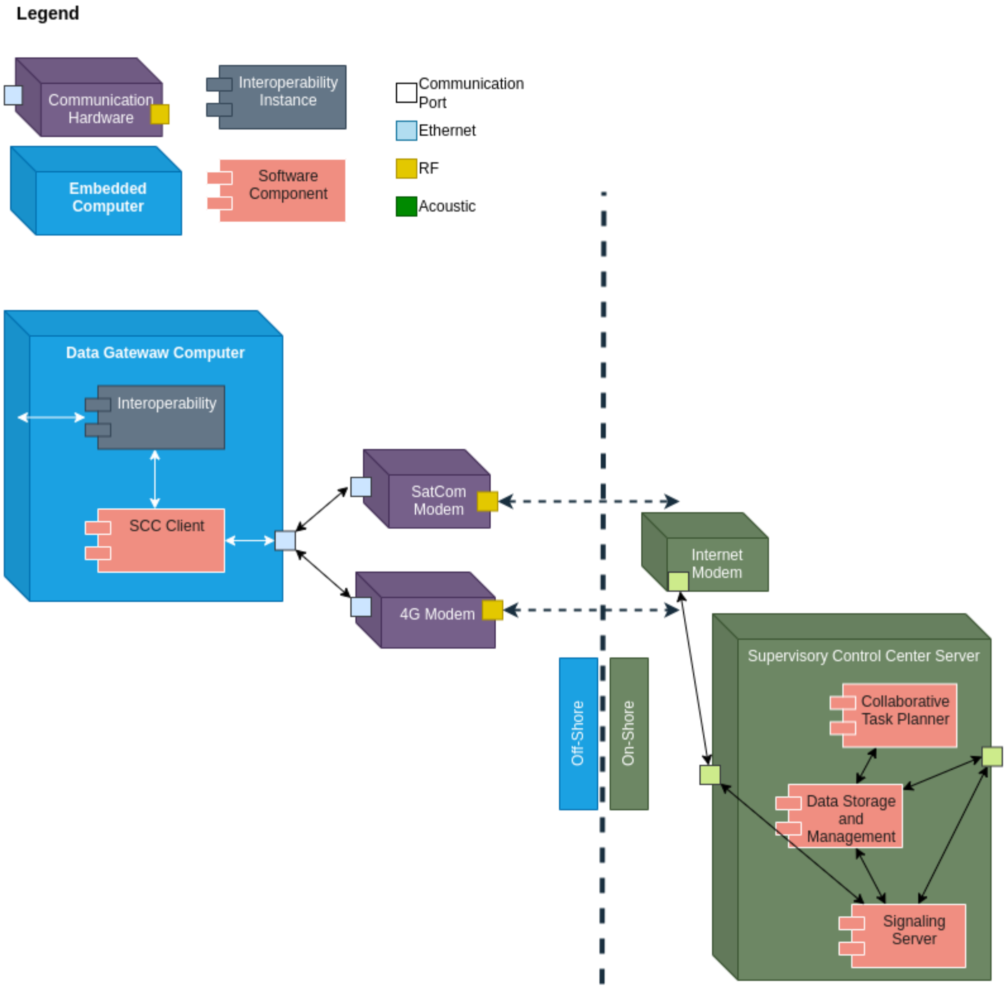


Figure 8. Atlantis Onshore to Offshore Link

Each one of the computers used during the execution of inspections will have an instance of the interoperability library installed. The data will be exchanged between the computers using different types of communication hardware. There are two aspects to take into consideration for defining the right type of physical transport. The first one is the proximity between two computers and the second one is the

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

medium in which the base platform is active. The connection between the offshore and onshore segments of the pilot is done through a data gateway computer. The computer will act as a node in the network of robots and will translate the relevant information to be sent at the onshore data centre for observation storage and analysis. The communication links connected to the data Gateway have to support long range communication. We have selected for the pilot to use satellite link and a 4G connection interchangeably. This will provide more than one auction to evaluate and test with different amounts of data.

Exceptionally for the coastal testbed we have introduced a point-to-point communication link using directional Wi-Fi antenna. This will be used for the coastal operations as a backup high speed link between the local network including the robots and the local supervisory control centre (SCC).

2.2.2. Local Networks for Robot IMR

The rest of the network topology includes the local communications between robots and the data gateway computer.

The **ROV** is teleoperated on-site by an operator. The operator will manipulate the ROV HMI from the same platform as the data gateway computer is installed in. This can be either the floating structure or support vessel. The link between the local network and the ROV will be done through the HMI computer, which will be connected with the rest of the local network via ethernet.

Among the platforms used during the project there are two components that have to operate under water. These are the **AUV and the Docking Station**. For staying connected while underwater each of these platforms will be equipped with acoustic modern. The acoustic communication will be used for both data exchange and as an underwater positioning system since we cannot use radiofrequency positioning systems like the GPS. The AUVs will mostly be used to gather data by using onboard sensors in the proximity of the floating structure. These data have to be transmitted back to the supervisory control centre for analysis, storage and observation. While useful underwater, the acoustic link is not able to support a large bandwidth, so the underwater vehicles will be equipped with Wi-Fi access points which will enable them to connect using a radio link after emerging to the surface. All the AUVs will be equipped with both acoustic and Wi-Fi communication endpoints as shown in Figure 9. On the docking station side, It is also important to enable large data exchange especially during the docking operations which have to be monitored during execution. To be part of the same network, the docking station can be connected to a buoy that has ethernet tether and a Wi-Fi access point.

The other vehicles there are used for inspection and maintenance activities are the surface and the aerial vehicle. Each one of these we expect to have a permanent Line of Sight with the base platform (coastal testbed or support vessel). This will allow us to permanently keep an open RF connection and we have chosen to use Wi-Fi for this purpose. No oher mid range communication modems are foreseen on these types of vehicles.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

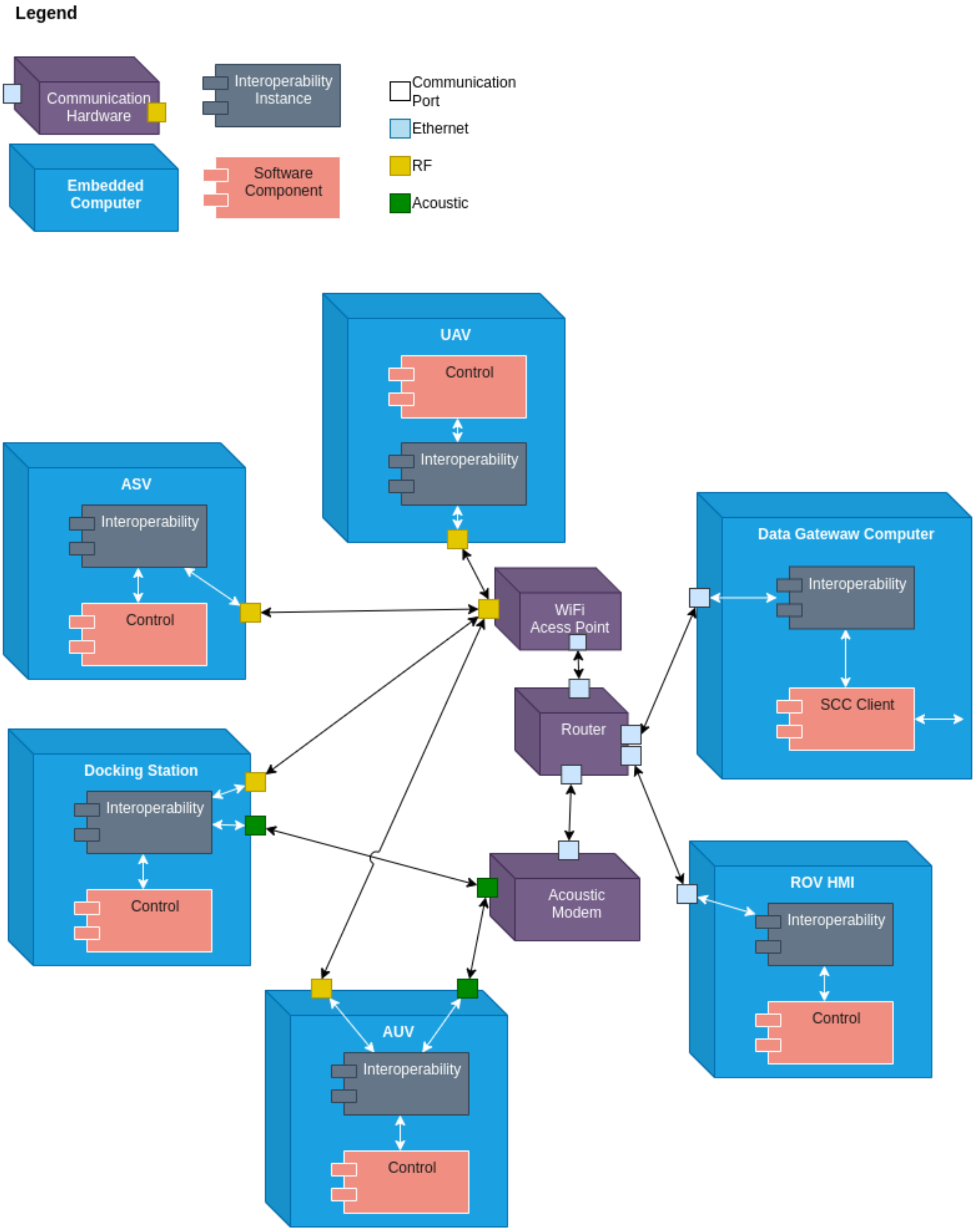


Figure 9. Atlantis Interoperability Deployment

2.2.3. Communication Networks by Location

Based on the defined network architecture, each location of the testbed has a list of communication endpoints associated. This list is presented in table 2.

Table 2. Communication endpoints for each location

Node	WiFi	SatCom	4G	Cable
INESC Support Vessel	X	X	X	
SCC Room	X			X
Coastal Testbed	X	X	X	

Figure 10. is showing the communication types that will be available on all three locations: Onshore (SCC), Coastal Testbed, Offshore (INESC Vessel)

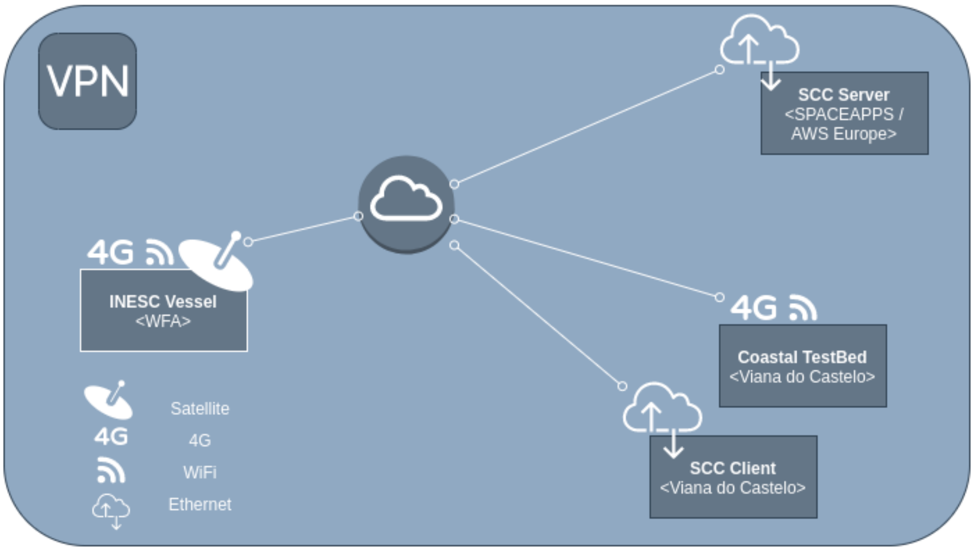


Figure 10. ATLANTIS Remote Communication Endpoints and Connection Options

On the pilot we will install three networks to connect with the robot: Ethernet, Wi-Fi and acoustic links. The communication hardware installed for the pilot on the testbed platform acts like a data gateway by linking the unmanned vehicles, local control units and the testbed sensor nodes to the remote link and implicitly to the SCC.

The onshore testbed will be equipped with the communication systems described in the above sections. The topology of the local network and the cabling schematic can be observed in Figure 11.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

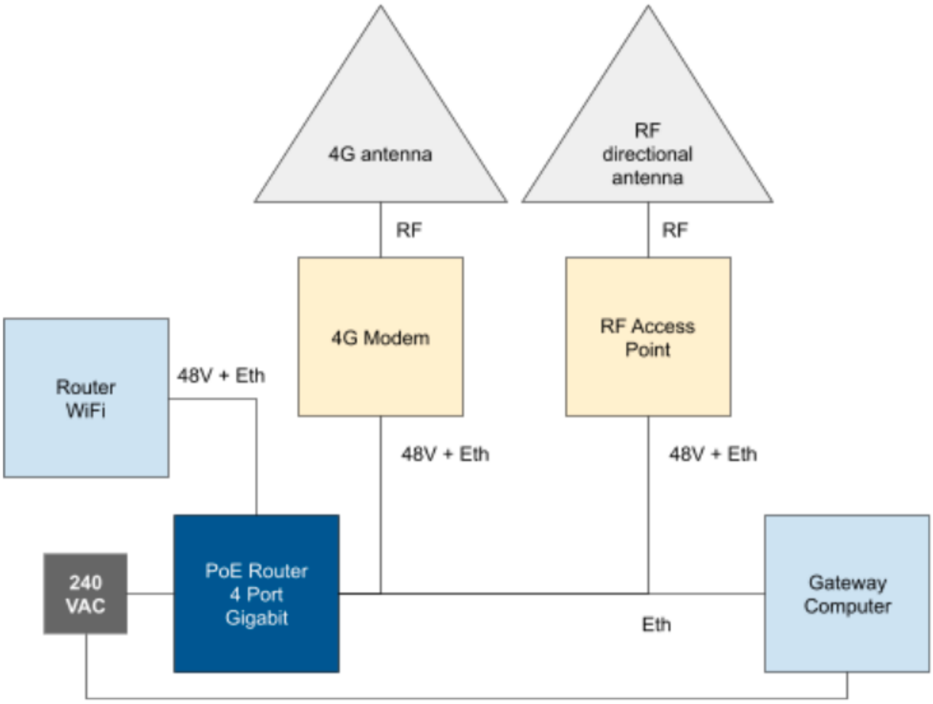


Figure 11. Onshore Pilot Hardware High Level Setup

2.2.4. Communication Systems Distribution

Based on the deployment diagram identifying each one of the vehicle types, the list of communication end points for each of the vehicles and test platforms is composed: Table 3. In addition to the communication hardware the table also shows the positioning devices that use acoustic and RF (USBL respectively GPS) since they imply data exchange using the same medium. When selecting the hardware and designing the mounting strategy for both acoustic and rf antennas it is highly important to consider the positioning systems as well.

Table 3. Communication and positioning end points per vehicle and test location:

Vehicle	**WiFi**	**Acoustic**	**Satellite**	**USBL**	**RC RF**	**4G**	**GPS**
Girona 500	X	X		X			X
Sparus	X	X		X			X
Zarco	X	X		X			X
Stork	X				X		X
Mares	X	X		X			X
ECA ROV				X			

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Docking St.	X	X					
Support Vessel	X	X	X	X	X	X	X
Coastal Testbed	X	X	X	X	X	X	
SCC	X						

This table will serve as a check tool during the pre-test verification for any robot and location.

2.3. Hardware Trade-off Analysis

2.3.1. Satellite Solution Trade-off

The selection of the satellite communication solution had two stages. First the tradeoff between different established products aimed for the marine industry was made. The best fit product was chosen Thalys VesseLINK for the following reasons:

- Partners experience working with this hardware
- Pre-existing compatible devices
- Cost

A second option was to investigate the comparison with a new low-cost high-speed satellite internet like Starlink (Table 4).

Table 4. Tradeoff analysis between Starlink and Iridium Certus

	Iridium Certus	**Starlink**
Maturity	2nd Generation (2017)	Beta
Availability	Now	End of 2021 (?)
Advertised Speed	700 kbps (down) 352 kbps (up)	50 Mbps
Advertised Latency	200 ms	40 ms
Communication Band	L Band	Ku-, Ka-, and E-band
Data Limit	5GB	N/A
Monthly Cost	$ 1800	$ 100
HW Cost	$ 4500	$ 500
Marine Applications	Yes	No Available Data

Compatibility		
Antenna	Omnidirectional	Directional with 1DoF Actuation.
Lead Time	2 weeks	Unknown (first come, first served)

Iridium Certus with the Thalys Vesse model and antenna are the best option for our application. While the Starlink will have a very good connection for a much lower price, the antenna is built for fixed ground, directional operations which makes it unusable onboard a vessel or a floating structure.

2.3.2. Remote Connection Type (Testbed to onshore)

Another important design choice is related to the main type of communication to be used for the data exchange between onshore and testbed. The four options taken into consideration are:

- Satellite Communication
- Cable Communication
- 4G Communication
- WiFi

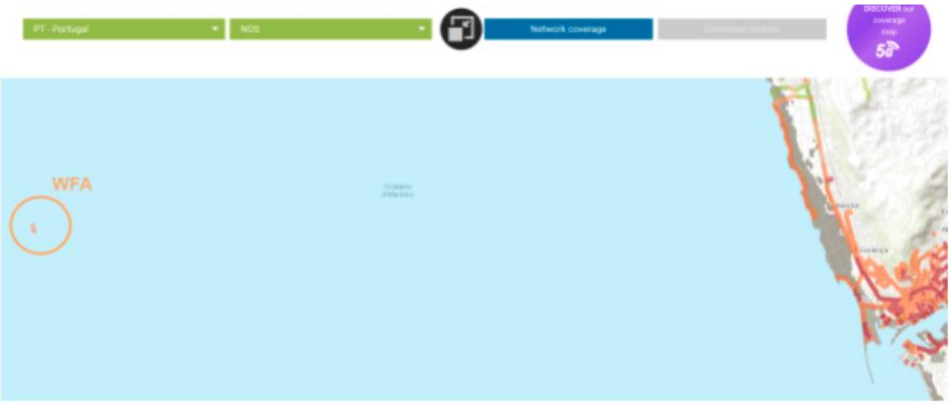


Figure 12. NOS 4G Coverage - WindFloat

The trade-off is done by assigning a score from **1 (Most Inconvenient)** to **5 (Most Convenient)** for a series of criterions: Cost of Equipment and Service, Installation Overhead, Reliability and Performance. The added scores will present a hierarchy to facilitate the most convenient option.

Table 5. Coastal Testbed Communication Option Assessment

	Satellite	**Cable**	**4G**	**WiFi**
Equipment Cost	3	1	4	4
Service Cost	2	5	4	5

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Installation	4	1	4	4
Reliability	4	4	4	3
Performance	3	5	4	4
Relevance	5	3	4	2
Total	**21**	**19**	**24**	**22**

In the particular case of WindFloat we can observe in Figure 12 that the 3G or 4G connectivity can be achieved from the test site. The scores provide small differences between Satellite, 4G and WiFi connections and the partners decide to install all three of them on the pilot structure. The Satcom has a high usage price compared with the other two, but it is important as a representative means of data exchange in the open sea. The 4G and WiFi can act as backup and cost reduction methods that will be used during the trials.

2.4. Communication Hardware Selection

2.4.1. WiFi Hardware

The agreed network configuration for using short range communication links is to use 802.11a (2.4 GHz Band) and 802.11g (5GHz Band) which provide communication capability up to 56 Mbps. The two WiFi protocols are compatible with most of COTS hardware that is currently used in the industry.

Table 6. X: WiFi access points

Vehicle	WiFi Access Point Model and Reference	Owner
Girona 500	Ubiquiti Bullet M 2 (BM2HP)	UdG
Sparus II	Ubiquiti Bullet M 2 (BM2HP)	UdG
SENSE	Wavlink AC 1200	INESC
CROW/RAVEN	Wavlink AC 1200 or TP-Link AC1300 High Gain	INESC
RAYA	TBD	INESC
Docking St.	TBD	UdG
Vessel	Outdoor Router 3G/4G WiFi	INESC
Testbed	Outdoor Router 3G/4G WiFi	INESC

2.4.2. Acoustic Communication & USBL

All the underwater vehicles are equipped with acoustic modems that support communication and USBL localization. The compatibility between these units is given by the producer. Table 7 describe the models mounted on each vehicle.

Table 7. Acoustic Modems

Vehicle	Acoustic Modem and Ref	Owner
Girona 500	Evologics S2CR18/34 USBL	UdG
Sparus	Blueprint SeaTrack X150 USBL/Modem beacon	UdG
RAYA	Evologics S2CR 18/34 USBL	INESC
ECA ROV	Blueprint SeaTrack X150 USBL/Modem beacon	ECA
Docking St.	Blueprint SeaTrack X150 USBL/Modem beacon	UdG
Vessel	Evologics S2CR 18/34 USBL Blueprint SeaTrack X150 USBL / Modem beacon	INESC
Testbed	Evologics S2CR 18/34 USBL Blueprint SeaTrack X150 USBL / Modem beacon	INESC

The offshore operations will be performed with the support of a vessel provided by INESC (Figure 13). This vessel will host the communication nodes in the same architecture as the one designed for the onshore platform.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

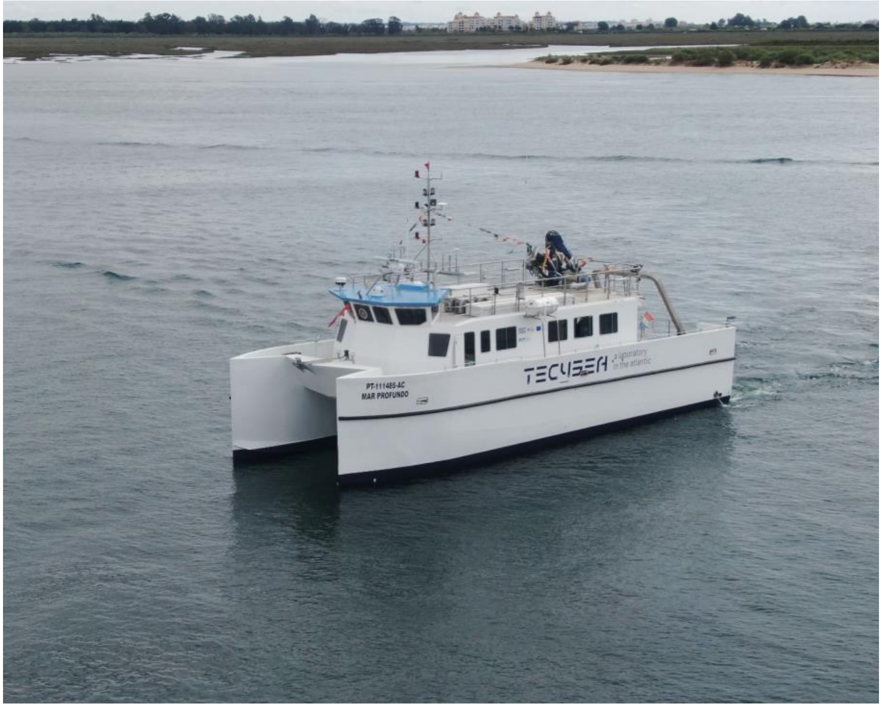


Figure 13. Support Vessel TE4SEA

The list of communication hardware for each of the identified communication endpoint that is part of the common Atlantis infrastructure are listed in the tables below:

Table 8. X: Hardware selection for the Testbed and Support Vessel

Item	Power	Procured by	Product
4G Modem	PoE: 48 Volt	INESC	Outdoor WiFi 4G Router
4G Antenna	-	INESC	Outdoor WiFi 4G Router
Router - Optional	48V PoE	INESC	MicroTIK
RF Access Point	48V PoE	INESC	Outdoor WiFi 4G Router
RF Antenna	-	INESC	Outdoor WiFi 4G Router
P2P Access Point	PoE	INESC	Mikrotik CPE
Switch	100 - 240 AC / 2 A	INESC	STARTECH

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Gateway Computer	100 - 240 AC / 2 A	INESC	ODROID-H2+
Acoustic Modem 1	240 AC	UdG	Blueprint SeaTrack X150
Acoustic Modem 2	240 AC	INESC	Evologics S2CR 18/34
Satellite Link	240 AC	SPACEAPPS	Thalys VesseLINK
Power Distribution Sys	100 - 240 AC	INESC	Battery
Power Cables	-	INESC	-

Table 9. SCC Communication endpoints:

Item	Power	Procured By	Product
Switch	240 AC	INESC	STARTECH
P2P Access Point	PoE	INESC	Mikrotik CPE
Local Servers	240 AC	INESC	TBC

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

3. Interoperability

For communicating with the robots, we have chosen the most common type of interface which is using Robot Operating System (ROS) messages. In a set up that implies long distance communication it is important to have control over the data transmission by imposing a set of rules that are known as quality of service (QoS) specifications. By using a QoS specification we will be able to make sure that specific messages arrive at the destination, signal the operators related to any kind of failure in communication, control the number of retries that the system does or select different based protocols to be used depending on the specific scenario. One of the most common communication protocols that is being used in the industry is the data distribution service (DDS). Since 2018 the community behind ROS have decided to make DDS the main middleware that is used for exchanging information between robots. The new version of ROS that is built on top of DDS it's called ROS2 and the stable version of it is called foxy. We have decided to build the SCC compatible with the ROS 2 common messages. The vehicle interface computer is running an instance of ROS2 Foxy and a software module called Data Gateway that is connected to the ROS2 network containing the robots and to the network containing the SCC back end services and database. The main functionalities are off the data gateway are:

- Signal the system when a new robot requires connection;
- Listen to the data data channels that are used by the robot to publish its internal state;
- Send data to the SCC in a recognizable, compatible form;
- Parse the robot configuration and send it to the SCC back end
- handle connectivity interruptions
- Forward the commands received from the user and reply with the outcome of the commands
- Forward to the robots the mission plans and signal the SCC with the progress of the plans and their outcomes.

The main constraints of the data gateway are that it has to be in both networks and it is compatible with the ROS software. As its main purpose is to communicate data from the robots, which are using ROS messages it uses a similar communication strategy to pass the data, namely socket.IO.

3.1. Abstract design

From a software perspective, the proposed design follows the layer structure described. At the lowest level sensory devices are responsible for the data acquisition. As this data is the core input for the whole operation, there is a path of communication all the way to the Server Infrastructure where the data is (1) stored, (2) analysed and (3) displayed on the operator views. There are two possible end points for the sensors: the testbed which can support fixed sensors continuously gathering environment characteristics and the robots where we have mobile sensors that gather on demand information related to a specific component of the testbed.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

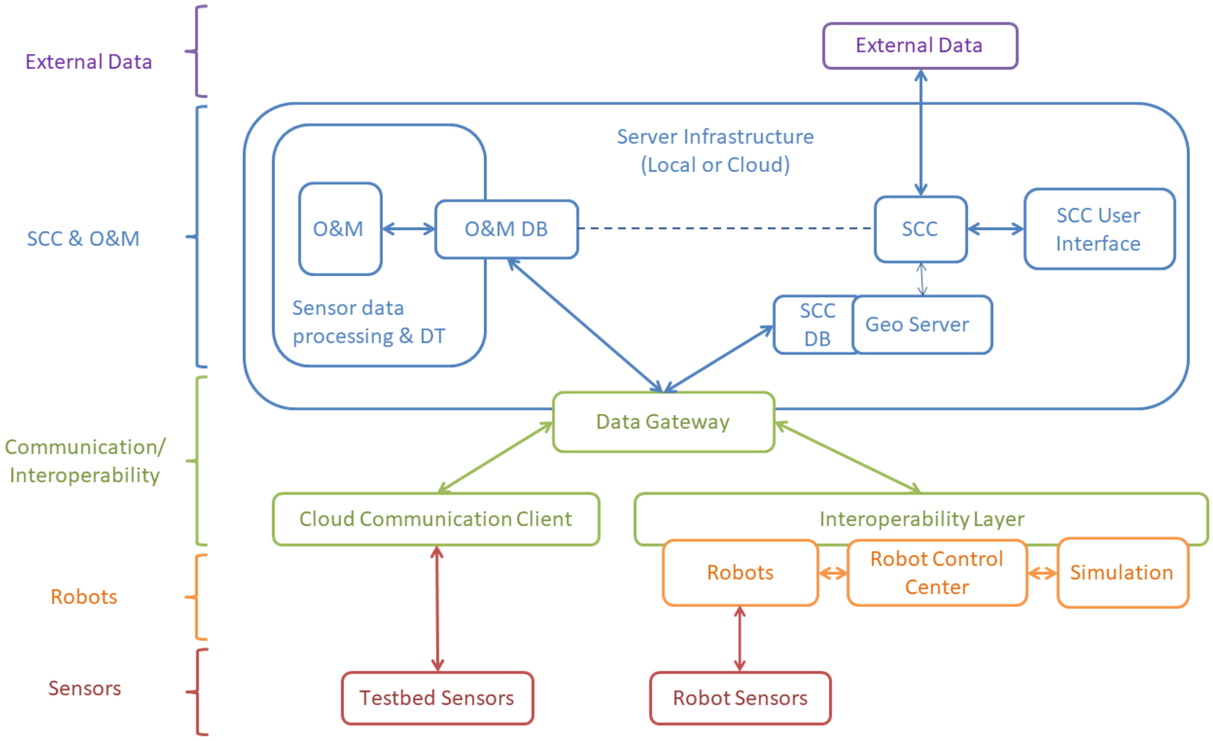


Figure 14. Atlantis Software Architecture

The robots (in simulation or deployment), together with their control units, are performing the IMR actions in the Pilot. They get, pre-process and send away the data from the sensors, advertise their state and receive instructions from the supervisory control centre. While each of the robotic devices have its own on-board sensors with a particular software interface, the data exchange with the control centre is done in a unified way from all the robots. Communicating using a similar convention will be possible due to an interoperability software that has the role to translate the commands and data from the SCC format to the robot specific format. Most of the robotic devices are using the Robot Operating System (ROS) middleware to exchange data on-board and with the robot control centres.

The communication software layer that links the Pilot with the server infrastructure from the shore is composed of:

- Interoperability Layer;
- Cloud Communication Client;
- Data Gateway.

Using the DDS protocol: DDS is an OMG communication standard represented as a middleware providing publish-subscribe communications with robust characteristics. DDS features control of QoS parameters, including reliability, bandwidth, delivery deadlines, and resource limits. The DDS standard enables its implementation by different vendors to remain interoperable while providing the flexibility for defining data specific structures to be communicated between software components.

The SCC can receive and store the data from the robots in its own database which is accessible from the user interface. Data can be telemetries, images, point clouds, positions etc. Since numerical telemetries

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

are represented in graphs or tables and geographical data are shown on a map, the user can follow the position of the robots as well as their sensor data in real time or by accessing the history. Through this interface, the user can also define an objective for an action or a mission and send it to the desired robots. An automated planning feature is embedded in the SCC. It can be used to compute a plan for the robot(s) involved in the mission using information stored in the database (such as initial position and robot characteristics) and input from the operator. A plan is composed of actions (possibly timestamped) aiming to satisfy the mission goal and a list of points to visit. Feedback and results of these actions / missions are received by the SCC and are displayed on the UI.

The O&M module receives and stores inspection data from the robots in a MIMOSA OSA-CBM compliant database. It further processes the inspection data in a pipeline to operation and maintenance advisories that are sent to the SCC for displaying. An ISO 13374 compliant processing pipeline is considered but adapted as needed. Image and video acquisitions are stored and processed in MIMOSA OSA-CBM DABLOB Data objects, other data acquisitions are stored in their complying data objects. In addition to data from robots, O&M may also receive external data from infrastructure sources on the Coastal Testbed and Offshore Testbed.

3.2. Communication Protocols and Middleware

3.2.1. Communication Protocol Selection: DDS

The communication protocol that is used within the interoperability layer for onshore to offshore communication is Data Distribution Service (DDS). DDS is a messaging exchange protocol developed by the industry to communicate within a distributed system having strict operations requirements. Using this protocol will allow us to strictly control the delivery of data from one side to the other. DDS is designed as a publish-subscribe system in which any component can broadcast data for any of the other components to listen to. When the data is broadcast, parameters can be set to configure the messaging system for management of message delays, delivery failure, communication latency etc. DDS existed in the Industrial ecosystem for a long time. In recent years there was a high interest presented by the robotics communities to integrate the DDS standard as the main message exchange modality in the robotics system. The choice is based on the long history of DDS being successfully used in safety-critical systems. Due to this increased interest, the second generation of the Robot Operating System (the most used middleware for robotics projects) is built on top of the DDS protocol. In ATLANTIS , most of the robots support ROS natively, so the main way of message exchange between the robots and the interoperability layer will be done through it.

3.2.2. Middleware Selection: ROS2

The Robot Operating System, ROS is used to help robots from different manufacturers to communicate together. ROS provides standardised message formats for basic data types, such as strings or numbers, and more complex data types such as poses, images or point clouds. ROS is based on a fully distributed ecosystem allowing all the robots being part of the same network to communicate. The second version of the robotic operating system, ROS2, is using DDS by default on the communication layer allowing it to benefit from all the advantages of DDS using the features offered by ROS.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Since the ROS2 communication protocol is different from the one used in the first version of ROS, ROS2 also provides a bridge to allow ROS1 code to be compatible with ROS2 and DDS. To facilitate the development of the software of a robot, ROS offers client libraries in different languages (C++, Python, JavaScript, …) and multiple tools to examine the network and the message exchanges:

- ROSBag: record and replay data exchange
- CLI: Command line interface to subscribe to messages, publish messages, analyse traffic …
- RVIZ: 3D Visualisation tools.

3.3. Data Flows

The Interoperability Layer has the purpose of connecting a set of heterogeneous robots to a single system in which the devices are aware of each other and that can be observed and controlled from a single point of operations. The main role of the layer is to expose the deployed assets using a common language that can be understood by other parts of the systems like the supervisory control centre, the sensor data processing units and even by other robots. The functionality of the layer includes one part that focuses on the interaction unification and the other that focuses on the operations requirements.

Features

The main functional features supported by the interoperability layer are:

- Registration
- Discovery
- State Monitoring
- Commanding
- Data retrieval
- Mission based operations
- Cross robot interaction

Architecture

The data **gateway communication** component is deployed on the onshore computers. The role of the Gateway is to convert DDS messages to protocols used by the other SCC modules and vice versa. The conversion will be done according to each of the module specifications and can be in the form of http or web socket json payloads, files or binary data. The data is then passed from the gateway to the storing services and, if configured in such a way, to the live data providers.

Robot interface and Capabilities Description. Each of the robots will register to the SCC by publishing their available and supported data types. The configuration will be stored in a file (e.g. YAML) on the processing unit that runs the interoperability layer robot interface (could be on-board or on the robot specific control centre).

Given that we are dealing with devices with different characteristics and capabilities, the generic configuration structure will be subject to change during the development phases of the project. Configuration will imply a set of required fields that help for the device identification, position and generic mission deployment. These elements will be mandatory for all the configuration, so the other systems can have basic interaction. The rest of the configuration can be specific, with certain limitations.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

The options from the configuration advertises to the interoperability layer and the SCC which are the types of interactions available. The two can be split into available telemetry and available commands. As an effect to the SCC, depending on the configuration, the operator interface will adapt to display certain data types and make available specific actions that will be labelled as defined in the configuration files.

The proposed list of configuration elements can be split into required and optional elements. The required information that a robot provider has to configure is:

- **Namespace**: The name of the robot. It has to be unique and it will have the role to help the SCC to identify it. The operators will see this string associated with the robot.
- **Pose:** The position of the robot. It will be done by calling the interoperability layer or by adding the name of a topic in the configuration.
- **Mission Execution:** Routine to execute mission. If using ROS it will be an *action server* capable of receiving a plan as a goal, otherwise it will be a registered callback.

For each robotic device, the robot owner can register to the interoperability layer data types that can be sent and commands that are supported. The number of interfaces is not limited. The supported types of telemetry data is boolean or numeric and the commands can have four types of payloads: no payload, point, path or period. Table 10 represents a possible configuration file for an unmanned vehicle.

Table 10. Example Configuration AUV

```
### ROS Namespace
namespace: /robot_namespace
### Telemetry (for ROS: name of topics)
boolean_telemetry_topics:
 - /submerged
 - /in_flight
 - /WiFi_on
 - /docked
 - /lidar_on
numeric_telemetry_topics:
 - /battery
 - /altitude
 - /motor1/current
pose_topic: /my_robot_pose
### Commands (for ROS: name of action servers)
mission_execution: /my_plan_execution_srv
command_plain: #no parameters
 - /publish_all_images:
   conditions:
     - WiFi_on
 - /recharge
   conditions:
     - docked
 - dock
command_waypoint: #point payload
 - /fly_to_point
 - /land_to_point
 - /submerge_to_point
 - /survey_point
command_path: #path payload
```

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

```
 - /my_robot_follow_path
 - /scan_path:
   conditions:
     - /submerged
     - /lidar_on
command_time:
 - dock_until_timestamp:  #time payload
```

The optional fields in the config file are:

- **Boolean and Numeric telemetry topics:** List of all the ROS topics where the robot sends boolean / numeric data.
- **Command plain:** List of all the ROS commands that do not require any payload. A condition can be added to allow or not the operator to execute the command. The condition is based on the value of a boolean telemetry topic.
- **Command waypoint / path / time:** Similar as command plain but for commands requiring payloads. The waypoint, path and time commands should receive points, path path and timestamp respectively.

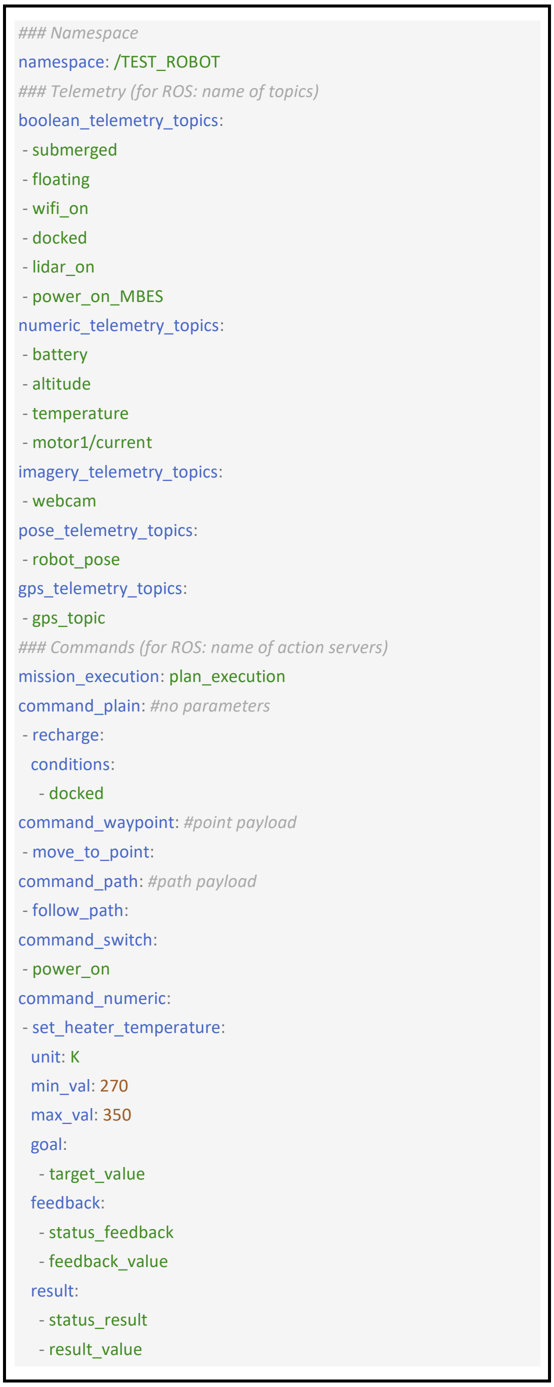
3.3.1. Robot Configuration Structure

The configuration will be sent to the Control Centre by each robot following the specifications provided. The recommended way to define the robot capabilities is by writing YAML file (see table 11) that includes:

- A unique string identifier of the robot called namespace
- An authentication token to authentify the robot to the SCC.
- Telemetry topics for data:
  - Boolean
  - Numerical
  - Imagery
  - Position and orientation
  - GPS position
- A list of video streams links.
- An identifier for the algorithm responsible for executing the mission plan
- List of commands based on their payload types
  - Plain (no payload)
  - Waypoint bath
  - Switch
  - Numeric

An example of such a file can be observed in the table cell below. Space apps has created a software module for integrating the robots which will read and transmit the information contained in the robot configuration file to the SCC. The information contained in this configuration will be absorbed by the Control Centre, stored in the database, associated with the vehicle and used for the configuration of the user interface. Depending on what kind of information is introduced in the configuration, the user will have specific views for the data monitoring and specific controls for sending commands.

Table 11. Example Configuration for Interoperability



```yaml
### Namespace
namespace: /TEST_ROBOT
### Telemetry (for ROS: name of topics)
boolean_telemetry_topics:
 - submerged
 - floating
 - wifi_on
 - docked
 - lidar_on
 - power_on_MBES
numeric_telemetry_topics:
 - battery
 - altitude
 - temperature
 - motor1/current
imagery_telemetry_topics:
 - webcam
pose_telemetry_topics:
 - robot_pose
gps_telemetry_topics:
 - gps_topic
### Commands (for ROS: name of action servers)
mission_execution: plan_execution
command_plain: #no parameters
 - recharge:
   conditions:
    - docked
command_waypoint: #point payload
 - move_to_point:
command_path: #path payload
 - follow_path:
command_switch:
 - power_on
command_numeric:
 - set_heater_temperature:
   unit: K
   min_val: 270
   max_val: 350
   goal:
    - target_value
   feedback:
    - status_feedback
    - feedback_value
   result:
    - status_result
    - result_value
```

3.3.2. Robot Discovery and Approval

One of the most important aspects of the system is to be able to establish and maintain communication with robots and the environment for which the communication is needed.

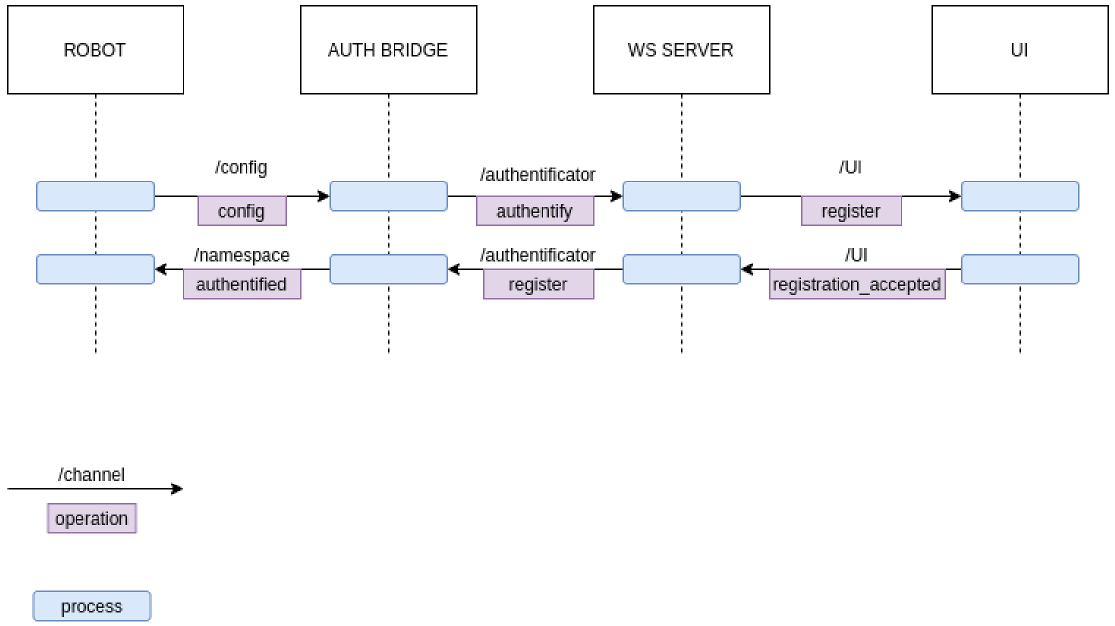


Figure 15. Robot Discovery Sequence of Operations

To communicate with robots the SCC needs to know the capabilities and identifiers for each one of them. In addition, to allow robots to communicate with other robots the same information is needed. To avoid manually registering this information in the SCC allows other robots to receive it, the control centre introduces an automatic discovery process allowing it to detect all the robots running on the same network and get all the communication information relative to the entity.

Every robot present on the network should continuously publish its configuration until it is discovered and registered in the SCC. The configuration contains information needed to communicate with the entity: essentially the namespace (identifier), the list of actions that the robot can execute and associated commands and the list of telemetries. The control centre automatically enables communication with new robots on the network by creating a ROS bridge based on the robot configuration content. The same process can also be used by any robot to be notified or aware of other devices that may connect (or re-connect) at differing moments in time to the network and directly communicate with them (see figure 15). On the control centre side, when the communication is established with a robot, all data received is then stored in the database for persisting data and sent through websocket to every operator monitoring station subscribing to real time robot data.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

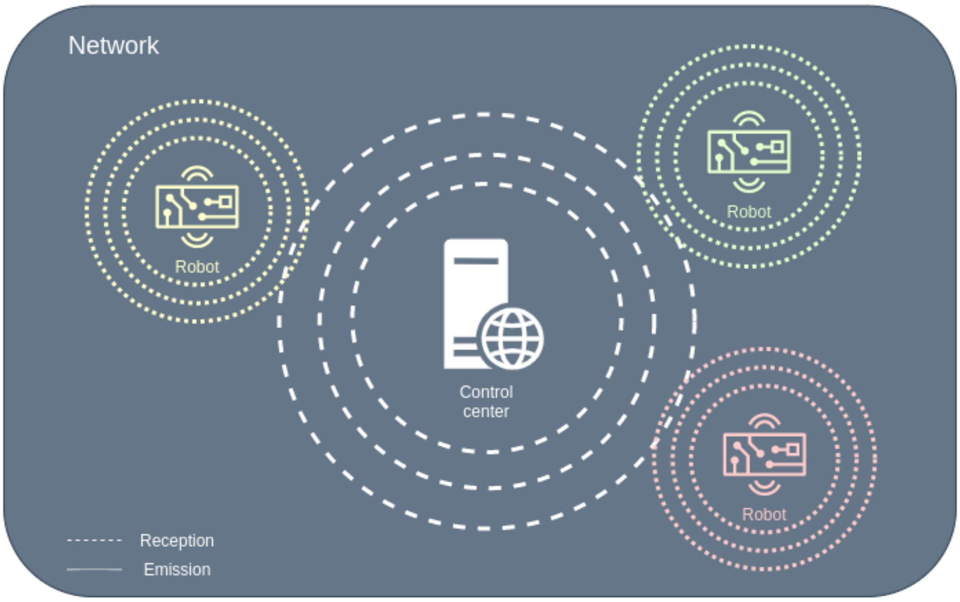


Figure 16. Robot discovery: Control centre subscription

The user interface can be used by the operator or SCC supervisor for the automatic discovery process. When a new robot on the network is detected by the control centre it can ask the user if the new robot should be authorised or not (see figure 17). If the user rejects the robot registration, the robot will not be able to communicate with the control centre, it is equivalent to having the robot outside of the network.

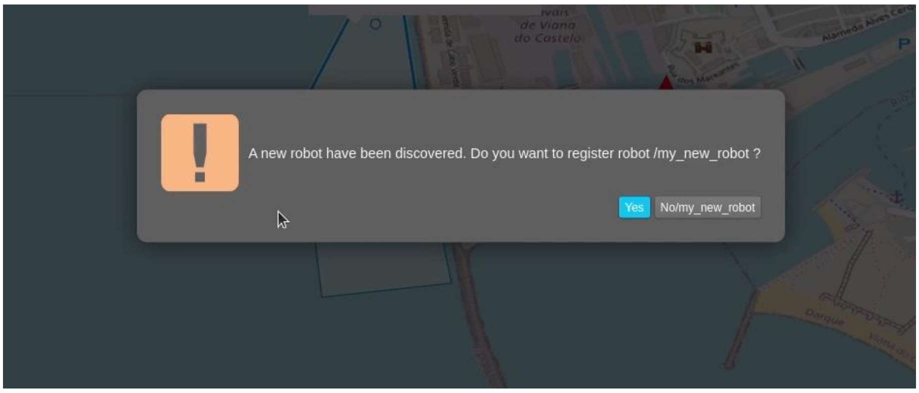

A new robot have been discovered. Do you want to register robot /my_new_robot ?

Figure 17. New robot registration UI supervision

3.3.3. Situational Awareness (Monitoring)

To allow the SCC and the rest of the connected robots to receive and understand the data coming from the vehicles, we are using standard and custom ROS messages. These data can be published at various rates and the publishing does not imply any acknowledgement from a subscriber.

The following standard messages are used:

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

- **geometry_msgs/Pose:** Position and orientation of the robot in a local map.
- **sensor_msgs/NavSatFix:** GPS coordinates (longitude, latitude, altitude) of the robot.
- **sensor_msgs/Image:** Image coming from robot camera.
- **sensor_msgs/PointCloud:** Point clouds coming from robot sensors.

To extend the different data types that can be received by the SCC we have created the following custom ROS messages:

- **BoolTimestamped:** Boolean telemetry associated with timestamp.
- **FloatTimestamped:** Numerical telemetry associated with timestamp.

The interoperability network will handle the data coming and going from multiple robots and will not host any kind of closed loop control. To protect the network from clogging and in the same time to remain informative towards the operators receiving the data on the SCC side, the guideline in publishing rates are:

- Numerical data (including poses): 1 - 4 Hz
- Images: 4 - 5 Hz, low resolution
- Point clouds: less than 1Hz.

3.3.4. Commanding

The commanding mechanism is based on the ROS action service. The service is based on a state machine (Figure 18) that allows the asynchronous trigger based on a goal, constant feedback, publishing a result, monitoring failures and asynchronous cancelling.

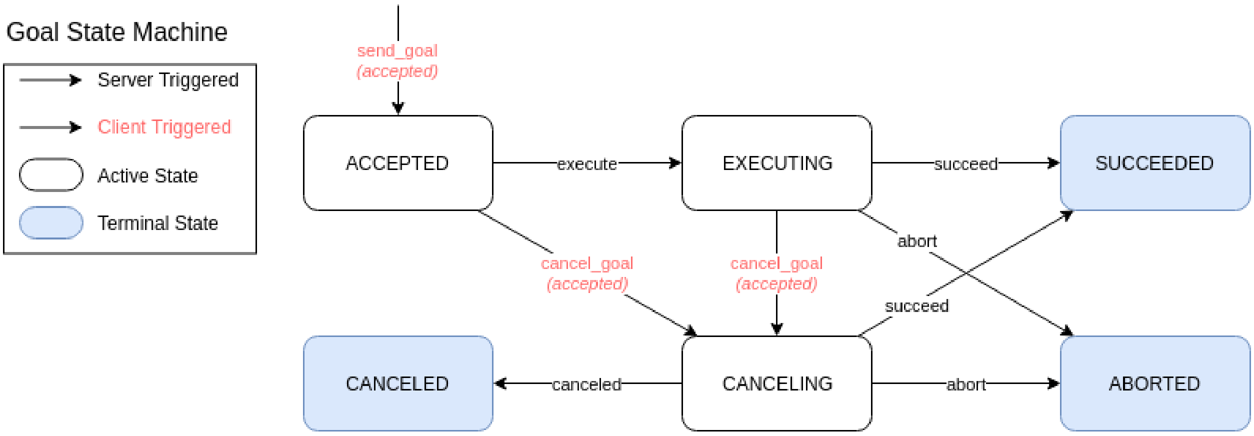


Figure 18. ROS action state machine (ROS2 documentation)

The SCC needs also to be able to send commands to the robots connected to the interoperability. Only custom actions are used for that purpose, all accepting different goal types. Every command can send an integer as a feedback and result to indicate the status of the action as shown in the list of commands bellow:

- **CommandBool**

Table 12. Command Boolean Action Message Structure

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Goal	Feedback	Result
Bool	Int	Int

- **CommandNumeric**

Table 13. Command Numeric Action Message Structure

Goal	Feedback	Result
Float	Int	Int

- **CommandPath**

Table 14. Command Path Action Message Structure

Goal	Feedback	Result
nav_msgs/Path	Int	Int

- **CommandPlain**

Table 15. Command Plain (no payload) Action Message Structure

Goal	Feedback	Result
/	Int	Int

- **CommandWaypoint**

Table 16. Command Waypoint Action Message Structure

Goal	Feedback	Result
geometry_msgs/Pose	Int	Int

3.3.5. Autonomous Operations

To send a mission plan that the robots can execute we are using custom messages to convert a mission plan planned on the SCC into a list of actions that the robot can execute. Every action of the plan is represented by a **Step** message which is composed of:

- **Index:** Index of the step starting from 0.
- **Name:** Name of the action to be executed.
- **JSON data:** Represents the parameters needed for the action to be executed in a JSON format.

The json data should look like the following:

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

{"name_parameter_1": value_param_1, "name_parameter_2": value_parameter_2}

This allows us to send different types of data for the parameters using the same ROS message. In order to execute a plan, an **ExecutePlan** ROS action message has been created. The ExecutePlan action is defined as follow:

- **Goal:** List of Step ROS messages, represents the list of actions to execute to perform the plan.
- **Feedback:** The feedback message is composed of:
    - A list of the steps already executed. The CompletedStep message type is used here, containing the Step and the start and end time of its execution.
    - The list of steps in progress (Step message).
    - The list of remaining steps (Step message).
- **Result:** List of CompletedSteps.

On Table 17 a plan example of an AUV based on a mission for Scenario 2 is detailed. For every step the parameters to execute the action are given. As the example plan should be executed by an AUV all the actions should receive the namespace of the AUV as a parameter. The "/AUV" namespace is used in the example.

Here is a description of the different actions of the plan in Table 17:

- **Submerge:** Submerge the robot to fix depth.
- **Dive:** Dive the robot to a given target point.
- **Activate / Deactivate sensor:** Activate / deactivate a sensor based on its name.
- **Activate / Deactivate comm:** Activate / deactivate a communication device based on its name.
- **Asses:** Evaluate a target at a given position before inspection using the given sensor.
- **Inspect:** Inspect target using given sensor.
- **Send data:** Send data collected during inspection of the target with a given sensor using the communication device given as parameter.

Table 17. Sample plan containing the actions for an AUV to execute for achieving a goal in Scenario 2.

Step	Action	Params

0	submerge	{
		"robot": "/AUV"
		}

1	dive	{
		"robot": "/AUV",
		"start_point": [-8.840203550001069, 41.68788297257113],
		"target_point": [8.838826631079701, 41.68641999621718],
		}

2	activate_sensor	{
		"robot": "/AUV",
		"sensor": "camera"
		}

3	activate_comm	{

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

		"robot": "/AUV", "communication_device": "wifi" }

4	asses	{
		"robot": "/AUV",
		"point": [-8.838826631079701 41.68641999621718],
		"target": "mooring",
		"sensor": "camera"
		}

5	inspect	{
		"robot": "/AUV",
		"point": [-8.838826631079701, 41.68641999621718],
		"target": "mooring",
		"sensor": "camera"
		}

6	send_data	{
		"robot": "/AUV",
		"target": "mooring",
		"sensor": "camera",
		"communication_device": "wifi"
		}

7	deactivate_comm	{
		"robot": "/AUV",
		"communication_device": "wifi" }

8	deactivate_sensor	{
		"robot": "/AUV",
		"sensor": "camera" }

9	dive	{
		"robot": "/AUV",
		"start_point": [-8.838826631079701, 41.68641999621718],
		"target_point": [-8.840117492568483, 41.68692851740973]
		}

10	activate_sensor	{
		"robot": "/AUV",
		"sensor": "camera"
		}

11	activate_comm	{
		"robot": "/AUV",
		"communication_device": "wifi"
		}

12	asses	{ "robot": "/AUV",

		"point": [-8.840117492568483, 41.68692851740973], "target": "foundation", "sensor": "camera" }

13	inspect	{
		"robot": "/AUV",
		"point": [-8.840117492568483, 41.68692851740973],
		"target": "foundation",
		"sensor": "camera"
		}

14	send_data	{
		"root": "/AUV",
		"target": "foundation",
		"sensor": "camera",
		"communication_device": "wifi"
		}

15	deactivate_comm	{
		"robot": "/AUV",
		"communication_device": "wifi"
		}

16	deactivate_sensor	{
		"robot": "/AUV",
		"sensor": "camera"
		}

17	dive	{
		"robot": "/AUV",
		"start_point": [-8.840117492568483, 41.68692851740973],
		"target_point": [-8.840000141524049, 41.68571588995057]
		}

3.4. Unmanned Vehicle Interoperability Library

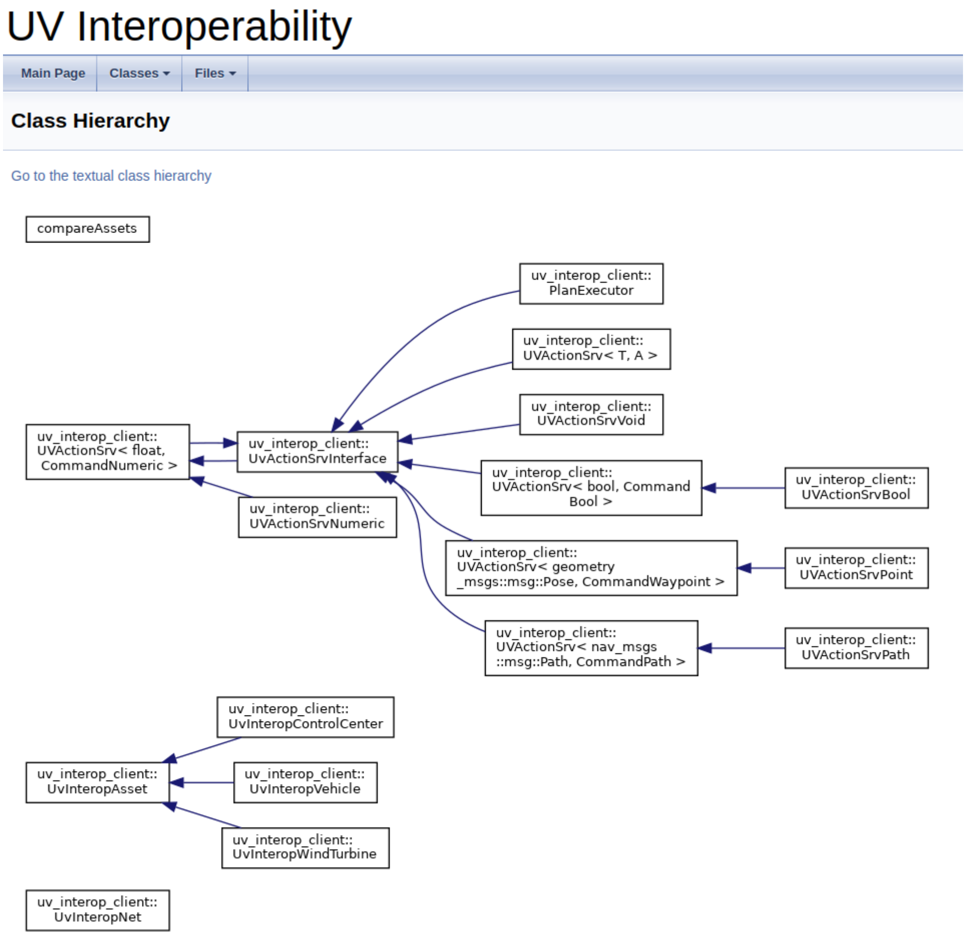
The interoperability specification was implemented in a cpp library and distributed to all the partners involved in the technical development of the project. The implementation of the library used ROS2 and DDS to transmit the data in the network. For using the library to interface a hardware the following dependencies have to be met:

Table 18. Interoperability Dependency List

Operating System	Ubuntu 20.04 (preferred) Ubuntu 18.04 Windows 10

Local Dependencies	ROS 2 Foxy

The project is designed using object oriented programming principles and makes use of the publish - subscribe pattern implemented in the ROS2 middleware. The high level class hierarchy of this library is presented in figure 19. The main class containing the monitoring features is Asset. For vehicle specific interactions the class asset is extended with commanding and plan execution capabilities in a child class named Vehicle. Using the library to connect a simple robot implies creating the vehicle object, running the connect function with the config file path as a parameter. This will start the publication of the config at a rate of 1Hz.



UV Interoperability

Main Page
Classes ▾
Files ▾

Class Hierarchy

Go to the textual class hierarchy

compareAssets
uv_interop_client:: PlanExecutor
uv_interop_client:: UVActionSrv< T, A >
uv_interop_client:: UVActionSrvVoid
uv_interop_client:: UVActionSrv< float, CommandNumeric >
uv_interop_client:: UvActionSrvInterface
uv_interop_client:: UVActionSrvNumeric
uv_interop_client:: UVActionSrv< bool, Command Bool >
uv_interop_client:: UVActionSrvBool
uv_interop_client:: UVActionSrv< geometry _msgs::msg::Pose, CommandWaypoint >
uv_interop_client:: UVActionSrvPoint
uv_interop_client:: UVActionSrv< nav_msgs ::msg::Path, CommandPath >
uv_interop_client:: UVActionSrvPath
uv_interop_client:: UvInteropControlCenter
uv_interop_client:: UvInteropAsset
uv_interop_client:: UvInteropVehicle
uv_interop_client:: UvInteropWindTurbine
uv_interop_client:: UvInteropNet

Figure 19. Class Hierarchy of UV Interoperability Library

The asset interface serves as a base class for connecting, authentication and publishing telemetry for one of the assets being part of the Atlantis testbed:

- Vehicle
- Control Centre
- Wind Turbine

The relationships between the classes used for connecting different types of assets are shown in figure 20.

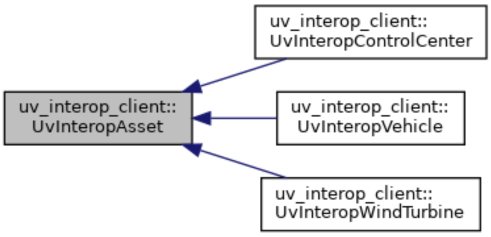


Figure 20. UVInteropAsset Hierarchical Relationships

UV Interoperability was documented using code comments compatible with Doxygen. The reference of the interface including their methods, functions and description for the Asset and Vehicle classes is listed in table 18 and table 19

Table 19. Reference of class Asset

void	**release_all_assets ()**
	Release all the assets
bool	**connect (std::string config_file_path)**
	Connect the asset to the interop network by publishing the given config
bool	**connect ()**
	Connect the asset to the interop network by publishing the a config containing only the asset namespace
bool	**disconnect ()**
	Disconnect the asset from the interop network
bool	**get_connection_health ()**
	Get the connection health object.
bool	**pub_image (std::string topic, std::string file_path)**
	publish pictures data from file; Supported formats: *.png, *.jpg, *.jpeg, *.tiff.

bool	**pub_pointcloud (std::string topic, std::string file_path)**
	publish pointcloud data from file; Supported formats: *.laz, *.las.
template<typename T >	
bool	**publish_data (std::string topic, T payload, std::time_t timestamp=0)**
	publish numerical data.
bool	**pub_stream (std::string topic, std::string stream_URL)**
	publish a data stream (i.e. video).
bool	**send_emergency_stop (std::string ns, int32_t timeout_ms)**
	send an emergency stop to a vehicle.

Table 20. Reference of class Vehicle

template<typename T >	
bool	**register_command (std::string command_name, bool(*command_callback)(T cmd_payload), bool(*cancel_callback)())**
	Register command with given name to execute callback.
bool	**register_command (std::string command_name, bool(*command_callback)(), bool(*cancel_callback)())**
	Register command with given name to execute callback without payload.
bool	**register_plan_executor (std::vector< c4i_asset_msgs::msg::CompletedStep >(*plan_callback)(std::vector< c4i_asset_msgs::msg::Step > steps), bool(*cancel_callback)())**
	Register plan executor to execute command a mission plan.
void	**publish_plan_feedback (std::vector< c4i_asset_msgs::msg::CompletedStep > executed_steps, std::vector< c4i_asset_msgs::msg::Step > step_in_progress, std::vector< c4i_asset_msgs::msg::Step > remaining_steps)**

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

	Publish feedback during execution of a plan.
void	**abort_command (std::string command_name)**
	Abort the given command.
void	**abort_plan_executor (CompletedSteps executed_steps)**
	Abort the given plan execution.
template<typename T >	
bool	**register_emergency_stop (std::string command_name, bool(*command_callback)(T cmd_payload))**
	Register emergency stop command.
bool	**send_system_fault (ErrorMsg err)**
	Send system faults.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

4. Integration Results

During the last months of the task the focus fell on the integration test and validation of the interoperability library. The code and documentation were distributed to all the partners. Each of the partners providing an unmanned vehicle have installed the UV Interoperability on the corresponding machines and developed integration code. Along with the interoperability library, a series of examples have been provided to serve as a guide and facilitate the integration process.

SPACEAPPS have organised the first integration workshop with all the partners in July 2021. During the workshop the following tasks were successfully achieved:

- Installation of the interoperability on partners machines
- Successful launch of a test robot instance
- Deployment of the data gateway node
- Deployment of a preliminary version of the SCC
- Successful discovery of a test robot running on the host computers of each partner.

The remote tests were performed using a VPN setup. The partners present in the workshop were: INESC, IQUA, ECA, UdG and VTT.

After the workshop the integration started at each partner. The communication platform for tracking issues was a gitlab server set up by SPACEAPPS. Each partner defined the list of data categories supported for monitoring. Once the config file was done, the library was integrated with the robot using either simulation of the actual hardware. In the next section each one of the integrated platforms are presented. The data interfaces defined are extracted from the config files that are defined by the partners and the integration results include the types of tests that were successfully done.

Each of the data exchanges from the tables added to the “Data Interfaces” chapters are following the numbering formalism defined in D2.2 . Each of the subsystems have assigned a letter and each of the interfaces has the structure SW_<from><to>_number . The literals for each of the subsystems are:

- **A** - Interoperability
- **B** - AUV
- **C** - ROV
- **D** - ASV
- **E** - UAV
- **F** - SCC

4.1. AUV (Sparus, Girona)

4.1.1. Data Interfaces

For the AUV, the type of commands that can be passed as part of an inspection plan the

Table 21. Data Structures passed from Interoperability to AUV

From Interoperability Layer (Commands)

Channel	From	To	Partner
WiFi/Acoustic	Interop	AUV	UdG / IQUA
Ref	**Title**		
SW_CD_015	submerge		
Short description: Submerge the robot to a fixed depth. *Parameters:* ● NA			
	dive		

Short description: Move from point A to point B.
Parameters:
● Start_point : expected start point of the robot
● Target_point: destination of the robot
Example step structure:
{
"start_point": [-8.840203550001069, 41.68788297257113],
"target_point": [8.838826631079701, 41.68641999621718]
}			

	activate_sensor		

Short description: command to activate one of the onboard sensors
Parameters:
● Sensor: name of the sensor to be activated
Example step structure:
{
"sensor": "camera"
}			

	activate_comm		

Short description: command to activate one of the communication endpoints onboard the robot
Parameters:
● Communication_device: name or identifier of the communication device
Example step structure:
{
"communication_device": "wifi"
}			

	asses		

Short description: Evaluate a target at a given position before inspection using a specified sensor.
Parameters:
● Communication_device: name or identifier of the communication device
Example step structure:
{
"point": [-8.838826631079701 41.68641999621718],
"target": "mooring",
"sensor": "camera"
}			

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

inspect

Short description: inspect a target using an onboard sensor

Parameters:

- Point: the location of the target.
- Target: the name of the target.

Example step structure:

```
{
  "point": [-8.838826631079701, 41.68641999621718],
  "target": "mooring",
  "sensor": "camera"
}
```

send_data

Short description: upload survey data associated with a target to the SCC.

Parameters:

- Target: the name of the inspected target.
- Sensor: the name of the target.
- Commnication_device: name or identifier of the communication device

Example step structure:

```
{
  "target":                                                                    "mooring",
  "sensor":                                                                     "camera",
  "communication_device":                                                           "wifi"
}
```

deactivate_comm

Short description: command to deactivate one of the communication endpoints onboard the robot

Parameters:

- Communication_device: name or identifier of the communication device

Example step structure:

```
{
  "communication_device": "wifi"
}
```

deactivate_sensor

Short description: command to deactivate one of the onboard sensors

Parameters:

- Sensor: name of the sensor to be deactivated

Example step structure:

```
{
  "sensor": "camera"
}
```

Command plain

- Assessment
- Inspection
- Publish_data

Table 22. Data Structures passed from AUV to Interoperability

To Interoperability Layer (Data & TM)			
Channel	**From**	**To**	**Partner**
WiFi/Acoustic	Interop	AUV	UdG / IQUA
Ref	**Title**		
SW_DC_029	Numerical Telemetry		
- Elapsed_time - Altitude - Heading - Current_waypoint			
SW_DC_030	Boolean Telemetry		
- High_temperature - Water_inside - No_altitude_error - Navigation_error - Low_battery_warning - Low_battery_error - Watchdog - Emergency_surface - Abort_surface - Abort - Submerged - Autonomous_mode - Teleop_mode - Docked - Sonar_active			
SW_DC_031	GPS Telemetry		
- USBL_pose_navsatfix			
SW_DC_032	Pose Telemetry		
- Robot_pose - USBL_pose			

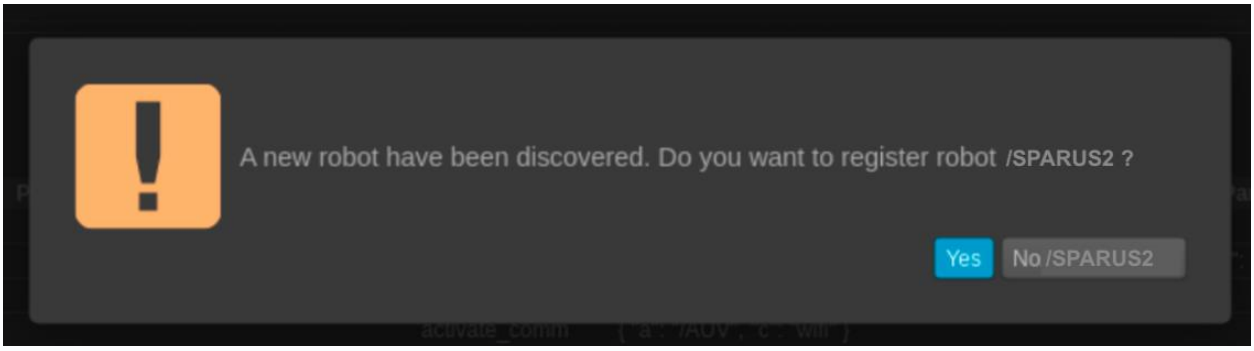
4.1.2. Integration Results

The AUV was integrated with the SCC using the interop. The following has been tested:

- **VPN connection:** SCC and robot connected to the same network using OpenVPN connection.
- **Discovery and identification of the Robot:** Robot automatically discovered by the SCC by receiving the robot config.
- **Boolean and Numerical Telemetry:** Reception and visualisation of the robot telemetries.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

- **Poses:** Reception and visualisation of the pose (position and orientation) of the robot in real time on the SCC.



A new robot have been discovered. Do you want to register robot /SPARUS2 ?

Figure 21. Discovery alert for the SPARUS2

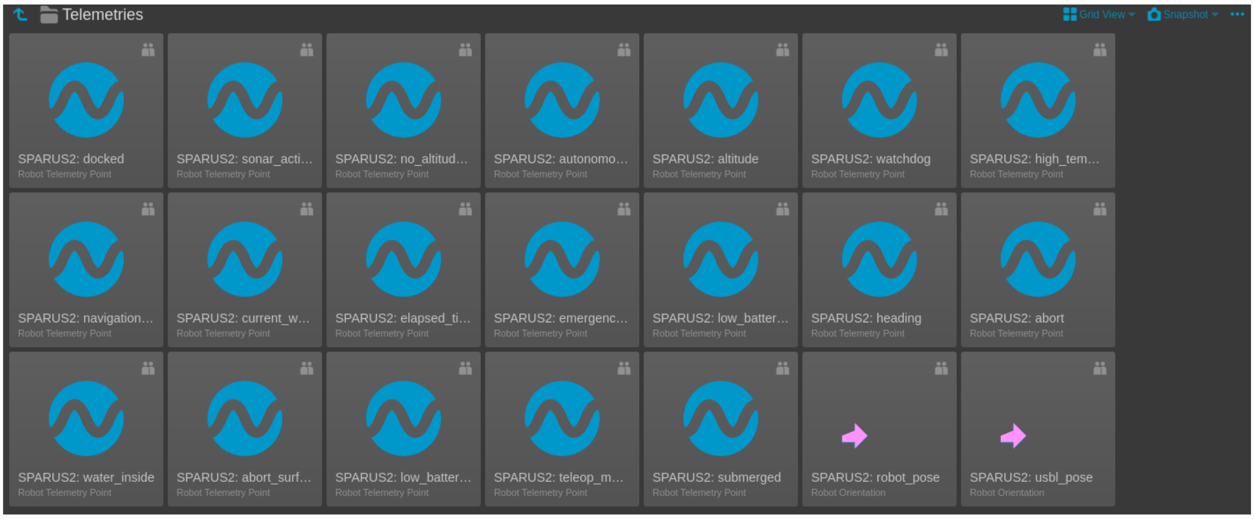


Figure 22. Telemetry topics received from SPARUS2

4.2. ROV (ECA ROV)

The underwater remote operated vehicle from ECA called ROVING BAT software stack has been tested with the interoperability library.

4.2.1. Data Interfaces

The data interfaces were designed using the yml config structure defined from the interoperability.

Table 23. Data Structures passed from Interoperability to ROV

From Interoperability Layer (Commands)			
Channel	**From**	**To**	**Partner**
Ethernet	Interop	ROV	ECA
Ref	**Title**		

	Command Plain
	- ComState - RovState
	Command Numeric
- X - Y - Z - Roll - Pitch - Yaw - ArmX - ArmY - ArmZ - ArmRX - ArmRY - ArmRZ - CamSelec - LightSelec - PanCam - TiltCam	
	Command Bool
- AutoDepth - AutoHeading - LightsPower - CamRecord - CleaningToolPower	

The ROVINGBAT is capable of publishing all the robot operation information to the SCC. This includes numerical, boolean, positional and video streams as shown in Table 24.

Table 24. Data Structures passed from ROV to Interoperability

To Interoperability Layer (Data & TM)			
Channel	**From**	**To**	**Partner**
Ethernet	ROV	Interop	ECA
Ref	**Title**		
SW_DC_029	Numerical Telemetry		
- depth - Eca/Vfr_Hud/airspeed - Eca/Vfr_Hud/altitude - Eca/Vfr_Hud/climb - Eca/Vfr_Hud/groundspeed - Eca/Vfr_Hud/heading		- Eca/Battery/current_consumed - Eca/IMU/temperature - Eca/IMU/xgyro - Eca/IMU/ygyro - Eca/IMU/zgyro - Eca/IMU/xaccel	- Eca/Thruster3/current - Eca/Thruster4/current - Eca/Thruster5/current - Eca/Thruster6/current - Eca/Thruster7/current - Eca/Thruster0/Cmd

- Eca/Vfr_Hud/throttle - Eca/Eca_Telemetrie/output_thrusters_current - Eca/Eca_Telemetrie/input_current - Eca/Eca_Telemetrie/Humidity1 - Eca/Eca_Telemetrie/Humidity2 - Eca/Eca_Telemetrie/Humidity3 - Eca/Eca_Telemetrie/temperature1 - Eca/Eca_Telemetrie/input_voltage - Eca/Eca_Telemetrie/output_general_current - Eca/Eca_Telemetrie/output_thrusters_voltage - Eca/Eca_Telemetrie/output_general_voltage - Eca/Eca_Telemetrie/error_flags - Eca/Eca_Telemetrie/temperature2 - Eca/Eca_Telemetrie/temperature3 - Eca/Attitude/yaw_speed - Eca/Attitude/roll_speed - Eca/Attitude/pitch_speed - Eca/Attitude/roll - Eca/Attitude/pitch - Eca/Attitude/yaw - Eca/Scaled_pressure/Absolute_pressure - Eca/Battery/battery_function - Eca/Battery/current_battery - Eca/Battery/battery_remaining - Eca/Battery/type - Eca/Battery/time_remaining - Eca/Battery/temperature	- Eca/IMU/yaccel - Eca/IMU/zaccel - Eca/IMU/xmag - Eca/IMU/ymag - Eca/IMU/zmag - Eca/Rangefinder/Altitude - Eca/Scaled_pressure/temperature - Eca/Scaled_pressure/press_diff - Eca/Global_Position_Int/relative_alt - Eca/Global_Position_Int/hdg - Eca/Global_Position_Int/alt - Eca/Nav_Controller_Output/alt_error - Eca/Nav_Controller_Output/nav_bearing - Eca/Thruster0/voltage - Eca/Thruster1/voltage - Eca/Thruster2/voltage - Eca/Thruster3/voltage - Eca/Thruster4/voltage - Eca/Thruster5/voltage - Eca/Thruster6/voltage - Eca/Thruster7/voltage - Eca/Thruster0/velocity - Eca/Thruster1/velocity - Eca/Thruster2/velocity - Eca/Thruster3/velocity - Eca/Thruster4/velocity - Eca/Thruster5/velocity - Eca/Thruster6/velocity - Eca/Thruster7/velocity - Eca/Thruster0/current - Eca/Thruster1/current - Eca/Thruster2/current	- Eca/Thruster1/Cmd - Eca/Thruster2/Cmd - Eca/Thruster3/Cmd - Eca/Thruster4/Cmd - Eca/Thruster5/Cmd - Eca/Thruster6/Cmd - Eca/Thruster7/Cmd - Eca/Thruster0/count - Eca/Thruster1/count - Eca/Thruster2/count - Eca/Thruster3/count - Eca/Thruster4/count - Eca/Thruster5/count - Eca/Thruster6/count - Eca/Thruster7/count - Eca/Thruster0/rpm - Eca/Thruster1/rpm - Eca/Thruster2/rpm - Eca/Thruster3/rpm - Eca/Thruster4/rpm - Eca/Thruster5/rpm - Eca/Thruster6/rpm - Eca/Thruster7/rpm - Eca/Thruster0/temperature - Eca/Thruster1/temperature - Eca/Thruster2/temperature - Eca/Thruster3/temperature - Eca/Thruster4/temperature - Eca/Thruster5/temperature - Eca/Thruster6/temperature - Eca/Thruster7/temperature - Eca/Thruster0/errors - Eca/Thruster1/errors - Eca/Thruster2/errors - Eca/Thruster3/errors - Eca/Thruster4/errors - Eca/Thruster5/errors - Eca/Thruster6/errors - Eca/Thruster7/errors - Eca/Navigation_Mode - Eca/Sys_Status/drop_rate_comm - Eca/Sys_Status/errors_comm - Eca/Sys_Status/errors_count1 - Eca/Sys_Status/errors_count2 - Eca/Sys_Status/errors_count3 - Eca/Sys_Status/errors_count4

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

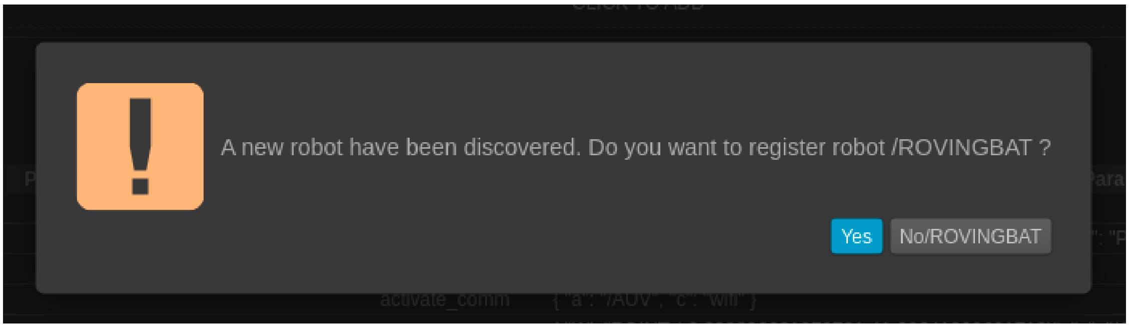
- Eca/Battery/id - Eca/Battery/charge_state - Eca/Battery/energy_consumed - Eca/Battery/charge_state		
SW_DC_030	Boolean Telemetry	
- Navigation_mode - Exploration_mode - Water_intake - Overcurrent - Overvoltage - Overheating - ThrusterFault		
SW_DC_031	GPS Telemetry	
SW_DC_032	Pose Telemetry	
robot_pose_crawler		
SW_DC_033	VIdeo Telemetry	
- Nav_cam : rtsp://192.168.1.134:5554/nav #dummy rtsp - ITV_cam: rtsp://192.168.1.154:5554/back #dummy rtsp - Arm_Cam : rtsp://192.168.1.137:5554/back #dummy rtsp		

4.2.2. Integration Results

The tests were done by SPACEAPPS and ECA using the following:

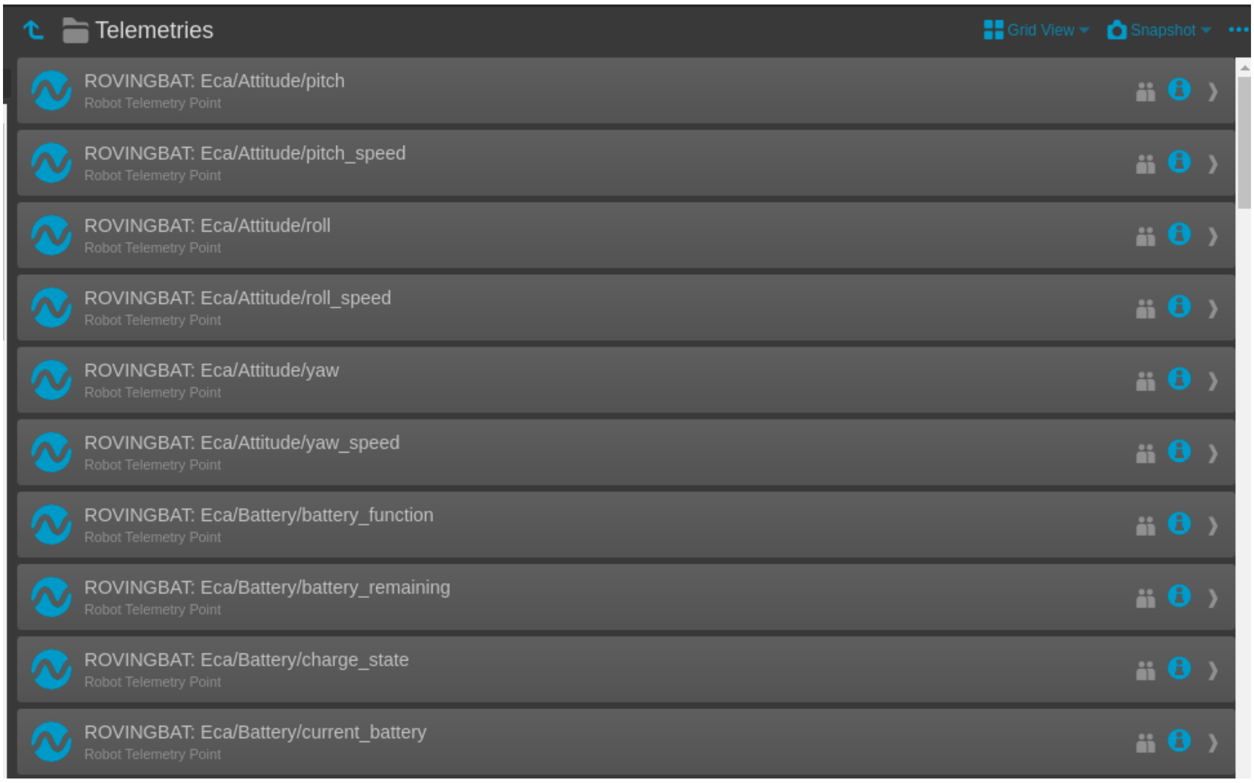
- A remote VPN connection: SPACEAPPS provided an openVPN server to which the ECA board was connected to. The robot configuration and the telemetry has been successfully observed using ros2 subscribers on the SPACEAPPS machine.
- ECA sent a rosbag archive which SPACEAPPS could test and validate with the SCC software. The SCC was able to
    - Discover the robot connecting in the interoperability network
    - See the telemetry topics provided by the robot

Once the interoperability network was connected to the robot and the SCC, the SCC UI was able to recognize a new robot (Figure 21) and to display the list of telemetries published by this robot (Figure 22).



A new robot have been discovered. Do you want to register robot /ROVINGBAT ?

Figure 23. Discovery alert for the ROVINGBAT



ROVINGBAT: Eca/Attitude/pitch
Robot Telemetry Point

ROVINGBAT: Eca/Attitude/pitch_speed
Robot Telemetry Point

ROVINGBAT: Eca/Attitude/roll
Robot Telemetry Point

ROVINGBAT: Eca/Attitude/roll_speed
Robot Telemetry Point

ROVINGBAT: Eca/Attitude/yaw
Robot Telemetry Point

ROVINGBAT: Eca/Attitude/yaw_speed
Robot Telemetry Point

ROVINGBAT: Eca/Battery/battery_function
Robot Telemetry Point

ROVINGBAT: Eca/Battery/battery_remaining
Robot Telemetry Point

ROVINGBAT: Eca/Battery/charge_state
Robot Telemetry Point

ROVINGBAT: Eca/Battery/current_battery
Robot Telemetry Point

Figure 24. ECA telemetry discovered in the SCC User Interface after passing the interoperability layer

4.3. SENSE ASV (INESC)

The ASV to be used during Scenario 1 in the ATLANTIS demonstration is SENSE (Figure 23), provided by INESC TEC. It will serve as a surface vehicle bringing the UAV close to the target of inspection, wait for the inspection to finish and

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

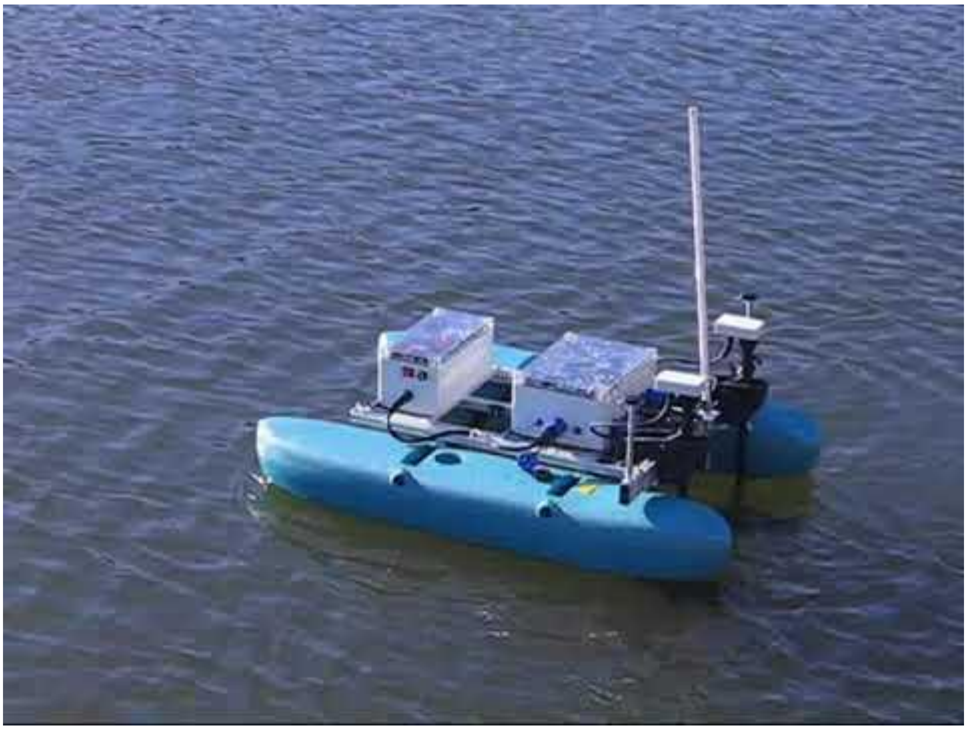


Figure 25. Old version of the Autonomous Surface Vehicle

4.3.1. Data Interfaces

The data interfaces for the ASV have been identified using the config file structure specific for interoperability. The possible commands that can be sent from the interop to tha ASV are listed in table 25, while the telemetry types supported are listed in table 26.

Table 25. Data Structures passed from Interoperability to ASV

From Interoperability Layer (Commands)			
Channel	**From**	**To**	**Partner**
WiFi	Interop	ASV	INESC
Ref	**Title**		
SW_CD_015	move_to		

Short description: step in an inspection process to move from point A to point B.
Parameters:
● Start_point : expected start point of the robot
● Target_point: destination of the robot
Example step structure:
{
"start_point": [-8.838826631079701, 41.68641999621718],
"target_point": [-8.840117492568483, 41.68692851740973]
}			

	release_uav		

<table>
<tr><td colspan="2"><i>Short description:</i>
<i>Parameters:</i>
● NA
<i>Example step structure</i>:</td></tr>
<tr><td></td><td>activate_sensor</td></tr>
<tr><td colspan="2"><i>Short description:</i> command to activate one of the onboard sensors
<i>Parameters:</i>
● Sensor: name of the sensor to be activated
<i>Example step structure</i>:
{
  "sensor": "camera"
}</td></tr>
<tr><td></td><td>activate_comm</td></tr>
<tr><td colspan="2"><i>Short description:</i> command to activate one of the communication endpoints onboard the robot
<i>Parameters:</i>
● Communication_device: name or identifier of the communication device
<i>Example step structure</i>:
{
  "communication_device": "wifi"
}</td></tr>
<tr><td></td><td>hover</td></tr>
<tr><td colspan="2"><i>Short description:</i> command for the robot to wait around the current point
<i>Parameters:</i>
● Duration_s: required duration of the hovering in seconds
<i>Example step structure</i>:
{
  "duration_s": "1800"
}</td></tr>
<tr><td></td><td>inspect</td></tr>
<tr><td colspan="2"><i>Short description:</i> inspect a target using an onboard sensor
<i>Parameters:</i>
● Point: the location of the target
● Target: the name of the target.
● Sensor: sensor name used for this inspection
<i>Example step structure</i>:
{
  "point": [-8.838826631079701, 41.68641999621718],
  "target": "splash_zone",
  "sensor": "camera"
 }</td></tr>
<tr><td></td><td>send_data</td></tr>
<tr><td colspan="2"><i>Short description:</i> upload survey data associated with a target to the SCC
<i>Parameters:</i>
● Target: the name of the inspected target.</td></tr>
</table>

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

- Sensor: the name of the target.
- Commnication_device: name or identifier of the communication device

Example step structure:

```
{
 "target":                                                     "splash_zone",
 "sensor":                                                         "camera",
 "communication_device":                                             "wifi"
}
```

	deactivate_comm

Short description: command to deactivate one of the communication endpoints onboard the robot

Parameters:

- Communication_device: name or identifier of the communication device

Example step structure:

```
{
 "communication_device": "wifi"
}
```

	deactivate_sensor

Short description: command to deactivate one of the onboard sensors

Parameters:

- Sensor: name of the sensor to be deactivated

Example step structure:

```
{
 "sensor": "camera"
}
```

Table 26. Data Structures passed from ASV to Interoperability

To Interoperability Layer (Data & TM)			
Channel	**From**	**To**	**Partner**
WiFi	ASV	Interop	INESC TEC
Ref	**Title**		
SW_DC_029	Numerical Telemetry		
/zarco/state/system_status /zarco/state/armed /zarco/mavros/global_position/rel_alt /zarco/altitude /zarco/angular_vel /zarco/state/mode /zarco/linear_vel /zarco/safety/stop_interop			
SW_DC_030	Boolean Telemetry		
/zarco/state/power_on /zarco/safety/stop			

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

SW_DC_031	GPS Telemetry
/zarco/gps/fix	
SW_DC_032	Pose Telemetry
/zaco/odometry/filtered /zarco/mavros/local_position/pose /zarco/odometry/pose	
SW_DC_033	Image Telemetry
mynteyeleft/image_color/compressed /zarco/usb_cam/image_raw/compressed	
SW_DC_033	Point Cloud
/zarco/velodyne_points	

4.3.2. Integration Results

The ASV was integrated with the SCC using the interop. The following has been tested:

- **VPN connection:** SCC and robot connected to the same network using OpenVPN connection.
- **ROS1 to ROS2 bridge:** ROS1 messages translated and republished in a ROS2 format.
- **ROSbag record and replay:** ROS2 record of the telemetries topic and the config topic on the robot side and replay of the data on the SCC side.
- **Discovery and identification of the Robot:** Robot automatically discovered by the SCC by receiving the robot config.
- **Boolean and Numerical Telemetry:** Reception and visualisation of the robot telemetries.

The Zarco config was published and successfully added to the SCC list of robots (Figure 26)

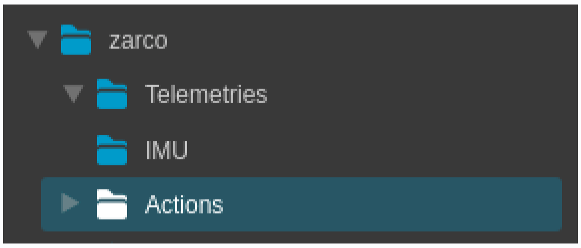


Figure 26. Zarco robot showing in the SCC UI after discovery

4.4. CROW/RAVEN UAV (INESC)

In the scope of Scenario 1, a collaboration between an ASV and a UAV is foreseen. Both vehicles are provided by INESC TEC and will be connected to the rest of the system. The name of the UAV is CROW. The data interfaces of the UAV are listed in the

4.4.1. Data Interfaces

Table 27. Data Structures passed from UAV to Interoperability

<table>
<tr><th colspan="4">From Interoperability Layer (Commands)</th></tr>
<tr><td><b>Channel</b></td><td><b>From</b></td><td><b>To</b></td><td><b>Partner</b></td></tr>
<tr><td>WiFi</td><td>Interop</td><td>UAV</td><td>INESC</td></tr>
<tr><td><b>Ref</b></td><td colspan="3"><b>Title</b></td></tr>
<tr><td>SW_CD_015</td><td colspan="3">fly_to</td></tr>
<tr><td colspan="4"><i>Short description:</i> step in an inspection process to move from point A to point B.
<i>Parameters:</i>
● Start_point : expected start point of the robot
● Target_point: destination of the robot
● Altitude: destination altitude above sea level
<i>Example step structure</i>:
{
"start_point": [-8.838826631079701, 41.68641999621718],
"target_point": [-8.840117492568483, 41.68692851740973],
"altitude": 83
}</td></tr>
<tr><td></td><td colspan="3">take_off</td></tr>
<tr><td colspan="4"><i>Short description:</i>
<i>Parameters:</i>
● NA
<i>Example step structure</i>:</td></tr>
<tr><td></td><td colspan="3">activate_sensor</td></tr>
<tr><td colspan="4"><i>Short description:</i> command to activate one of the onboard sensors
<i>Parameters:</i>
● Sensor: name of the sensor to be activated
<i>Example step structure</i>:
{
"sensor": "camera"
}</td></tr>
<tr><td></td><td colspan="3">activate_comm</td></tr>
<tr><td colspan="4"><i>Short description:</i> command to activate one of the communication endpoints onboard the robot
<i>Parameters:</i>
● Communication_device: name or identifier of the communication device
<i>Example step structure</i>:
{
"communication_device": "wifi"
}</td></tr>
</table>

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

	hover

Short description: command for the robot to wait around the current point

Parameters:

- Duration_s: required duration of the hovering in seconds

Example step structure:

```
{
 "duration_s": "1800"
}
```

	inspect

Short description: inspect a target using an onboard sensor

Parameters:

- Point: the location of the target
- Target: the name of the target.
- Sensor: sensor name used for this inspection

Example step structure:

```
{
 "point":                [-8.838826631079701,                41.68641999621718],
 "target":                                                             "blade_1",
 "sensor":                                                              "camera"
 }
```

	send_data

Short description: upload survey data associated with a target to the SCC

Parameters:

- Target: the name of the inspected target.
- Sensor: the name of the target.
- Commnication_device: name or identifier of the communication device

Example step structure:

```
{
 "target":                                                             "blade_1",
 "sensor":                                                              "camera",
 "communication_device":                                                  "wifi"
}
```

	deactivate_comm

Short description: command to deactivate one of the communication endpoints onboard the robot

Parameters:

- Communication_device: name or identifier of the communication device

Example step structure:

```
{
 "communication_device": "wifi"
}
```

	deactivate_sensor

Short description: command to deactivate one of the onboard sensors

Parameters:

- Sensor: name of the sensor to be deactivated

Example step structure:

{
  "sensor": "camera"
}	

	land

Short description: land on target
Parameters:
● Point: destination point (lat lon)
● Target: destination target.
Example step structure:
{
  "point": [-8.838826631079701, 41.68641999621718],
  "target": "ASV",
}	

Table 28. Data Structures passed from UAV to Interoperability

To Interoperability Layer (Data & TM)			
Channel	**From**	**To**	**Partner**
WiFi	UAV	Interop	INESC TEC
Ref	**Title**		
SW_DC_029	Numerical Telemetry		
/crow/state/system_status /crow/state/armed /crow/mavros/global_position/rel_alt /crow/altitude /crow/angular_vel /crow/state/mode /crow/linear_vel /crow/safety/stop_interop			
SW_DC_030	Boolean Telemetry		
/crow/state/power_on /crow/safety_stop			
SW_DC_031	GPS Telemetry		
/crow/gps/fix			
SW_DC_032	Pose Telemetry		
/zaco/odometry/filtered /crow/mavros/local_position/pose /crow/odometry/pose			
SW_DC_033	Image Telemetry		

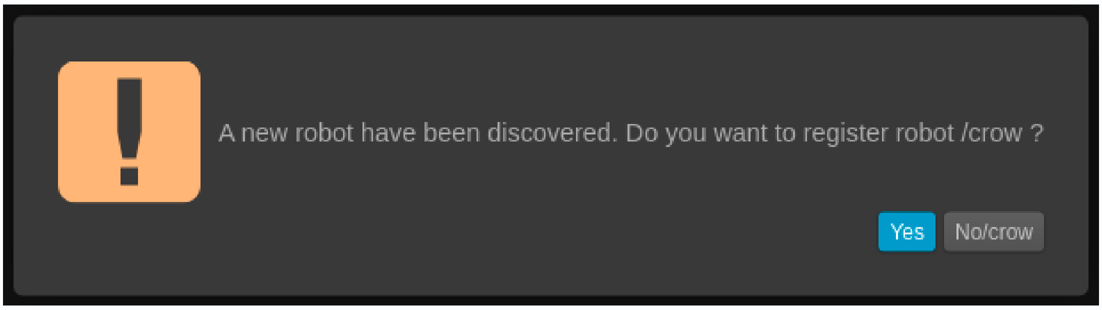
mynteyeleft/image_color/compressed /crow/usb_cam/image_raw/compressed	
SW_DC_033	Point Cloud
/crow/velodyne_points	

4.4.2. Integration Results

The UAV was integrated with the SCC using the interop. The following has been tested:

- **VPN connection:** SCC and robot connected to the same network using OpenVPN connection.
- **ROS1 to ROS2 bridge:** ROS1 messages translated and republished in a ROS2 format.
- **ROSbag record and replay:** ROS2 record of the telemetries topic and the config topic on the robot side and replay of the data on the SCC side.
- **Discovery and identification of the Robot:** Robot automatically discovered by the SCC by receiving the robot config.
- **Boolean and Numerical Telemetry:** Reception and visualisation of the robot telemetries.

A rosbag prepared by INESC was run on the local SCC network. The rosbag play simulated the publishing of the config and data. The SCC recognized the new robot connected and showed a pop up for the user to add it to the list of robots (Figure 27).



A new robot have been discovered. Do you want to register robot /crow ?

Figure 27. Robot discovery SCC pop-up for crow robot

4.5. Supervisory Control Centre

The communication between different components deployed onshore will be based on the client-server architecture. At the centre of the deployment will be a database containing the information gathered from the operations. All the other components will act like clients to the database and will read or write information using decoupled endpoints that offer a RESTFul API and an OGC Standardised interface.

4.5.1. Data Interfaces

As all the other processing nodes of the Atlantis architecture, the SCC exchanges data using the data gateway software. The definition of data interfaces is shown in Table 29 (from SCC to interoperability) and Table 30 (from Interoperability to SCC)

Table 29. Data Structures passed from SCC to Interoperability

SCC To Interoperability			
Channel	**From**	**To**	**Partner**
RF	SCC	Interop Layer	SPACEAPPS
Ref	**Title.**		
SW_AC_001	Get Robot Configuration		
Send request to retrieve robot configuration.			
SW_AC_002	Get Robot State		
Send request to retrieve robot state.			
SW_AC_003	Execute Plan		
Deploy execution of a plan. The message will contain: ● mission ● robots ● plan - list of actions ● parameters ● timeout - the time for the robot to react to the request			
SW_AC_004	Execute Command		
Deploy the execution of a single command. The message will contain: ● robot ● command ● parameters ● timeout - the time for the robot to react to the request			
SW_AC_005	Get Data Payload		
Send request to retrieve data payload. The message will contain: ● robot ● data type ● time interval ● mission - optional			
SW_AC_006	Get System Overview		
Query to the interoperability layer to retrieve the full map of the system. This will include the registered robots, their types and their capabilities.			

Table 30. Data Structures passed from Interoperability to SCC

Interoperability To SCC			
Channel	**From**	**To**	**Partner**
RF	Interoperability	SCC	SPACEAPPS

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Ref	**Title.**		
SW_DA_002	Robot Alive		
Periodic signal sent to the SCC confirming that the robot is available. The packet has two functions: ● inform the operator about the robot being active ● check the communication link The “robot alive” data packet will contain the namespace of the robot, and the time stamp from the emission time.			
SW_DA_004	Mission Plan Execution Feedback		
For each of the missions deployed on the robot, the operator will view the state of the execution for each of the tasks, including any replanning activities. The feedback will contain the current action that is performed, together with the time stamp. Since we are working in a multi-agent system, the feedback can contain information from multiple robots.			
SW_DA_005	Mission Plan Execution Result		
At the end of an automatic plan, the SCC will receive the outcome of the mission plan execution. This can be: ● SUCCESS ● ABORTED ● CANCELED_BY_OPS			
SW_DA_006	Command Execution Feedback		
Execution for an ongoing requested command. The ongoing command feedback is not mandatory for all commands. The feedback can be: ● Boolean ● Numerical ● Positional ● Binary Payload It can contain one or multiple data formats listed above. Each data payload will be named.			
SW_DA_007	Command Execution Result		
● SUCCESS ● ABORTED ● CANCELED_BY_OPS			
SW_DA_008	Numeric Telemetry		
Numeric data associated with the state of the robotic subsystem. For example: battery level, motor current, speed etc.			
SW_DA_009	Position		
Robot Position and Orientation			
SW_DA_010	Image		
Visual data acquired by the on board perception sensors. It can be the output of a camera or the output of a processed sensor (e.g. bathymetry map).			
SW_DA_011	Point Cloud		
A list of positioned points containing various metadata.			

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

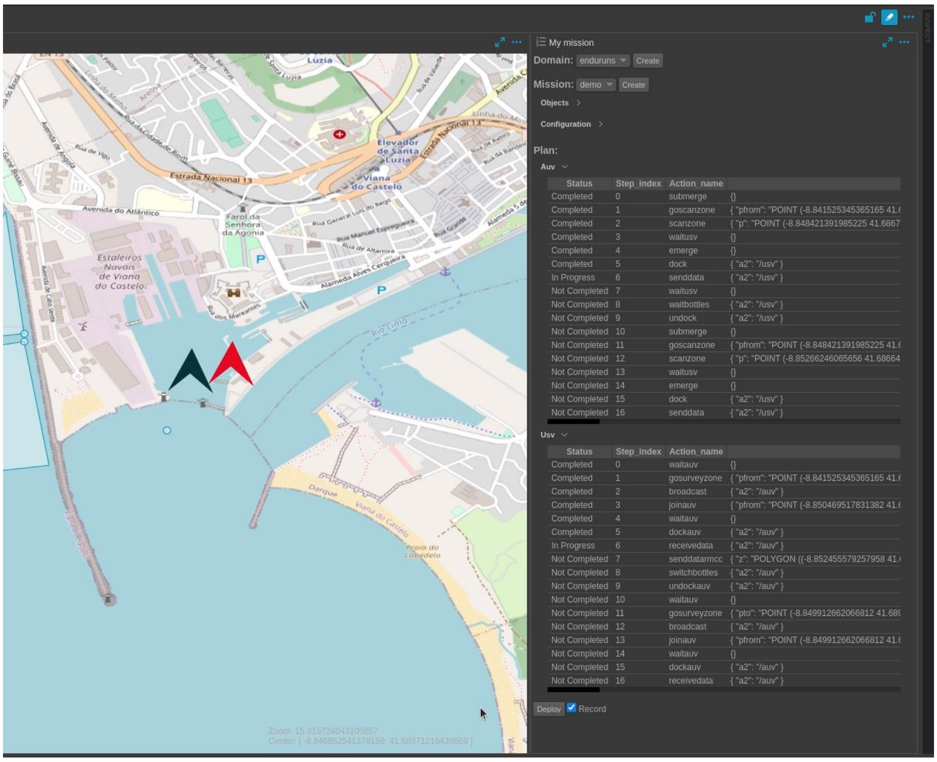
SW_DA_012	3D Mesh
A list of 3D vertices and a list of edges that link the vertices.	
SW_DA_013	Device Wakeup
Data packet sent at the startup of the robot. It contains the robot namespace and the time stamp. This message will be followed up by the alive periodic message.	
SW_DA_014	Error
Error code associated with a robot or the communication units.	

4.5.2. Integration Results

Mock robots have been created using the interop to connect to the SCC. By using mock robots we were able to test most of the features of the interop and the SCC, from the basics of monitoring by receiving telemetries to more complex features such as deploying a mission plan on the robot and monitoring its execution directly on the SCC.

- **VPN connection:** SCC and robot connected to the same network using OpenVPN connection.
- **Robot discovery:** Automatic discovery of the robots on the network. Also tested with more simple assets publishing their config.
- **Robot position and orientation:** Reception of the robot pose, position and orientation of the robot, and its GPS coordinates. Successfully received and displayed on a 2D and 3D word map. See figure 28.
- **Telemetries reception**: Reception and display of numerical and boolean telemetries.
- **Send mission plan:** Deploy mission plan on a robot.
- **Monitor plan execution:** Monitor plan status during plan execution. See Figure 29.
- **Images and video reception:** Reception and display of images and videos coming from the robot. See figure 30.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.



My mission

Domain: enduruns Create

Mission: demo Create

Objects >

Configuration >

Plan:

Auv

Status	Step_index	Action_name	
Completed	0	submerge	{}
Completed	1	goscanzone	{ "pfrom": "POINT (-8.841525345365165 41.6
Completed	2	scanzone	{ "p": "POINT (-8.848421391985225 41.6867
Completed	3	waitusv	{}
Completed	4	emerge	{}
Completed	5	dock	{ "a2": "/usv" }
In Progress	6	senddata	{ "a2": "/usv" }
Not Completed	7	waitusv	{}
Not Completed	8	waitbottles	{ "a2": "/usv" }
Not Completed	9	undock	{ "a2": "/usv" }
Not Completed	10	submerge	{}
Not Completed	11	goscanzone	{ "pfrom": "POINT (-8.848421391985225 41.6
Not Completed	12	scanzone	{ "p": "POINT (-8.85266246065656 41.68664
Not Completed	13	waitusv	{}
Not Completed	14	emerge	{}
Not Completed	15	dock	{ "a2": "/usv" }
Not Completed	16	senddata	{ "a2": "/usv" }

Usv

Status	Step_index	Action_name	
Completed	0	waitauv	{}
Completed	1	gosurveyzone	{ "pfrom": "POINT (-8.841525345365165 41.6
Completed	2	broadcast	{ "a2": "/auv" }
Completed	3	joinauv	{ "pfrom": "POINT (-8.850469517831382 41.6
Completed	4	waitauv	{}
Completed	5	dockauv	{ "a2": "/auv" }
In Progress	6	receivedata	{ "a2": "/auv" }
Not Completed	7	senddatarmcc	{ "z": "POLYGON ((-8.852455579257958 41.(
Not Completed	8	switchbottles	{ "a2": "/auv" }
Not Completed	9	undockauv	{ "a2": "/auv" }
Not Completed	10	waitauv	{}
Not Completed	11	gosurveyzone	{ "pto": "POINT (-8.849912662066812 41.68
Not Completed	12	broadcast	{ "a2": "/auv" }
Not Completed	13	joinauv	{ "pfrom": "POINT (-8.849912662066812 41.6
Not Completed	14	waitauv	{}
Not Completed	15	dockauv	{ "a2": "/auv" }
Not Completed	16	receivedata	{ "a2": "/auv" }

Deploy ☑ Record

Figure 28. Mission execution monitoring on UI

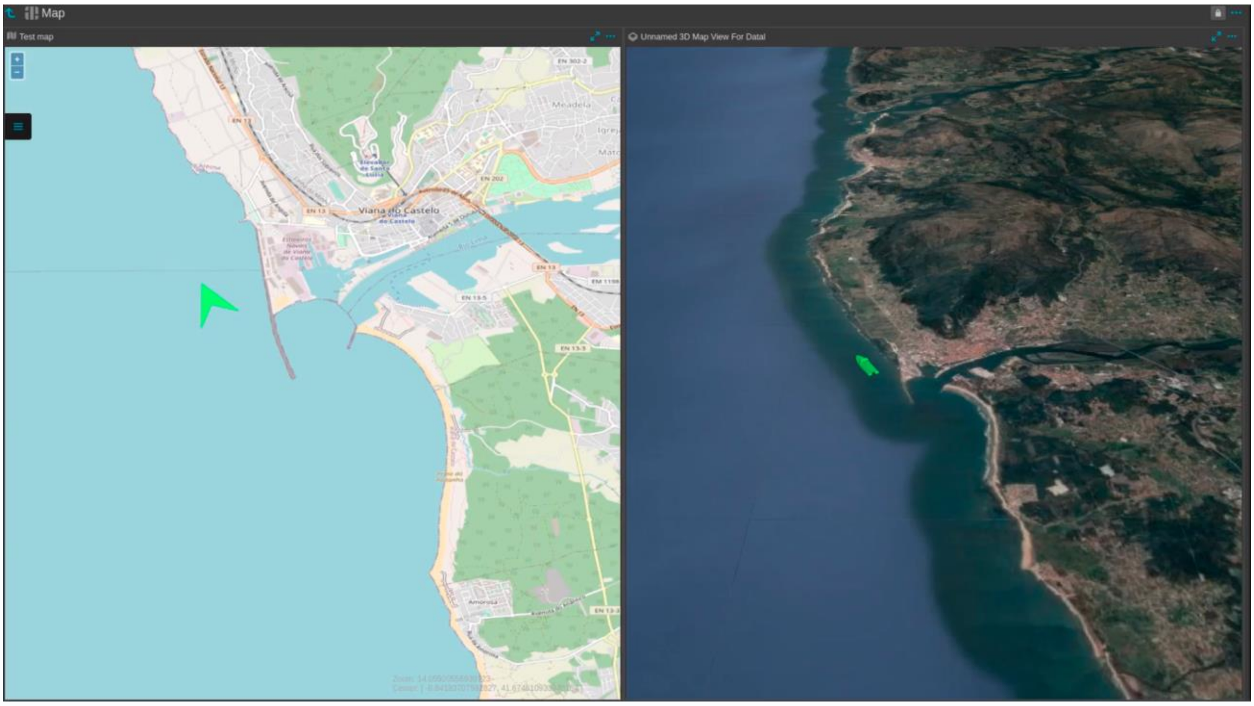


Figure 29. Robot position and orientation monitoring on 2D and 3D maps.

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

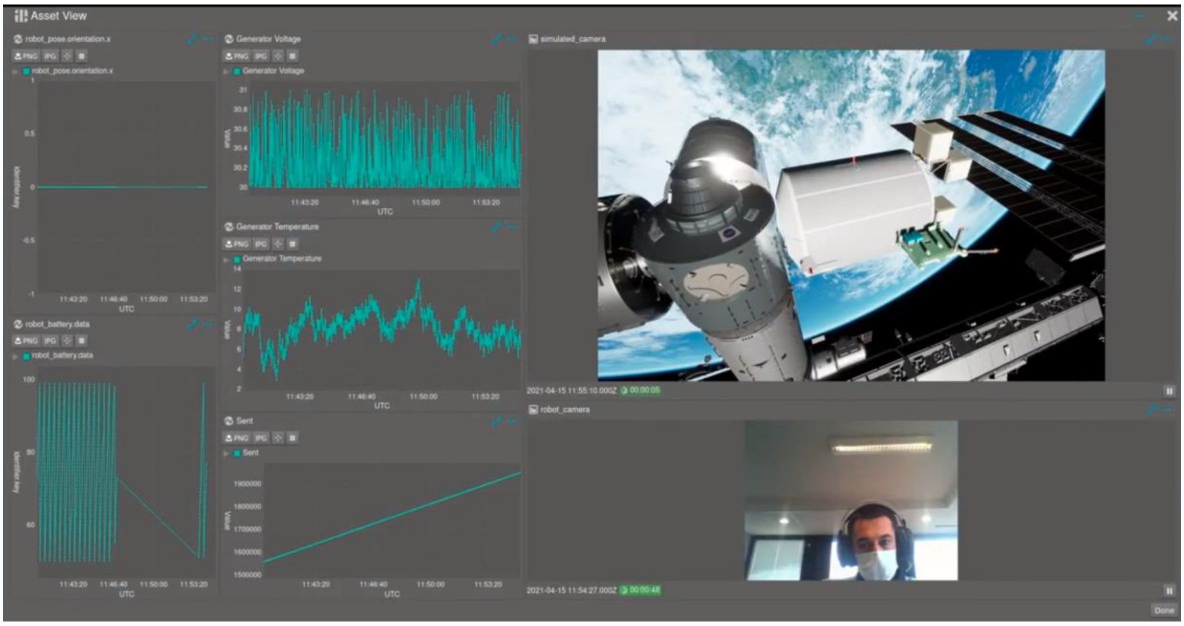


Figure 30. Telemetries from the mock robots, example images and local webcam displayed on the SCC UI

The remote control centre will be set up on the local premises in Viana do Castello, but also in Bruxelles at spaceapps premises. The setup from DexROV (Figure 31) will be reused for this purpose. The setup includes 3 operator spots each connected to three screens where the SCC UI will run. The backend will run in the cloud and will be connected with the interoperability layer via the data gateway component.

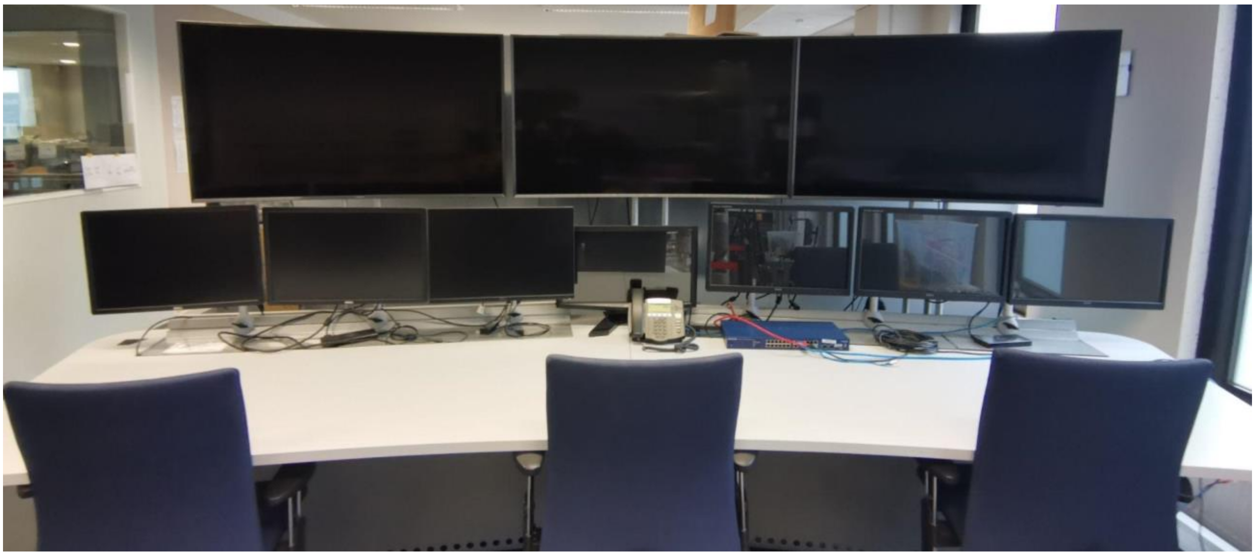


Figure 31. Remote control centre setup prepared for running the SCC

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

4.6. Communication Systems

The communication systems were procured by INESC and SPACEAPPS and commissioning was performed on spaceapps premises. The commissioning included initial configuration and testing of each individual equipment in a relevant network setup.

4.6.1. Local Network Hardware

Testbed local network is composed of routers, switches and embedded computers. The layout of the local network is shown in Figure 32. All devices except the outdoor antennas will be contained inside an IP67 case.

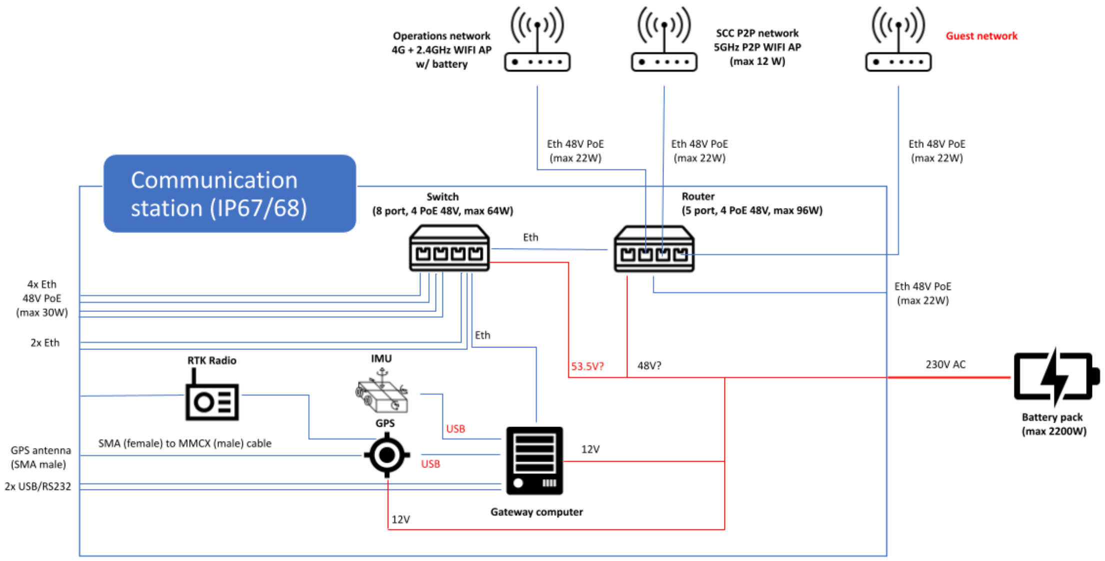


Figure 32. Onshore testbed connectictions layout

All communication hardware foreseen have been procured by INESC and SPACEAPPS (at least one device of a selected type). SPACEAPPS have performed the unpacking and commissioning of all hardware. The list of tested devices can be seen in table 31.

Table 31. List of network nodes tested hardware

Model	Quantity	Content
MicroTik Routerboard Hex-PoE	2	1x router board 1x Power Adapter 1x IEC cable Quick Guide
MicroTik Routerboard SXTG-5HPacD	1	1x 24V 0.8 A power adapter 1x Metal ring 1x Pole mounting bracket

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

		1x Gigabit oE injector 1x Quick Guide
Tp-Link 8-port PoE+Switch TL-SG108PE	1	1x 8-port switch 1x installation guide 1x Power adapter

For most of the communication nodes, powering will be provided through PoE. The power for PoE is delivered by the same router that handles the package exchanges from the network. The CPE access point sending data from the floating structure to the SCC is one of the devices receiving power from the router. The unboxing of both can be seen in Figure 33.

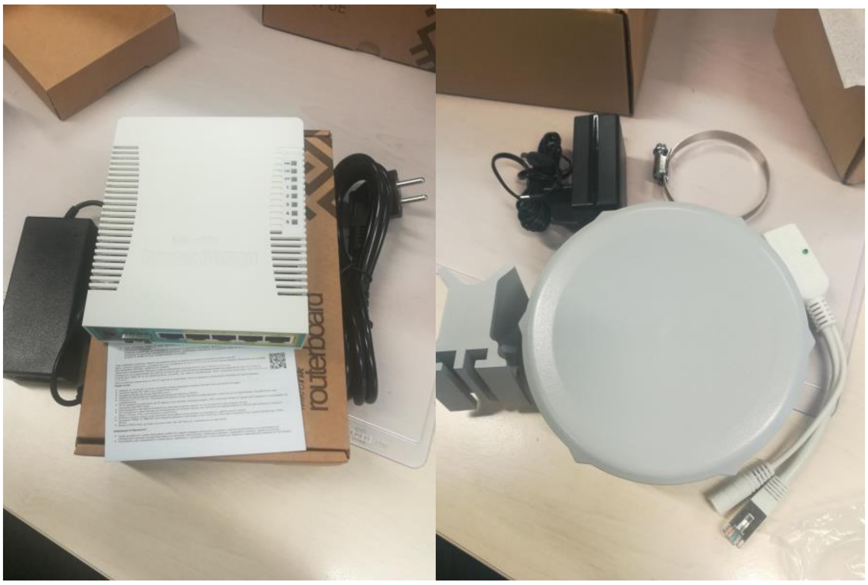


Figure 33. Mikrotik PoE and CPE with directional antenna Routers

The embedded computer running the data gateway is an ODROID N2 4G running Ubuntu 20.04. The content of the package is listed in Table 32 and the image of the hardware is shown in Figure 34.

Table 32. Gateway Computer Package Inventory

Model	Quantity	Content
ODROID-N2 4G	2	1x ODROID-N2 board
ODROID Component mix	1	2x Cooling fan 1x EU socket adapter 1x 12V2A power adapter

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

		1x USB3.0 to eMMC reader 2x Micron 1TB micro SD card

Figure 34. Odroid N2 and Accessories Package during package inspection

4.6.2. Software Installation Setup for Gateway Computers

An ODROID embedded computer was set up and prepared to be used in the pilot. The setup steps for installing the dependencies, testing them and installing the interoperability library are the following:

1.Operating System Setup

- Download an image of Ubuntu Minimal (see here).
- Download Etcher here to flash an image to your memory card.
- Flash the image:

1. Insert the eMMC card as shown below.
2. Open Etcher
3. Select the downloaded OS image
4. Select the inserted memory card. (Normally, the memory card is detected automatically.)
5. Click Flash!

- Connect the eMMC card to the ODROID board.

2. ROS2 Setup and test steps:

- Setup Locale

```
sudo apt update && sudo apt install locales
sudo locale-gen en_US en_US.UTF-8
sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
export LANG=en_US.UTF-8
```

- Setup Sources

```
sudo apt update && sudo apt install curl gnupg2 lsb-release
sudo curl -sSL <https://raw.githubusercontent.com/ros/rosdistro/master/ros.key>  -o
/usr/share/keyrings/ros-archive-keyring.gpg
echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/ros-
archive-keyring.gpg] <http://packages.ros.org/ros2/ubuntu> $(source /etc/os-release
&& echo $UBUNTU_CODENAME) main" | sudo tee /etc/apt/sources.list.d/ros2.list >
/dev/null
```

- Install ROS2 core packages

```
sudo apt update
sudo apt install ros-foxy-ros-base
```

- Install ROS2 core packages

```
sudo apt install python3-colcon-common-extensions
```

- Setup the environment

```
source /opt/ros/foxy/setup.bash
echo "source /opt/ros/foxy/setup.bash" >> ~/.bashrc
```

- Test middleware installation using the demo nodes

```
sudo apt install ros-foxy-demo-nodes-cpp ros-foxy-demo-nodes-py
ros2 run demo_nodes_cpp talker
ros2 run demo_nodes_py listener
```

3. UV-Interoperability Software build and install

```
sudo apt-get install libyaml-cpp-dev build-essential
git clone <https://gitlab.spaceapplications.com/c4i/uv-interoperability.git>
cd uv-interoperability
git submodule init && git submodule update --recursive
. /opt/ros/foxy/setup.sh
colcon build
. install/setup.sh
```

4. UV-Interoperability-Examples build:

```
git clone <https://gitlab.spaceapplications.com/c4i/uv-interoperability-example.git>
cd uv-interoperability-example
mkdir build
cd build
cmake ..
make
```

5. Connection Test

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Data Gateway computer installation was validated by the following steps:

- Run the C4i server and the C4i to Ros interface
- Build and source the UV Interoperability Library
- Run ./example-robot ../config/robot-config.yml from the UV Interoperability Example directory.
- TEST_ROBOT is connected to the SCC.

4.6.3. 3G / 4G Outdoor Router

The outdoor router for 4G and 3G communication was installed and tested (Figure 35). The performance observed in accessing the internet using the Belgium network was 32 Mbits/s for both download and upload. The test was done while passing the connection through the local WiFi network.

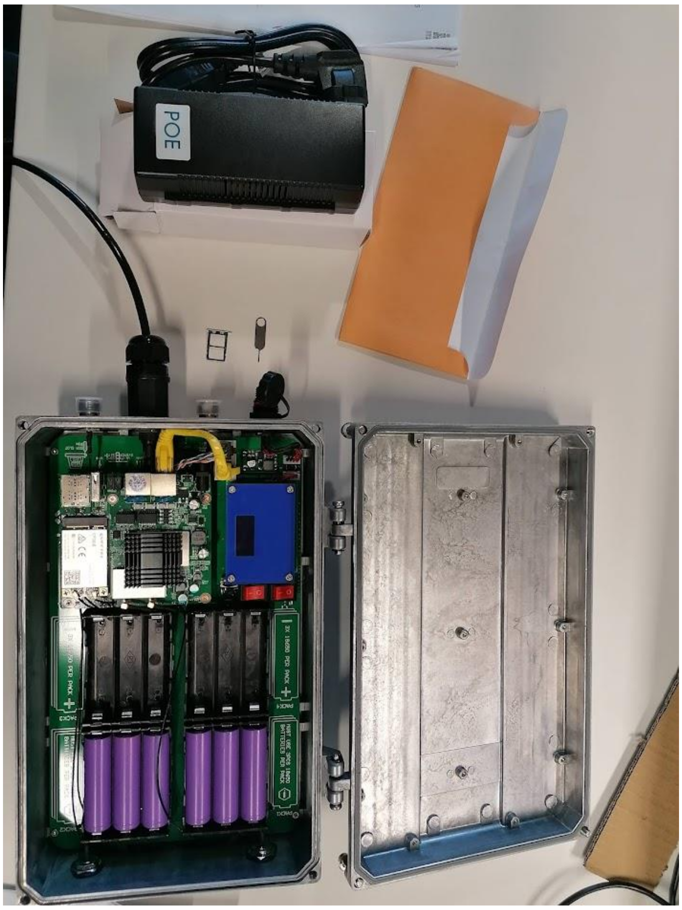


Figure 35. Outdoor 4G modem during unpacking

4.6.4. Satellite Modem and Antenna - Thales Vesselink

The Thales Vesselink satellite antenna has been tested to send ROS messages through its internet connection. To proceed we used two computers, one simulating the SCC, and the other a robot. The objective of this test was to send data from the computer simulating the robot to the other computer simulating the SCC through the satellite. To have the robot on the same network a VPN was used, having

This project has received funding from the European Union's Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

both computers connected to it. To perform the tests we mounted the satellite antenna on top of the Space Applications Services building rooftop (Figure 36).

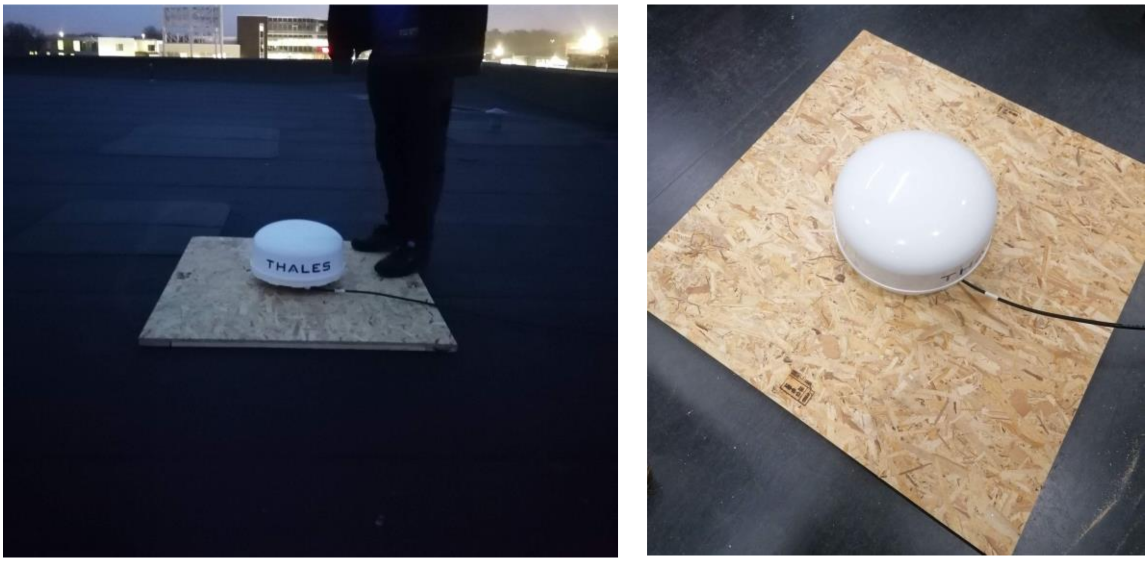


Figure 36. Thales VesseLINK test setup

The tests were performed with different data types and different sizes of data. The hardware is connecting well to the network and the tests were done without problems. The modem was set up to limit the traffic to the IPs used in the ATLANTIS network. Below the performance tests results.

Results using strings and numerical data using different rates are shown in table 33.

Table 33. Satcom test result with small telemetry

Data type	**Rate (Hz)**	**Received**
String	1	100 %
String	10	100%
String	100	100%
String	1000	95%
String	5000	50%

Results using images with different parameters are shown in Table 34.

Table 34. Satcom test result with images

Data Type	**Resolution**	**Rate**	**Received**
Image	160 x 120	2	100%
Image	160 x 120	4	100%
Image	160 x 120	8	100%

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, under the Grant Agreement no. 871571.

Image	160 x 120	16	100%
Image	160 x 120	16	100%
Image	160 x 120	30	100%
Image	160 x 120	30	100%

5. Conclusion

During task *T2.5 Communication Links and Interfaces for Interoperability* SPACEAPPS have designed a specification for interoperability to be used in robotics IMR operations based on the requirements identified in T2.2. The partners providing robot systems for demonstration have provided support to refine the way these robots are interconnected. The base communication protocol used for the data exchange is DDS and the middleware to integrate is ROS2. For the hardware interoperability, Atlantis supports acoustic, WiFi, Satcom, Ethernet and GSM communication.

For the software integration of the robots SPACEAPPS have developed a library and sample code that can be used to connect an existing system. All the partners providing robots (UdG, IQUA, ECA, INESC) have evaluated and tested the library with their robots. The test results are presented in this document. Due to travel limitation all the integration testing was done using remote calls and remote connection. The need of setting up remote networks for the interoperability layer has been a good opportunity to test the limitation of the designed connection. Further refinement of the integration will be done on site in the scope of WP5 during the final integration.

The work done in this task will be used as input for T3.2 IT Systems and SCC and WP5. In the scope of these WPs the hardware will be installed in the testbed and on the robots and will be tested in the real environment.