

Faculty of Engineering of the University of Porto



Acoustic Networking for Controlling Underwater Data Mules

Mariam Ahmed Osman Ahmed Mohamed

Master in Electrical and Computers Engineering

Supervisor: Rui Lopes Campos
Co-Supervisor: Filipe Borges Teixeira

31 March, 2022

Abstract

Oceans cover more than 70% of the earth. Innovative technologies such as underwater wireless communications (UWC) strongly impact the observation of marine life and tracking of water pollution. Moreover, they have a significant role in oil and gas industries, military, and environmental operations. Consequently, there is a noticeable interest in investigating UWC, focusing on supporting various applications and services. The increasing use of Autonomous Underwater Vehicles (AUV) and Remotely Operated Vehicles (ROVs) in underwater missions demand a broadband, cost-effective communications solution suitable for these environments. Different techniques can be used to provide underwater wireless communications. These include acoustic waves, radio frequency and optical wireless communications. Among the three technologies, acoustic communications enable long ranges up to 20 km but provide low bitrates and have high communication delays. RF waves are not affected by turbidity and acoustic noise, but they suffer from having a limited range through the water due to attenuation. While optical systems may provide high data rates, they are dependent on proper light beam alignment and clear line-of-sight. Hence, there is a need to create a solution that combines each technology's advantages and overcomes their limitations. The GROW solution developed by INESC TEC is able to provide a long-range, high bitrate underwater wireless communications. The GROW solution considers the use of AUVs as data mules, short-range high bitrate wireless RF or optical communications, and long-range, low bitrate acoustic communications for network control.

This dissertation aims to develop an improved version of the GROW Underwater Data Muling Protocol (UDMP), an out-of-band acoustic protocol for the operation and scheduling of the data mules considered in the GROW solution. The multi-node protocol takes advantage of JANUS underwater communications to avoid network collisions and ensure reliable acoustic communications. The solution was implemented and validated using the UnetStack simulator. The obtained results show that the successful exchange of the UDMP messages over JANUS standard for 1 and 2 data mules, in a maximum operation range of 5 km. We could also study the RSSI, SNR, delay, and frame loss ratio for different distances. These results show that the proposed solution is able to provide a proper out-of-band channel for the control of the data muling process.

Acknowledgment

First and foremost, thanks and praise be to God for all the countless grace and blessings. I am eternally grateful for all the circumstances that God arranged for me at the right time to achieve what I was dreaming.

I would like to thank my supervisors, Dr. Rui Lopes Campos and Eng. Filipe Borges Teixeira, who saved no effort and time to help and support me with their knowledge and expertise along the dissertation journey, and to INESC TEC, for offering all the help and guidance and giving me the opportunity to join the team and to know my supportive colleagues.

To my dad and mom, without them I wouldn't be who I am today. Thank you for everything they provided to me and for the sacrifices during all my life, and to my sisters for supporting, loving, and believing in me.

To my lovely husband, Abdelrahman, for all the help that I cannot describe in words showing how much he always helps, supports and cares about me. To my beloved son, who I missed playing with on many occasions to do my work and study, he was patient with me.

To my Portuguese friends, namely Catarina, Sergio, and Antonio for always being there to help me during our study in FEUP and being beside me through the difficult times.

To my professors I met throughout the journey who helped and supported me in difficult moments, for translating the Portuguese classes, for offering help even without waiting to be asked, also I would like to thank the professors, who didn't help, they taught me how to count on myself, trust myself, accept the challenges and never give up.

Today I am not like the first day I came to Portugal, today is a remarkable day in my life.

Mariam

“It does not matter how slowly you go, as long as you do not stop.”

Confucius

Contents

Abstract	iii
Acknowledgment	v
Chapter 1	1
Introduction.....	1
1.1 Context.....	1
1.2 Motivation	2
1.3 Objectives.....	3
1.4 Main Contribution	3
1.5 Document Structure	3
Chapter 2	5
State of the Art	5
2.1 Underwater Communications Technologies	5
2.1.1 Acoustic Wireless Communications	5
2.1.2 Optical Wireless Communications.....	7
2.1.3 Radio Frequency Wireless Communications	9
2.2 Underwater Network Simulation	9
2.2.1 Acoustic Channel Models.....	9
2.2.2 Underwater Acoustic Network Simulation Platforms	10
2.2.3 UnetStack Simulator	12
2.3 JANUS Underwater Communications Standard	16
2.3.1 Protocol Description.....	16
2.3.2 Key Elements and Features	17
2.3.3 JANUS-based Capabilities	22
2.3.4 JANUS Experimental and Results	23
Chapter 3	27
Proposed Solution	27
3.1 Overview.....	27
3.2 Reliable Underwater Data Muling Protocol.....	28
3.3 Protocol messages	28
3.4 Protocol message sequence.....	30
3.4.1 Simulation with a Single Data Mule Unit.....	30
3.4.2 Simulation with Multiple Data Mule Units	33
Chapter 4	36
Implementation and Experimentation Results	36

4.1 Test Scenario and Network Simulation 36

 4.1.1 One Data Mule Unit 36

 4.1.2 One Data Mule Unit with failure message 38

 4.1.3 Two Data Mule Units 38

4.2 Simulation Results 39

 4.2.1 Received Signal Strength Indicator (RSSI)..... 39

 4.2.2 Signal to Noise Ratio (SNR)..... 40

 4.2.3 Delay 41

 4.2.4 Frame Loss 42

Chapter 5..... 44

Conclusion and Future Work 44

References..... 46

List of Figures

Figure 1.1: Underwater wireless communications technologies [8].....	1
Figure 1.2: FCT GROW Solution [11].....	2
Figure 2.1: Absorption coefficient of electromagnetic radiation at various wavelength [14].....	8
Figure 2.2: Underwater absorption coefficients visible light wavelengths [13].....	8
Figure 2.3: Electromagnetic Spectrum [14].....	9
Figure 2.4: The architecture of UnetStack [17].....	13
Figure 2.5: The architecture of UnetStack-based UnetSim [16].....	13
Figure 2.6: Five network nodes deployed in Singapore waters. Locations P21, P22, P27, P28 and P29 correspond to the drop locations of the 5 equivalently numbered nodes [24]	14
Figure 2.7: Typical configuration of a Unet-PANDA network node [24].....	15
Figure 2.8: The STARFISH AUV being deployed as a mobile network node during the MISSION 2013 experiment [6].....	15
Figure 2.9: Seven static network nodes deployed in Singapore waters during the MISSION 2013 experiment [6]	16
Figure 2.10: Block diagram for the JANUS baseline packet encoding process [6]	17
Figure 2.11: The structure of a JANUS packet [23]	17
Figure 2.12: JANUS signal in a time-frequency plot [23].....	19
Figure 2.13: First contact and language switching [19]	23
Figure 2.14: JANUS Node Discovery	24
Figure 2.15: JANUS communication capability discovery	24
Figure 2.16: JANUS language switching	25
Figure 3.1: The GROW proposed Solution [11].....	28
Figure 3.2: Janus signal in a time-frequency plot [23].....	31

Figure 3.3: Message sequence diagram for 1 data Mule	32
Figure 3.4: Message sequence diagram for 1 Data Mule Unit Failure Example.....	33
Figure 3.5: Message sequence diagram for 2 data Mules	34
Figure 3.6: Message sequence diagram for 2 Data Mule Units failure	35
Figure 4.1: Simulation script on Central Station	37
Figure 4.2: Simulation script on Data Mule Unit 1.....	37
Figure 4.3: Simulation script on Survey Unit	38
Figure 4.4: Central Station Unit repeating the message because of failure message	38
Figure 4.5: Central Station requesting two Data Mule Units.....	39
Figure 4.6: Received Signal Strength Indicator vs Distance	40
Figure 4.7: Signal to Noise Ratio vs Distance.....	40
Figure 4.8: 1- way Delay vs Distance	41
Figure 4.9: 2-way Delay vs Distance	41
Figure 4.10: The failure percentage with the distance plot.....	42

List of Tables

Table 1: Variation of temperature and salinity with depth	6
Table 2: Variation of propagation delay with depth	6
Table 3: Commercialized acoustic modems specifications	7
Table 4: JANUS bit allocation table	21
Table 5: Message structure	29
Table 6: Specifications of nodes on UnetStack simulator	36
Table 7: Received Signal Strength Indicator (RSSI) vs Distance	39
Table 8: Signal to Noise Ratio vs Distance	40
Table 9: Delays vs Distance	41
Table 10: The failure percentage with the distance	42

Abbreviations and acronyms

AUV	Autonomous Underwater Vehicle
CMRE	Centre For Maritime Research and Experiment
CRC	Cyclic Redundancy Check
CTM	Centre for Telecommunications and Multimedia
DTN	Delay Tolerant Network
FH-BFSK	Frequency-Hopped Binary Frequency Shift Keying
MCS	Modulation and Coding Scheme
RF	Radio-Frequency
ROV	Remotely Operated Vehicle
RSSI	Received Signal Strength Indicator
SINR	Signal-to-Interference-plus-Noise Ratio
SNR	Signal-to-Noise Ratio
UAN	Underwater Acoustic Network
UDMP	Underwater Data Muling Protocol
UWA	Underwater Acoustic
UWC	Underwater Wireless Communication

Chapter 1

Introduction

1.1 Context

Underwater Wireless Communications (UWC) have become in recent decades one of the most important technologies, especially in the oil and gas industries, and search and rescue operations [10]. This pushed not only the use of Autonomous Underwater Vehicles (AUVs) and Remotely Operated Vehicles (ROVs) underwater to envisage various military and commercial applications, but also the existence of broadband, cost-effective communications solution for these environments. Currently, there are three underwater wireless communications technologies available: acoustic, radio and optical, the three technologies are illustrated in Figure 1.1.

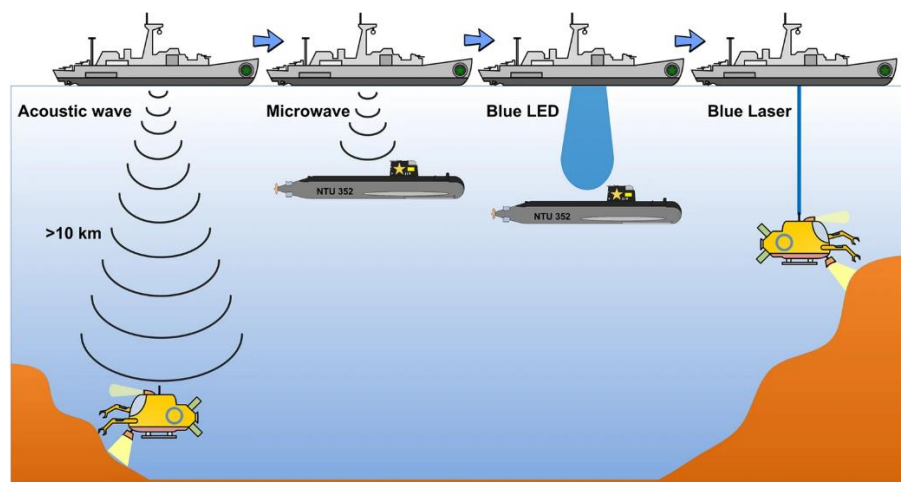


Figure 1.1: Underwater wireless communications technologies [8]

Acoustic, Optical, and Electromagnetic (RF) waves are used to envisage UWC techniques in underwater applications, but they are not very effective because of their limitations such as low bandwidth and high attenuation [4]. Optical communications provide high data rates, but they only deliver good performance in very clear water and require tight alignment. The RF electromagnetic waves propagate at

very high-speed underwater, but this technology is not very efficient due to high attenuation and losses in the water channel, limiting its range to a few meters. Acoustics is a proven technology for underwater applications, which offers long transmission ranges of up to 20 km but poor performance in shallow waters, where the transmission can be affected by turbidity, ambient noise, salinity, and pressure gradients [1,9].

Due to the harshness of the ocean and the propagation characteristics of the water, underwater broadband communications are limited to short-range applications. In a given mission underwater, AUVs can collect data in the order of Gigabytes, including video and bathymetric data. Due to the limitations of the current communications solutions, the vehicle is forced to upload the data only when it surfaces (e.g., at the end of the mission) and can deliver it to the Central Station Unit. This causes a high delay in data processing and increases energy consumption in the AUV. The GROW solution, which considers the use of data mules to enable data collection during the mission, is proposed to overcome the current limitation of underwater point-to-point communications [11].

1.2 Motivation

Due to the limitations of current and established wireless underwater technologies, the GROW solution [11], illustrated in Figure 1.2, aims to offer a system that uses data muling for wireless underwater broadband communications. The data mules, which are small and agile underwater drones, collect data from a Survey Unit, such as an AUV, and travel to the surface to deliver the collected data to a Central Station Unit. The data mules use a short-range broadband link (optical, RF) to exchange data. To enable the operation of the data mules, a long-range, low bitrate acoustic channel is required for control purposes.

In [7] a great achievement has been reached, it was shown that the use of data mules to transfer large files between the survey AUV and the Central Station Unit at surface (cf. Figure 1.2) is much more efficient than the traditional acoustic based solutions.

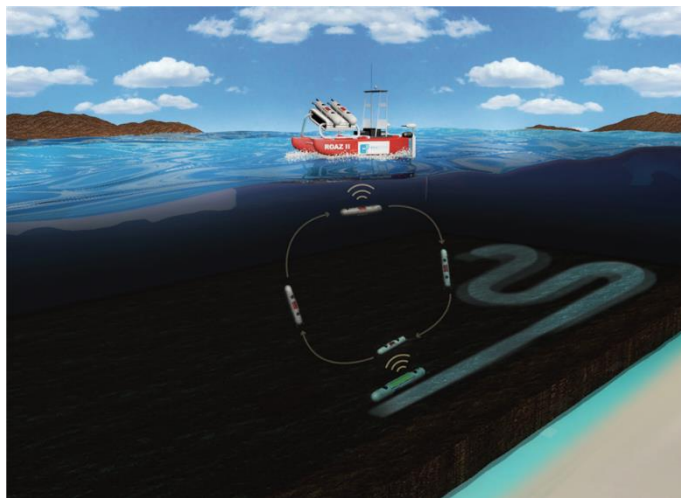


Figure 1.2: FCT GROW Solution [11]

In order to properly enable the data transmission, there is a need to implement a multi-node protocol for controlling the scheduling of the data muling process. The standard JANUS communications protocol developed by NATO is a good candidate to handle the MAC layer and allow to use its payload to exchange the control messages. The use of this protocol will allow the scheduling of data mules using long range acoustic communications.

By applying this strategy, the solution proposed in this dissertation will contribute to the control component of the GROW solution that aims to obtain a significantly higher bitrate and thus a much faster transmission of data when compared with acoustic communications.

1.3 Objectives

The main goal of this dissertation is to develop an out-of-band multi-node protocol for controlling the scheduling of the underwater data mules, taking advantage of the JANUS communications protocol developed by NATO. Then, it aims at implementing this protocol on UnetStack simulator to evaluate its ability to support the reliable control of the scheduling of the underwater data mules.

1.4 Main Contribution

The main contribution of this dissertation is an out-of-band multi-node protocol that enables the control and scheduling of the underwater data mules. The solution aims to advance the operation of unmanned underwater missions, improve the efficiency of data uploading and provide a benchmark of the acoustic point-to-point link for different metrics (e.g., RSSI, SNR, delay, and frame loss ratio) at different distances.

1.5 Document Structure

This document is organized as follows. Chapter 2 explores the existing underwater communications technologies includes a survey of different simulations and its features, and shows an overview of the JANUS Underwater Communications standard.

In Chapter 3, the solution proposed in this dissertation is explained. Chapter 4 presents the test scenarios and the implementation of the protocol within UnetSim, and shows the simulation results. Finally, in Chapter 5, some conclusions are drawn and the future work is pointed out.

Chapter 2

State of the Art

This chapter presents the state of the art related to underwater wireless communications technologies. Also, we introduce the UnetStack acoustic simulator and perform a comparison between different network channel models. The JANUS Underwater Communications standard is also explained.

2.1 Underwater Communications Technologies

In the present digital era, the benefits of short-range and high-bandwidth communications systems have become familiar to the users. Meanwhile, the oil industry, military, and environmental operations are demanding reliable, wireless, and short-range data link applications [1].

Underwater data exchange can be performed using three different technologies: Radio-Frequency (RF), acoustic waves and optical communications. Each approach has advantages and limitations. In the next three subsections, we give a brief overview of each of these types of underwater wireless communications.

2.1.1 Acoustic Wireless Communications

Acoustics is a proven technology for underwater sensor applications, which offers long transmission ranges of up to 20 km. However, the technology has limitations such as low data rates (in the order of kbit/s), high delays, high hardware costs, strong reflections and attenuation when transmitting through water/air boundary, poor performance in shallow water where the transmission can be affected by turbidity, ambient noise, salinity, and pressure gradients. In addition, acoustic technology can have an adverse impact on marine life [1, 3].

In [12] the authors have characterised the underwater acoustic channel based on the Propagation delay T_p , which is the time taken by the signal to travel from the transmitter to the receiver node in the network.

It can be calculated through the quotient between the distance (d), in meters from the destination and the transmitter, and the speed of underwater sound c (in meters/second), as we can see in Eq. 2.1.

$$T_p = \frac{d}{c} \quad (2.1)$$

This formula depends on the speed of sound formula (Eq. 2.2) which, in turn, depends on the Temperature (T) expressed in degree Celsius, Salinity (S) expressed in parts per thousand and Depth (D) expressed in meters.

$$c = 1449 + 4.6T + 0.055T^2 - 5.304 \times 10^{-2}T^2 + 2.374 \times 10^{-4}T^3 + 1.340(S - 35) + 1.630 \times 10^{-2}D + 1.675 \times 10^{-7}D^2 - 1.025 \times 10^{-2}T(S - 35) - 7.139 \times 10^{-13}TD^3 \quad (2.2)$$

Table 1 shows that as the depth of the sea is varied from 0 meters to 1500 meters, the temperature and salinity of water decreases, along with the sound speed [12].

Table 1: Variation of temperature and salinity with depth

Sr. No.	Depth (meters)	Temperature (T) in °Celsius	Salinity (S) in ppt	Sound Speed (c) in meters/second
1	0	18	0.03745	1475
2	50	15	0.03602	1466
3	100	10	0.03534	1448
4	500	8	0.03511	1447
5	1000	6	0.03490	1446
6	1500	4	0.03405	1446

Propagation delay has been computed by the Eq. 2.1. Table 2 shows the value of propagation delay at a different distance between the transmitter and receiver [12].

Table 2: Variation of propagation delay with depth

Sr. No.	Depth (D) in meters	Sound Speed (c) in meters/second	Propagation delay at 100-meter distance (seconds)	Propagation delay at 200-meter distance (seconds)
1	0	1475	0.06778	0.13555
2	50	1466	0.06821	0.13642
3	100	1448	0.06905	0.13810
4	500	1447	0.06913	0.13825
5	1000	1446	0.06914	0.13827
6	1500	1446	0.06916	0.13832

The authors of [12] also have characterized the underwater acoustic channel by other two parameters Transmission loss (T_L) and Spreading Loss (P_L).

Transmission loss is the decrement in sound intensity through the path from transmitting node to receiving node in the network, which depends upon the transmission range and attenuation. The T_L can be obtained using Eq. 2.3.

$$T_L = SS + \alpha \times 10^{-3} \quad (2.3)$$

where α is an attenuation factor in dB expressed as in Eq. 2.4 and SS is the spherical spreading factor that can be calculated by Eq. 2.5.

$$\alpha = \frac{0.11f^2}{1+f^2} + \frac{44f^2}{4100+f^2} \quad [dB/km] \quad (2.4)$$

$$SS = 10 \log(r); \quad (2.5)$$

where r is the range in meters and f is the frequency in kHz.

Spreading loss is a type of transmission loss, which occurs at the time when sound travels away from the source to destination. Spreading loss in dB is expressed as in Eq. 2.6.

$$PL_{spreading}(r) = K \times \log(r) \quad (2.6)$$

A survey of acoustic systems commercialized is presented in Table 3 [14].

Table 3: Commercialized acoustic modems specifications

Model	Distance (m)	Rate (kbit/s)	Operating Frequency (kHz)	Power (Watts)	Depth (m)
LinkQuest UWM1000	350	9.6 to 19.2	26.77 to 44.62	2	Up to 200
LinkQuest UWM2000	1500	9.6 to 19.2	26.77 to 44.62	8	Up to 4000
LinkQuest UWM3000	5000	2.5 to 5	7.5 to 12.5	12	Up to 7000
LinkQuest UWM4000	4000	4.8 to 9.6	12.75 to 21.25	7	Up to 7000
LinkQuest UWM10000	10000	2.5 to 5	7.5 to 12.5	40	Up to 7000
EvoLogics S2CR 48/78	1000	31.2	48 to 78	60	Up to 2000
EvoLogics S2CR 42/65	1000	31.2	42 to 65	60	Up to 2000
EvoLogics S2CR 18/34	3500	13.9	18 to 34	80	Up to 2000
EvoLogics S2CR 7/17	8000	6.9	7 to 17	80	Up to 6000

As shown in Table 3, the fastest acoustic modem can only transmit at 31.2 kbit/s at a 1000 m distance, while the one with the longer range (10 km) can transmit at 5 kbit/s. These data rates make acoustic communications impossible to be used for transferring large amounts of data such as videos, pictures or bathymetric information. Despite the low bitrate provided, acoustic communications are suitable for sending control messages, as considered in this dissertation.

2.1.2 Optical Wireless Communications

Optical communications is a solution that allows underwater short to medium range communications and high bitrate. Their very high capacity has recently stimulated several attempts at research on underwater optical communications [1].

As shown in Figure 2.1, visible light frequencies are the least attenuated in all of the electromagnetic spectrum. Wavelengths in the 470 nm range are, in general, the least attenuated, always depending on the characteristics of the water, since absorption and scattering are influenced by the chemical and biological makeup of the water [14].

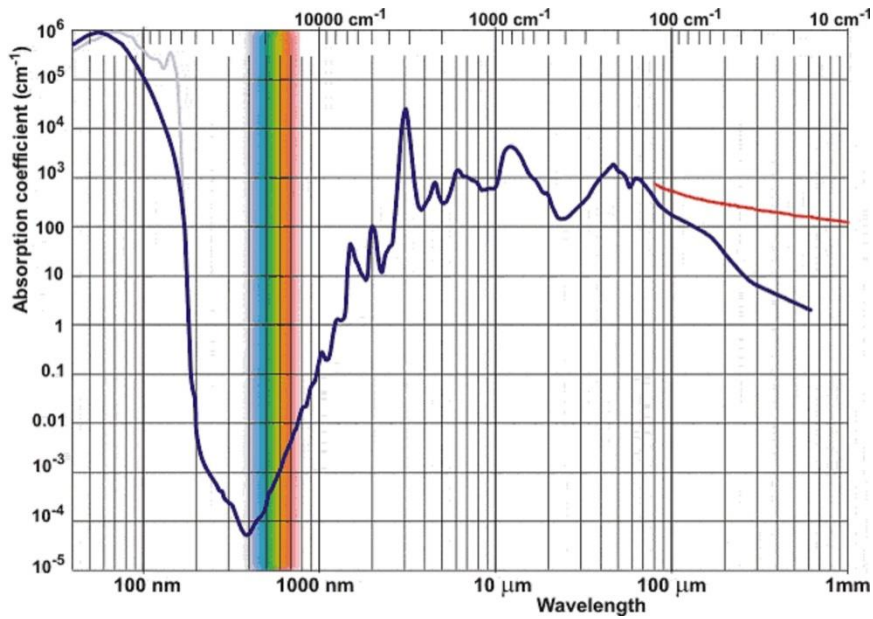


Figure 2.1: Absorption coefficient of electromagnetic radiation at various wavelength [14]

In Figure 2.2, it is possible to observe the minimum of underwater absorption is in the blue-light wavelength, and so, used by many commercial systems and experiments [13].

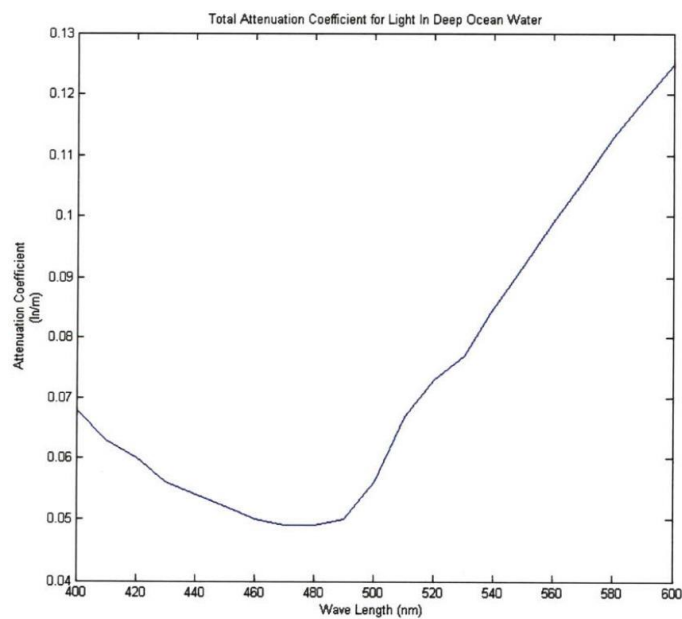


Figure 2.2: Underwater absorption coefficients visible light wavelengths [13]

The limitation of optical communications rely on the fact that light, being an electromagnetic wave, is strongly attenuated in water, which compromises the transmission range. An intrinsic limitation of optical communications is that they depend strongly on the line-of-sight so optical waves only deliver good performance in very clear water, which has imposed a significant constraint on its underwater applications [9].

2.1.3 Radio Frequency Wireless Communications

Radio Frequency waves are electromagnetic waves with frequencies below 300 GHz [1]. The electromagnetic spectrum propagates as a periodic disturbance of the electromagnetic field when an electric charge oscillates or accelerates. The electromagnetic spectrum is illustrated in Figure 2.3 [14].

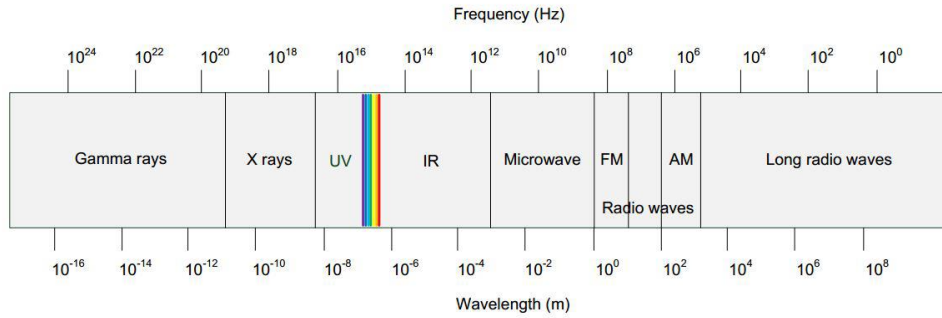


Figure 2.3: Electromagnetic Spectrum [14]

Although RF communications are not affected by turbidity and acoustic noise, and they do not require line-of-sight and propagate in underwater with very high speed, they have very limited range.

Electromagnetic communication technology is not very efficient due to large attenuation and losses in the water channel [1]. This fact implies short communication distances between devices, so EM is never chosen for long-distance communications [14]. The large refraction angle produced by the high permittivity launches a signal almost parallel with the water surface. This effect aids communications from a submerged station to the land and between shallow submerged stations without the need for surface repeater buoys [1]. However, at short distances, they enable the transmission of large amounts of data and are easily integrated in underwater devices such as observatories and AUVs.

2.2 Underwater Network Simulation

Simulation is an important part of the development and empirical evaluation of underwater acoustic network (UAN) protocols. The development, testing and validation of UAN protocols involves two main steps: simulations and sea experiments. The key feature of a credible network simulation model is a realistic channel model [15]. We will look into this in more detail in the next subsection.

2.2.1 Acoustic Channel Models

Generally, the channel models found in the UAN protocol literature can be split into three categories [15], as follows:

- **Binary range-based model**

There are different ways to model a UAN communication environment, but the simplest way is to derive a binary connectivity pattern among the nodes based on a fixed connection range (e.g., if the distance

between any two nodes is less than the maximum connection range, there is a link between them) and to assume a fixed propagation speed of 1500 m/s. This approach is oversimplifying the behaviour of a realistic Underwater Acoustic (UWA) channel and useful for theoretical UAN protocol development.

- **The analytical transmission loss model**

This model calculates the transmission loss on every link using mathematical expressions for distance-related spreading loss and frequency-related absorption loss. In contrast with the range-based model, it gives a measure of the received signal strength, allowing the researchers to estimate the Signal-to-Noise Ratio (SNR) and the Signal-to-Interference-plus-Noise Ratio (SINR). However, this model still omits many typical features of UWA channels, e.g., shadow zones due to acoustic wave refraction, delay spread and frequency selective fading due to multipath.

- **Specialized channel modelling software**

To take the previous approach a step further and to model more advanced characteristics of the UWA channel listed in the previous paragraph, specialized simulation models are required, e.g., based on ray/beam tracing or normal mode calculations. A popular open-source platform for this is BELLHOP, which employs beam tracing to predict acoustic pressure fields in specified underwater environments.

2.2.2 Underwater Acoustic Network Simulation Platforms

The authors of [15] provided a comparison between existing channel simulators regarding the features, capabilities and relative merits Of UWA channel simulation. In what follows, we present a description for each simulator, focusing on the main purpose, and the advantages and disadvantages of the simulator.

- **BELLHOP**

The main purpose of the BELLHOP simulator is that it is acting as a beam tracing model of UWA propagation, which has advantages such as it is well-established and verified, widely used as the channel model in UAN simulators and provides clear graphical insight into underwater acoustic propagation features. On the other hand, there are some disadvantages such as steep learning curve in underwater acoustics and typically requires software development by the user to adopt it in their research [29].

- **KRAKEN**

The main purpose of the KRAKEN simulator is that it is a normal mode model of UWA propagation which brings some advantages like being more appropriate than beam tracing for low-frequency propagation modelling. Meanwhile, there are some disadvantages as being less intuitive than beam tracing and not necessary for high-frequency propagation modelling [15].

- **VirTEX**

The main purpose of the VirTEX simulator is that it is a virtual signal transmission through a time-varying UWA channel (based on BELLHOP). Some advantages include taking into account the Doppler effect caused by node and sea surface motion and provides a more accurate representation of a UWA channel, compared with static BELLHOP. Contrary, disadvantages as being less applicable/feasible for UAN simulations with many point-to-point links [32].

- **Waymark**

The main purpose of the Waymark simulator is that it is a virtual transmission model through a time-varying UWA channel (similar to VirTEX). It has advantages such as having the same advantages as VirTEX, can integrate different UWA propagation models, other than BELLHOP and Not limited in the duration of a communication session. Meanwhile there are disadvantages as being less applicable/feasible for UAN simulations with many point-to-point links (similarly to VirTEX) [15]

- **WOSS**

The main purpose of the WOSS simulator is that it is acting as a network simulation using UWA channels modelled at specified geographical locations. It has some advantages such as it automates BELLHOP channel modelling in network simulations, uses real environmental data to model UWA propagation and integrates with C++ network simulators. Some disadvantages as less flexibility in channel modelling due to its automation and being limited to C++ network simulation tools (mostly used with ns2-MIRACLE) [30].

- **Aqua-Sim**

The main purpose of the Aqua-Sim simulator is that it is a UAN simulation platform based on ns-2, its advantages include integrating the ns-2 network simulator with a simple UWA propagation model, on the other hand, disadvantages as being limited to ns-2 simulations and less realistic UWA channel compared with WOSS [33].

- **DESERT**

The main purpose of the DESERT simulator is that it is a UAN simulation/emulation suite based on ns2-MIRACLE, it has some advantages as including mobility models to simulate node motion and includes an interface with WOSS for channel modelling. Some disadvantages as the limitation to ns2-MIRACLE network protocol simulations [34].

- **SUNSET**

The main purpose of the SUNSET simulator is that it is a UAN simulation/emulation suite based on ns2-MIRACLE, it has some advantages as it is designed to facilitate the easy transition between simulations and at-sea testing (more reliably than DESERT), and includes an interface with WOSS for channel modelling (same as DESERT). Other disadvantages as it is more complex than DESERT (for the transition from simulation to at-sea testing) and limited to ns2-MIRACLE network protocol simulations [31].

- **UnetStack**

The main purpose of the UnetStack simulator is that it is a UAN simulation/emulation suite with custom Java/Groovy and Python interfaces, some advantages as it is designed to make the simulation code portable to UnetStack-compatible acoustic modems and programmed in an agent-based framework for more efficient development. On the other hand, it has some disadvantages as being limited to the custom UnetStack software architecture and custom channel model is more difficult to implement than in DESERT/SUNSET [5].

2.2.3 UnetStack Simulator

UnetStack was developed under the Unet project at Acoustic Research lab of the National University of Singapore in 2004 [17]. The most noticeable characteristic of this testbed is that it is not based on the traditional layer-based protocol stack. Instead, UnetStack consists of a collection of software agents that provide well-defined services which result in a network stack that is flexible and allows software-defined underwater networks to be rapidly designed, simulated, tested and deployed [16][5].

UnetStack takes Java Virtual Machine (JVM) as part of its component. Therefore, once the protocols and applications are developed and tested via simulation on UnetStack, the simulation code blocks can be used in any modem that is compliant with UnetStack [16]. UnetStack can be used on desktop/laptop computers to simulate underwater networks and test protocol performance [5].

The architecture of UnetStack is shown in Figure 2.4. The architecture includes UnetStack agents, Java VM, and a modem. The agents in the stack provide well-defined functionalities similar to the layers in the traditional network protocol stack [17]. Generally, the agents communicate with each other via different types of messages, such as to request, response, and notifications.

UnetStack also supports high-level communications to monitor or control other agents. Besides message-based agent-to-agent communication, also supports message broadcasting service, by which a set of subscribed agents receive messages of a certain topic to which are subscribed, hence it enables researchers to develop, test, and add new functionalities to a process that was not supported in the traditional layer-based network architecture.

Figure 2.5 shows the architecture of UnetSim, where each node can communicate with the other nodes through physical agents deployed on them. The researchers can access the protocol stack directly or

remotely via an open-source fjage agent framework with text-based commands and can use the real hardware modem via the physical agent (driver) when it is in simulation mode [16].

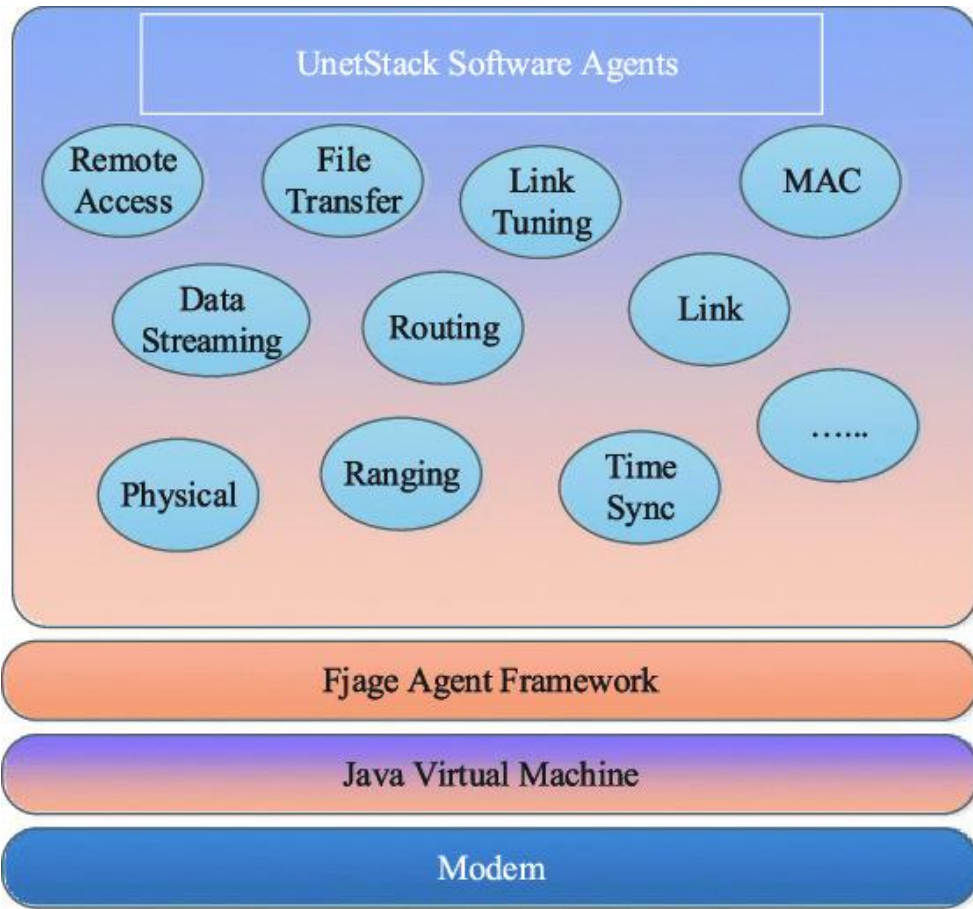


Figure 2.4: The architecture of UnetStack [17]

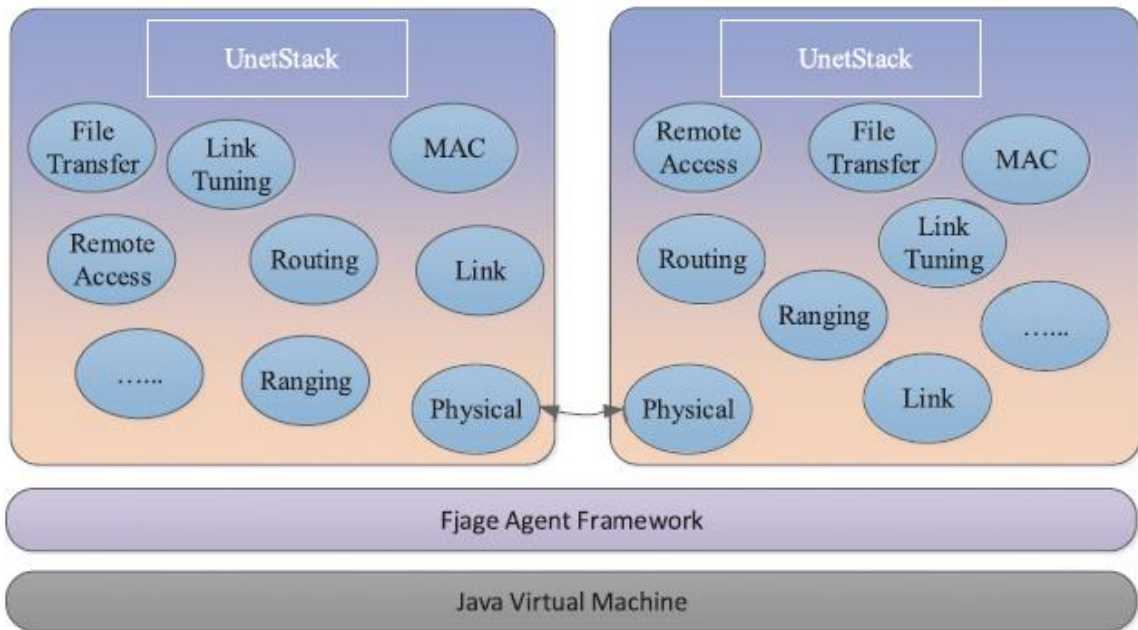


Figure 2.5: The architecture of UnetStack-based UnetSim [16]

- **Successful experiments on UnetStack**

In this subsection, we review two successful experiments which used UnetStack simulation.

Mission 2012 experiment

The Mission 2012 experiment took place in Singapore waters in October 2012 [24]; simultaneously, two networks were deployed; namely a UNET network and a Seaweb network. As shown in Figure 2.6, UnetStack is deployed on 5 UNET nodes. The P21 Node deployed from a barge as a surface modem, and the 4 Nodes are bottom-mounted UNET-PANDA nodes as Figure 2.7.

The network is controlled by a surface modem accessed from a laptop; on the other hand, the bottom-mounted nodes are accessible acoustically. This experiment examined several functionalities of the UnetStack in particular; the remote access, physical, baseband, link, MAC, ranging, and transport.



Figure 2.6: Five network nodes deployed in Singapore waters. Locations P21, P22, P27, P28 and P29 correspond to the drop locations of the 5 equivalently numbered nodes [24]

Measuring the communication channel's statistical variability was one of the objectives of the experiment, along with the experiment tasks; the 5 nodes managed more than 41000 transmissions of data frames and channel probe signals. The baseband received signals of each reception were logged by the nodes, which enabled the analysis of the channel's variability after the experiment [24].

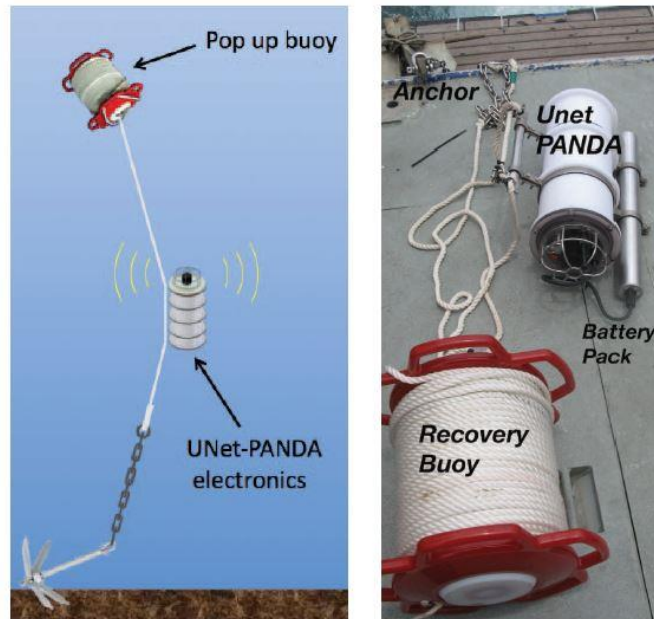


Figure 2.7: Typical configuration of a Unet-PANDA network node [24]

Mission 2013 experiment

The Mission 2013 experiment took place in Singapore waters in November 2013 [25]. Unlike the previous experiment of the Mission 2012, this 2013 experiment included more UNET and Seaweb nodes. Two autonomous underwater vehicles (AUVs) as mobile UNET nodes as shown in Figure 2.8, to ensure data flow between the UNET and Seaweb networks, a UnetStack was running on a gateway node [25].



Figure 2.8: The STARFISH AUV being deployed as a mobile network node during the MISSION 2013 experiment [6]

Seven static UNET nodes were deployed as shown in Figure 2.9 just one node -node 21- deployed from a barge as a surface modem, meanwhile, the others are bottom-mounted UNET-PANDA nodes limited to acoustic access only. During the experiment, testing each UnetStack agent required designing specific tests. Also, while the AUVs were moving across the network, dynamic communication was

performed using routing and route management services in UnetStack. Tracking and localizing the AUVs in a real-time were made by both time synchronization and OWTT ranging [25].

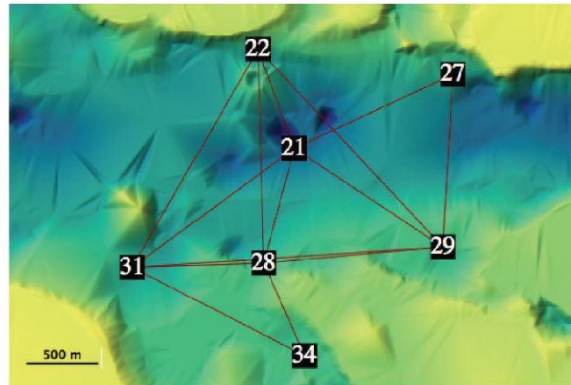


Figure 2.9: Seven static network nodes deployed in Singapore waters during the MISSION 2013 experiment [6]

In summary, after reviewing the different types of simulation and the real tests that validate its models, we can conclude that UnetStack is a suitable tool for performing simulations on the control channel of the GROW solution.

2.3 JANUS Underwater Communications Standard

JANUS is a simple multiple-access acoustic protocol designed and tested by the NATO Centre for Maritime Research and Experimentation (CMRE), to be used as the first standard to support digital underwater communications [20]. A very relevant feature is that JANUS is not intended to be limited to solely NATO military use, but also for civilian and international adoption [18]. The specification of the signal encoding and message format is fully available so that anyone may build a transceiver to communicate via JANUS to any other compliant platform [6].

2.3.1 Protocol Description

JANUS is a packet-oriented protocol, which transfers the file in "packets" or "blocks". It does not stop and wait for the receiver to acknowledge the reception of the packet. It simply assumes it was received correctly and immediately begin sending the next packet. If there was an error, the receiver would signal this back to the sender, and the bad packet would then be resent as soon as the current packet is completed [20]. The physical layer coding scheme is known as Frequency-Hopped (FH) Binary Frequency Shift Keying (BFSK). The reason of selecting FH-BFSK is its robustness and implementation simplicity [6].

The JANUS specification core feature is that once a frequency band is chosen, the chip duration C_d , wake-up tone duration (if present) and frequency slot width FS_w are calculated directly from the upper and lower band values, while the FH sequence and reverberation delay time remain constant for any band. Data

corruption is detected by an 8-bit Cyclic Redundancy Check (CRC). The coding operation sequence required to generate a Baseline JANUS Packet, without appended cargo data, is shown in Fig. 2.10.

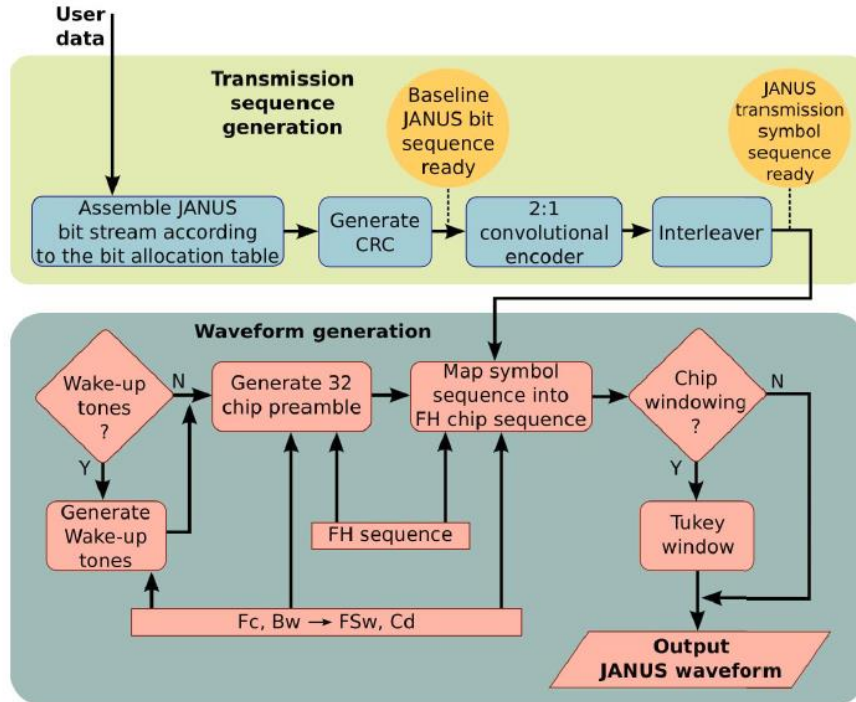


Figure 2.10: Block diagram for the JANUS baseline packet encoding process [6]

The standard provides for a "baseline JANUS Packet" (Figure 2.11) consists of an acoustic waveform that encodes 64 bits of information, which 34 bits may be user-defined according to their application. This approach provides almost unlimited flexibility in the nature and extent of the data to be sent [22].

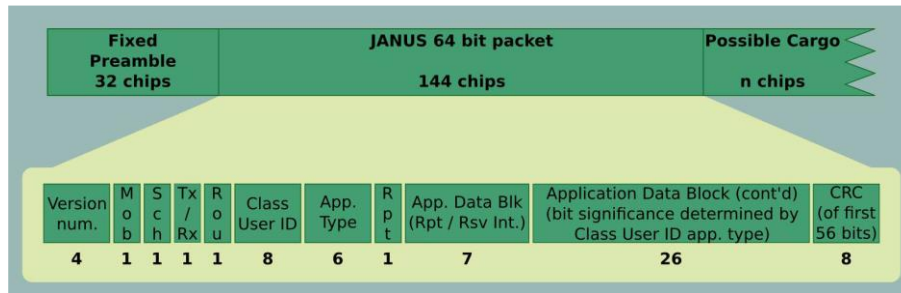


Figure 2.11: The structure of a JANUS packet [23]

2.3.2 Key Elements and Features

In this section, we describe main elements and features behind JANUS development.

- **Simple, robust design**

The main feature of JANUS design has always been the simplicity of implementation and robustness to challenging channels. The simple implementation concepts could be approved since there is no need for any type of advanced hardware or latest generation DSPs to implement JANUS in the manufacturer's

hardware. In fact, one may deploy JANUS on existing hardware. The robustness aspect typically comes at the cost of reduced data throughput, something that can well be accommodated in JANUS [18].

- **Openly available**

JANUS is open and free to all. Even though born from the efforts of NATO and CMRE, it attracted the broad international underwater communications community that has participated so strongly in JANUS's development [18].

In the JANUS wiki [20], there are references for a transmitter and receiver's implementations in both MATLAB and C++. NATO Industrial Advisory Group offering a practical guide to implement JANUS systems will be made available.

- **Flexible packet definition**

A baseline JANUS Packet consists of 64 bits of information. Besides control and specification bits, the "core" of the packet is primarily defined by the User Class ID, Application Type and Application Data Block [18].

- **Community engagement**

JANUS was built on community consensus. The fact that up to today, there is no single adopted underwater digital communications standard is proof to the fact that consensus is not easy to reach. The first JANUS community workshop was held back in 2008 and paved the way for the first implementation of JANUS [18].

- **Common frequency band(s)**

Figure 2.12 shows a time-frequency representation of a generic JANUS packet. A frequency hopping, frequency shift keying (FH-FSK) modulation scheme is employed as the base of JANUS. The JANUS waveform is fully parametrised based on the centre frequency [19] [23].

FH-BFSK has been selected for its known robustness in the harsh UW acoustic propagation environment and simplicity of implementation. A core feature of the JANUS specification is that once a frequency band is chosen, the Chip duration (Cd), Wake-up tone duration (if present) and Frequency Slot width (FSw) are calculated directly from the upper and lower band values, while the Frequency Hopping sequence and reverberation delay time remain constant for any band.

The JANUS packet may optionally be preceded by three "wake-up" tones that are intended for use in the cases where a modem needs to "wake-up" from a low power "sleep mode." The tones are followed by a short time to activate the receiver electronics, which precedes a fixed sequence of 32 chips.

This sequence is used as a detection and synchronization preamble. Once the fixed preamble phase of the waveform is complete, we are into the message section, which is composed by the "baseline JANUS Packet" followed by an optional "Cargo" section.

Increasing the center frequency will increase the bandwidth and possibly provide higher data rates. This will be achieved at the price of shorter communication range (due to frequency dependent absorption) and lower link reliability (due to smaller chip time). For the initial JANUS specification, a center frequency of 11 520 Hz was chosen, resulting in a frequency band between 9400 and 13 600 Hz and in a bit rate of 80 b/s. The choice of the initial frequency band for JANUS:

1. The 9–14-kHz band is attractive for a range of typical communication operational scenarios.
2. There are many devices operating in this frequency band, thus opening the door for the use of “hardware of opportunity” and the possibility of exploring the use of JANUS [27].

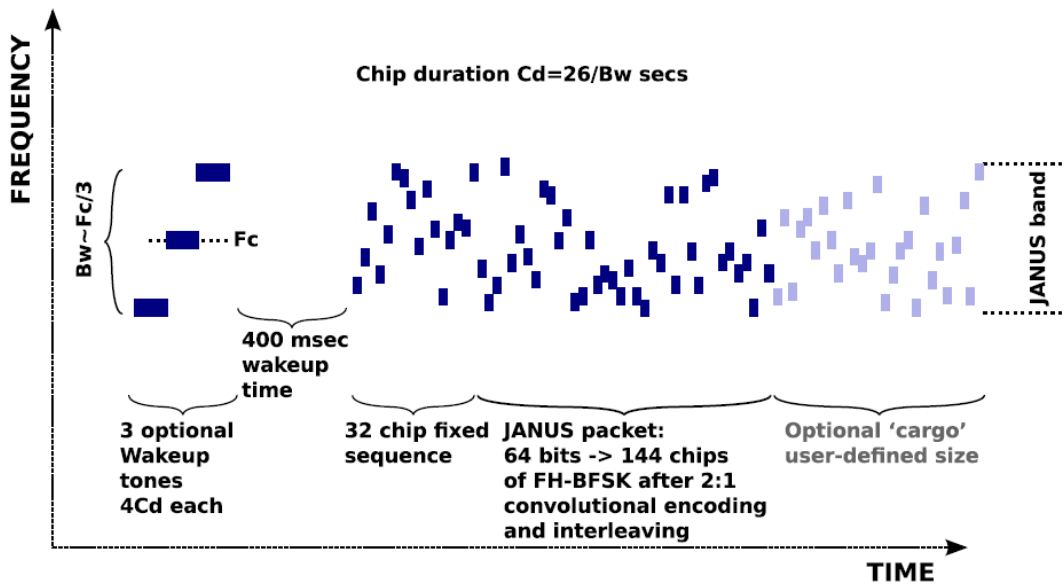


Figure 2.12: JANUS signal in a time-frequency plot [23]

- **Media Access Control (MAC)**

The JANUS MAC is species of Carrier Sensing Multiple Access (CSMA) with Collision Avoidance (CA) via Binary Exponential Backoff (BEB) with Global Awareness (GA). We take advantage of the frequency-scaling nature of the JANUS protocol to define parameters in chip lengths so that the Backoff and Carrier Sensing window lengths are scaled to the chip length [28].

1. Simple energy-detection scheme

To test whether the channel is 'busy' the node must have an estimate of the background (not busy) acoustic power in band. If the acoustic power in band over a window of 16 chip lengths exceeds the estimated background by more than 3 dB, the channel is deemed to be 'busy' in that window. If a node wishes to make a JANUS transmission, it must first carry out a background acoustic power in band estimation and then determine if the channel is 'busy'. If not, the node may transmit its JANUS message immediately. If 'busy', the node must apply a backoff before retrying. The JANUS protocol defines parameters in chip lengths so that the Backoff and Carrier Sensing window lengths are scaled to the chip length [28].

2. Binary Exponential Backoff (BEB)

If the channel is estimated to be ‘busy’ when a node intended to transmit, the node continues to sample windows of length $16 C_d$ until the channel is deemed no longer ‘busy’. The node then applies a BEB where it transmits in the next slot with probability P defined as:

$$P = \begin{cases} \frac{1}{1+2^{C-2}} & \text{for } 1 \leq c \leq 4 \\ \frac{1}{5} & \text{for } 4 < c \leq 8 \end{cases}$$

Where C is the number of potential transmissions slots the device has counted in the backoff process in which there has been at least one ‘busy’ window, initialized with $C=1$. The slot length is defined as the length of a Baseline JANUS Packet (i.e., $176 C_d$). If the node does not elect to transmit in the first available slot, it continues to sample 16-chip windows to detect if the channel is busy during the next slot, incrementing C by one (but only once per slot) if this is the case at any point during the slot, up to a maximum of $C=8$. Once the node elects to transmit its message, C is re-initialized to $C=1$. If C reaches 8, the attempt to transmit that packet is dropped [28].

- **Reservation Time**

The JANUS specification includes a built-in mechanism that allows participating nodes to reserve channel time. This means that with a basic JANUS packet, JANUS-compliant systems can silence neighbouring JANUS transmitters for a period of up to 10 minutes. To invoke the reservation of the channel, a node needs to be able to send JANUS messages with the correct bit flags in the baseline JANUS packets, i.e., scheduling bit (Sch) and repetition bit (Rpt) and reservation time encoded in the field as per the protocol specification [28].

- **JANUS Plug-Ins**

The usage of JANUS as a simple encoder/decoder of arbitrary payloads can be seen as the operation of an acoustic modem “talking” the JANUS language. To ensure interoperability in heterogeneous UANs, A baseline requirement is the ability to correctly read and write the content of JANUS messages. To support this capability while maintaining flexibility and modularity, the JANUS services have been developed using a design based on plug-ins. Plug-in writes and reads the appropriate bit sequence to/from the application data block and cargo section. Each plug-in uses a specific combination of the “user class ID” and “application type” fields to implement the respective data field translation [27].

- **Baseline JANUS Packet specification**

The coding operation sequence required to generate a Baseline JANUS Packet, without appended Cargo Data. It begins with the user data (determined by the user content, User Class and application specified by the user). The specification of each of the functional blocks is described in the following sub-sections.

A Baseline JANUS Packet consists of 64 bits of information, constructed according to Table 4. This packet includes a 34 bit ‘Application Data Block’ that is defined according to one of 64 possible schemes that may be specified by each User Class. There are 256 User Classes that are allocated to NATO organisations, NATO countries, other countries, specific organisations [27].

- **The Cyclic Redundancy Check specification**

Packet integrity is ensured by a Comité Consultatif International Téléphonique et Télégraphique (CCITT) 8-bit Cyclic Redundancy Check (CRC) which uses the polynomial $p(x) = x^8 + x^2 + x + 1$, initialized to 0. The 8 bits of the CRC are appended to the 56 bits of the main Baseline JANUS Packet as indicated in Table 4 [27].

Table 4: JANUS bit allocation table

Bits	Descriptor	0/1 bit set	Comments
1-4	Ver. #	0011	Unsigned 4-bit integer, Version 3
5	Mobility Flag	0=static 1=mobile	Indicates nature of the transmitting platform
6	Schedule Flag	0=off 1=on	If ‘On’, the first bit in the Application Data Block (ADB) indicates if the interval is to be interpreted as a reservation time (bit set to ‘0’) or a repeat interval (bit set to ‘1’). The time is specified from (different) look-up tables in bits 2-8 of the ADB, as specified in Annexes B & C
7	Tx/Rx Flag	0=Tx-only 1=Tx/Rx	Tx-only implies at least the ability to detect energy in band to satisfy the MAC requirements. Tx-Rx implies not only detect, but also decode capability.
8	Forwarding capability	0=No 1=Yes	Used for routing and Delay Tolerant Networking
9-16	Class user i.d.	[00000000:11111111]	Allows 256 classes of users, mostly individual nations (see Annex A)
17-22	Application Type	[000000: 111111]	Allows 64 different types of messages per user i.d. class – user specified
23-56	Application Data Block	Determined by user	For scheduled transmissions (bit 6 =1) the first 8 bits are dedicated to defining the nature of the schedule (reserved or repeat interval) with time defined in seconds from a lookup table.
57-64	8-bit Cyclic Redundancy Check (CRC)	8-bit CRC run on the previous 56 bits; polynomial $p(x) = x^8 + x^2 + x + 1$, init=0	
64			

- **Optional precursor ‘wake-up’ tones**

The JANUS packet may optionally be preceded by three ‘wake-up’ tones, each of four times that of a single chip duration, without pause between the tones, at frequencies:

$$F_c - Bw/2, F_c, F_c + Bw/2 \text{ [Hz]}$$

in that order. These are intended for use where a modem needs to ‘wake-up’ from a low power ‘sleep’ mode. The tones are not expected to be used when the intended receiver is already ‘awake’. If used, the tones should finish 0.4 [s] before the main preamble to allow reverberant energy in band to fade and for the intended modem to ‘wake up’. All JANUS transmitters should have the capability of sending ‘wake-up’ tones. Its use is left at the discretion of the user unless the packet includes time reservation, in which case the packet should be sent with ‘wake-up’ tones to make sure that ‘sleeping’ devices are aware of the channel reservation [27].

- **Center Frequency, Bandwidth, Chip duration and Frequency Slot Width**

The JANUS standard is anticipated to be applied at different center frequencies, each with a symmetrical frequency band Bw of approximately $F_c/3$ (within $\pm 10\%$) to meet diverse environmental, range and application scenarios. The Bw is divided into 13 pairs of Frequency Slots, each of width $F_{Sw} = Bw/26$.

The baseline C_d is the inverse of the Frequency Slot width, $C_d = 1/F_{Sw}$. This provides a scalable communication standard for which higher frequencies will be associated with a larger Bw and F_{Sw} , with correspondingly shorter C_d and a higher data transfer rate, at the cost of decreased practical range underwater due to stronger absorption [27].

- **The Optional Data Cargo Payload**

The baseline JANUS Packet may be followed, without a break, by additional data, encoded according to the user specified application into a continuation of the FH sequence. Unless the published user application specifies otherwise, the same convolutional encoder and interleaver are to be used as for the main Baseline JANUS Packet. The Baseline packet and the cargo are always separately encoded and interleaved.

A sufficient but not much time to transmit any such cargo must have been reserved in the preceding Baseline JANUS Packet by setting bit 6 to 1, bit 23 to 0 and specifying the reserve time in bits 24–30. If there is an intention to reserve the channel for emergency communications, e.g., using an underwater telephone such as one that implements STANAG 1074, bit 6 may be set to 1, bit 23 set to 0 and bits 24 – 30 to [1 1 1 1 1 1], thus reserving the channel for the maximum period of 10 minutes, without the need to transmit any data cargo [27].

2.3.3 JANUS-based Capabilities

In the recent past, CMRE has investigated the use of JANUS to support different services to be used in maritime operations [19]:

1. First contact and language switching

In such scenarios, JANUS can be used as the *lingua franca* to support the interaction and cooperation of the various devices. Through the use of a standardized modulation and coding Scheme (MCS), discovery and language switching procedures can be adapted to optimize the network operations and make better use of the underwater medium and node capabilities/resources (Figure 2.13).

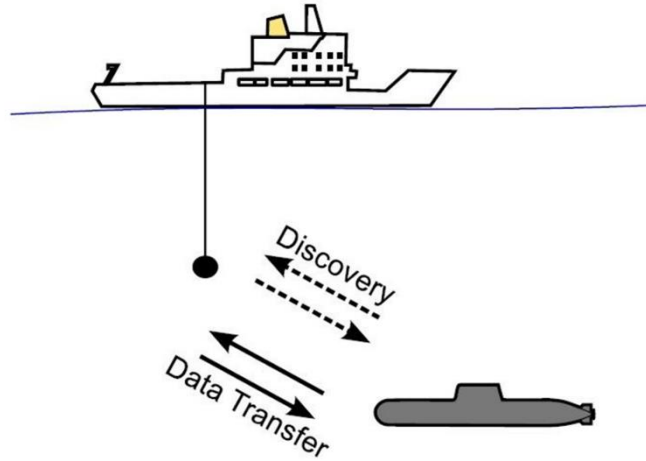


Figure 2.13: First contact and language switching [19]

2. Underwater AIS

The Automatic Identification System (AIS) is a status reporting service used to broadcast identification and localisation information to nearby vessels. The JANUS-based underwater AIS service can be applied in various scenarios and configurations in support for collision avoidance [23].

Underwater platforms could transmit their AIS data to inform the surface assets and other underwater vehicles about their presence. This would enable surface platforms to adapt their current navigation accordingly [21].

3. Underwater METOC service

Another capability of extreme value for the planning and execution of maritime operations is the availability of updated military meteorological and oceanographic (METOC) data.

2.3.4 JANUS Experimental and Results

To implement and test the designed service, a new JANUS plug-in was designed. The NIAG SG190 has investigated this use case for JANUS with the definition of three phases [19]:

1. Node discovery
2. Communication capability discovery
3. Language switching.

1. Node discovery

A node initiates the initial contact protocol by periodically broadcasting a discovery request. Another node that receives the discovery request can send back a reply (after a random delay) to make the requesting node, and other nodes in the area, aware of its presence, as presented in Figure 2.14.

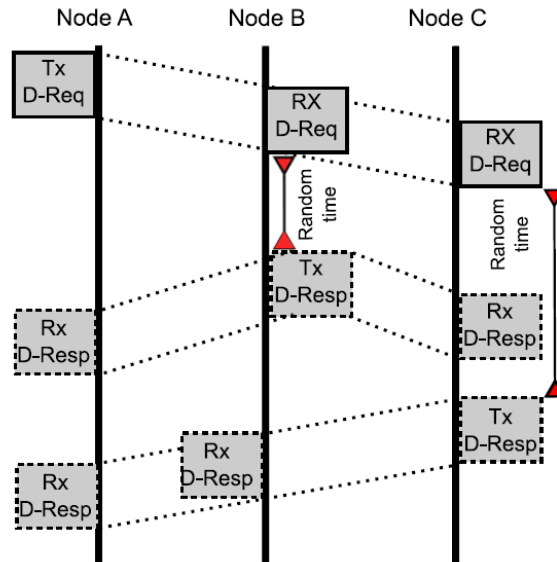


Figure 2.14: JANUS Node Discovery

2. Communication capability discovery

In this phase, a pair of nodes exchanges messages to agree on a common modulation scheme. One node initiates the process by sending a communications request message to the other node. This message includes a list identifying which modulation schemes it supports. The other node replies with a response listing which modulation schemes the two nodes have in common. If either of the packets is lost and the requesting node doesn't receive any response, it will retransmit the request when a timeout expires.

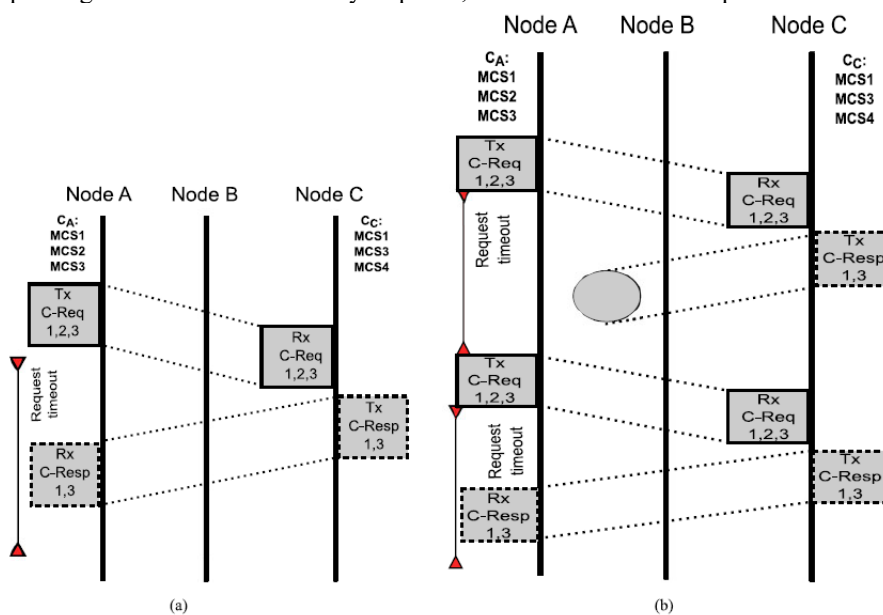


Figure 2.15: JANUS communication capability discovery

3. Language switching

Here, the requesting node from the previous phase decides which common modulation scheme (“language”) to use with the other node and sends it a language request message. The other node sends a response where it accepts or rejects the request. If it accepts, it also sends back values for timeouts which determine when to switch back to JANUS. If either of the packets is lost and the requesting node does not receive any response, it will switch back to JANUS after a timeout expires. The other node will also switch back to JANUS if it does not receive any data in the “new language” before a timeout expires, as presented in Figure 2.16.

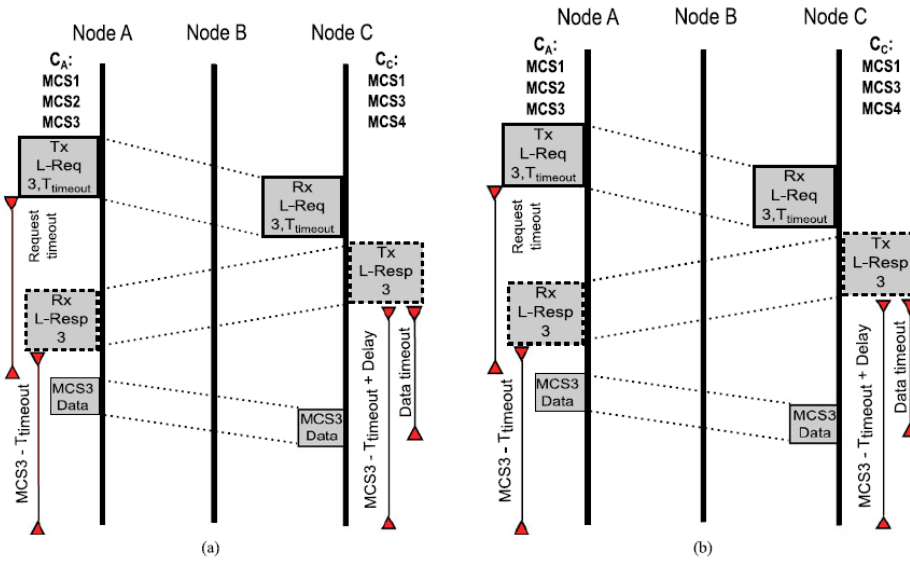


Figure 2.16: JANUS language switching

In summary, we close this section by reinforcing that JANUS can be used as a feasible solution to increase maritime situational awareness in the underwater domain, improving navigation safety, reducing the risk of collisions involving underwater assets.

Chapter 3

Proposed Solution

In this chapter we present the proposed solution for controlling the underwater data mules. The system architecture is explained, including the reliable protocol scenario, message body, message types and the message sequence diagrams are also presented.

3.1 Overview

Due to the harshness of the ocean and the propagation characteristics of the water, underwater broadband communications are limited to short-range applications. The main reason for this, is the limitations of current technologies for underwater communications to provide reliable and long-range broadband. GROW [11] is an innovative solution for Long-Range Broadband Underwater Wireless Communications.

GROW Solution, shown in Figure 3.1, takes advantage of concepts such as DTN and AUVs to implement a system in which a Survey Unit (stationary or mobile) collects data in the underwater environment, a Central Station Unit at surface works as a final receiver, and fast agile AUVs as Data Mules to transfer data between these two nodes. High bitrate short-range wireless communications are used to transfer the data between the data mules and the other nodes, and there is long-range acoustic out-of-band control link based on acoustic communications.

In previous work [13], the short-range advantages of the RF communications are used to transfer data between the nodes. In addition to that, an out-of-band acoustic link is used to aid the Data Mule Units scheduling and positioning. The Underwater Data Muling Protocol (UDMP) takes advantage of the permanent acoustic link and establishes a set of messages that are exchanged between the nodes, in order to coordinate the mules and allow the transfer of files using the tools available in the IBR-DTN implementation.

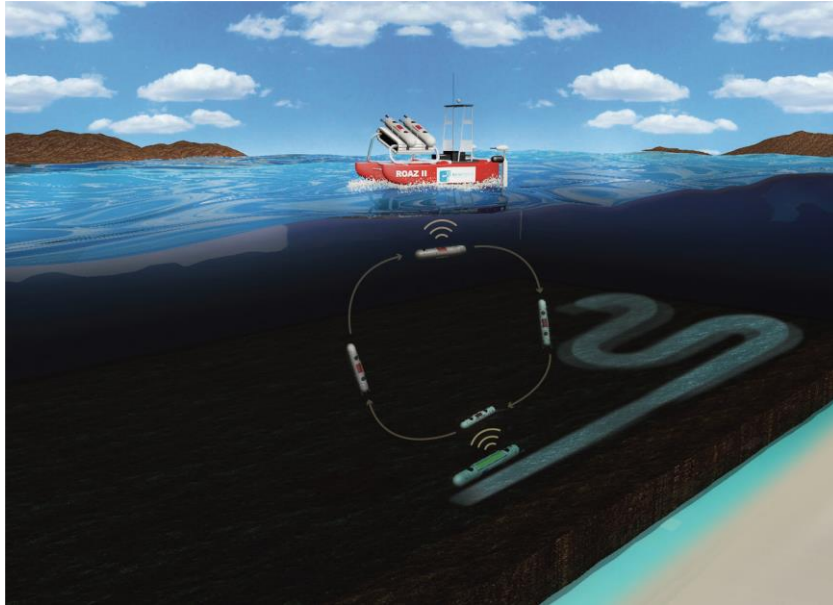


Figure 3.1: The GROW proposed Solution [11]

In this work we present an enhanced version of the UDMP protocol, adding a reliable solution for controlling the data mules in underwater scenarios based on JANUS underwater communications standard, improving the system that have already been used for regular file transfer in [7].

3.2 Reliable Underwater Data Muling Protocol

Due to the harshness and complex logistics of offshore missions, this dissertation aims to develop a reliable multi-node protocol that is an enhanced version of the UDMP presented within the GROW solution [11]. This protocol is designed to control the data muling process and allow to get the status of each Data Mule Unit during the process. This new version is built on top of the JANUS communication standard, which will handle the MAC layer functions and allow interoperability between modems of different vendors. It also avoids collisions between the messages and retransmissions in case a collision occurs.

In our solution, the data is transferred using two data mules between the survey unit and the central station taking advantage of JANUS protocol, these data were the payload data in the JANUS frame. These data is presented as different messages between the nodes, for each message has been sent, it should be received at the right destination within a timeout, if a node has sent a message and is expecting a response message from anther node and it does not receive it within the timeout, it repeats sending the request message again to make the transfer process continue successfully without missing any data, and by applying this way of exchanging data, we avoid missing data due to a collision between messages or other reasons like losing a Data Mule Unit or discharge of battery.

3.3 Protocol messages

The Underwater Data Muling Protocol (UDMP) which was previously designed in [13] is a communications protocol for data muling scenarios that enables the control and execution of the scheduling

of the Data Mule Units. The messages of this protocol are prepared for scenarios where we want to exchange files between nodes. In this work we propose an enhanced version of UDMP to make this protocol more reliable, including new message types and applying a retransmission mechanism to cope with packet losses and collisions.

The new version of the protocol will be used with the long rang acoustic links. The message structure differs from the message structure that was proposed in the UDMP. The complete message is built upon a JANUS frame and the payload of the JANUS frame as shown in Table 5.

- The JANUS Frame is composed of 64 bits, and it is generated by the simulator when we choose to use the JANUS channel.
- The payload consists of Source ID, Destination ID, Type, and Body, the maximum size of the cargo data of JANUS frame is 960 bits.

The Source ID and the Destination ID are the address of the nodes and has size 1 byte, the field type is referring to the type of message, used to differentiate the message data and has a fixed size of 1 byte. The Length field specifies the size of the body field. Finally, the Body field contains the data of the message, and its size is specified on the Length field.

Table 5: Message structure

Janus Frame	Source ID	Destination ID	Type	Body
64 bits	8 bits	8 bits	8 bits	936 bits

UDMP already had message types defined for almost every basic operation needed in these scenarios and we have added 4 more messages (Type 10, Type 12, Type 13, and Type 14). The all message types are as following:

Type 01: `get_data_size()` - Used to query the size of the file requested by user.

Type 02: `data_size()` - Used to inform the size of the file.

Type 03: `number_of_mules()` - Used to inform the AUV of how many mules will be dispatched.

Type 04: `ack()` - Used to acknowledge the reception of the `number_of_mules()` message.

Type 05: `send_mule_req()` - Used to inform the Data Mule Unit of the file chunk id.

Type 06: `send_mule_resp()` - Used to acknowledge or deny the send mule request.

Type 07: `dock_req()` - Used to inform the mule arrival, the file chunk id and request to dock.

Type 08: `dock_resp()` - Used to acknowledge the dock request and allow it or not.

Type 09: `data_chunk_sent()` - Used to inform the Station Unit that the file chunk has been transferred to the mule

Type 10: `data_chunk_sent_ack()` - Used to acknowledge the reception of the `data_chunk_sent()` message.

Type 11: `data_received()` - Used to inform all the nodes that the file chunk has been received at the Station Unit.

Type 12: `data_received_ack()` -- Used to acknowledge the reception of the `data_received()` message.

Type 13: `Keep_alive`- Used to request that there is a connection between CS and Data mule.

Type 14: `Keep_alive_ack()`– Used to acknowledge that there is a stable connection.

In case the mule is too late while the docking process, we can use `Keep_alive` message and waiting the response from the mule to be sure that it didn't get lost. This happens as following, after the Data Mule Unit sends the dock response to the Central Station Unit, the Docking process is started and every hour the Central Station Unit is sending `Keep_alive` message to be sure that the process is going on correct way, then within a timeout the Data Mule Unit should response with `Keep_alive_ack` message as acknowledge that is still continuing the docking process, if the response failed to be received within that timeout, the Central Station Unit will send `Keep_alive` message again, if there is no response for the second time, the Central Station Unit will dispatch a new Data Mule Unit.

In case failure of one Data Mule Unit, the Central Station Unit will dispatch another Data Mule to continue the transfer process. The causes for this failure could be Data Mule Unit getting lost, being out of reach of the acoustic link, battery depletion or hardware failure.

3.4 Protocol message sequence

For better understanding the protocol, how it works and how the messages are exchanged between the nodes, we will go through the following figures and understand each case.

3.4.1 Simulation with a Single Data Mule Unit

This scenario is the simplest scenario since use only one Data Mule Unit to transfer the data between the Central Station and the Survey Unit.

The process is started by requesting the file size and after that the Central Station Unit informs the survey Unit of how many Data Mules are being dispatched, the Central Station requests the Data Mule by sending `send_mule_request()` message, after that the Data Mule Travels to the Survey Unit to collect the file chunk then travel back to the Central Station. When the Survey Unit deliver the data, it sends `Data_Chunk_Sent()` message as notification and the Central Station replays with `Data_Chuck_Sent_Ack()` as acknowledgment for receiving the message.

When the data is received from the mule at the Central Station, the nodes are informed with a `data_received()` message and replay with the `Data_received_Ack()` message. In this scenario, in case a response fails to be received after a given timeout, there is a retransmission mechanism that will retransmit the message one more time. Figure 3.3 presents the case while we use only one Data Mule, and it successfully was able to exchange the messages between the nodes. Figure 3.4 presents the scenario where there is failure for one message, where a response fails to be received after a given timeout, there is a retransmission mechanism that will retransmit the message one more time, in this case the failure for `Send_mule_1_req()` message as an example but in case of failure of any message, the same mechanism is applied.

To calculate the timeout, we need to calculate the total delay time between the nodes. The delay of a packet is calculated by adding the following four components: propagation delay, transmission delay, queuing delay, and processing delay. The total delay depends on the distance between the nodes, the cargo

data, and the bitrate. Since we are taking advantage of JANUS Protocol, we will consider the specification of JANUS frame, as seen in Figure 3.2

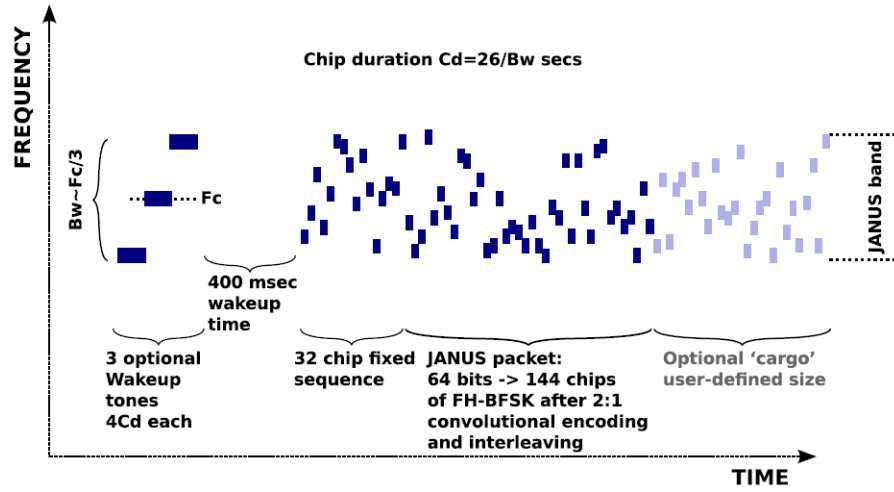


Figure 3.2: JANUS signal in a time-frequency plot [23]

According to the previous figure and in order to calculate the total delay, we need to calculate other time components

$$Total\ delay = T1 + T2 + T3 + T4 \quad (3.1)$$

Where:

$T1$: presents the time spending through the first three tones before the JANUS frame, each tone equals 4 Cd.

$T2$: presents the time needed to make the energy in the band faded.

$T3$: presents the transfer time for a JANUS packet with maximum size of our message.

$T4$: presents the propagation delay for long range communication

$$T1 = 3 * 4Cd \quad (3.2)$$

Where $Cd = 26/Bw$

Knowing that the JANUS signal uses only 1/3 of the central frequency as its bandwidth since the central frequency is 12000 Hz so the $Bw = 4000$ Hz.

So, the $Cd = 26/4000 = .0065$ s

$T1 = .078$ s

$$T2 = 400\ ms = .4\ s \quad (3.3)$$

$$T3 = \text{number of bits} / \text{bit rate} \quad (3.4)$$

Knowing the bit rate of Janus Frame = 80 bit/sec and the maximum number of bits of the JANUS packet with cargo data on UnetStack simulator = 1024 bits

$T3 = 1024/80 = 12.8$ s

$$T4 = \text{Distance} / \text{Sound speed} \quad (3.5)$$

$T4 = 8000/1500 \sim = 5$ s

To calculate the Timeout, we need to multiply the transfer time times 2 because it presents the time spending transfer data through two ways between the nodes.

$$Timeout = (T1 + T2 + T3 + T4) * 2 \quad (3.6)$$

$$Timeout = (.078 + .4 + 12.8 + 5) * 2 = 18.6 * 2 = 37.2 \text{ s}$$

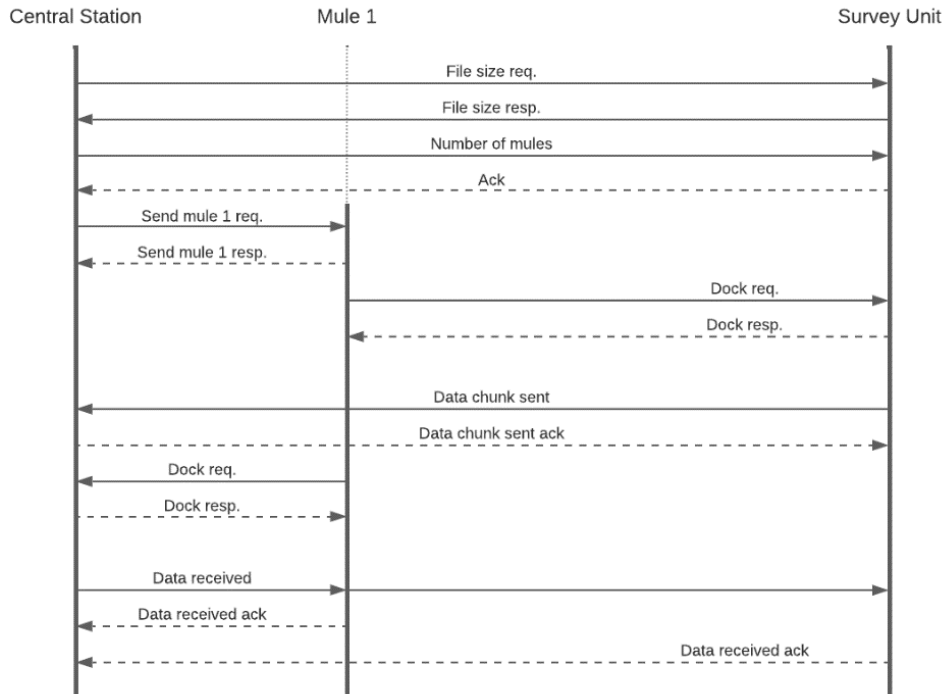


Figure 3.3: Message sequence diagram for 1 data Mule

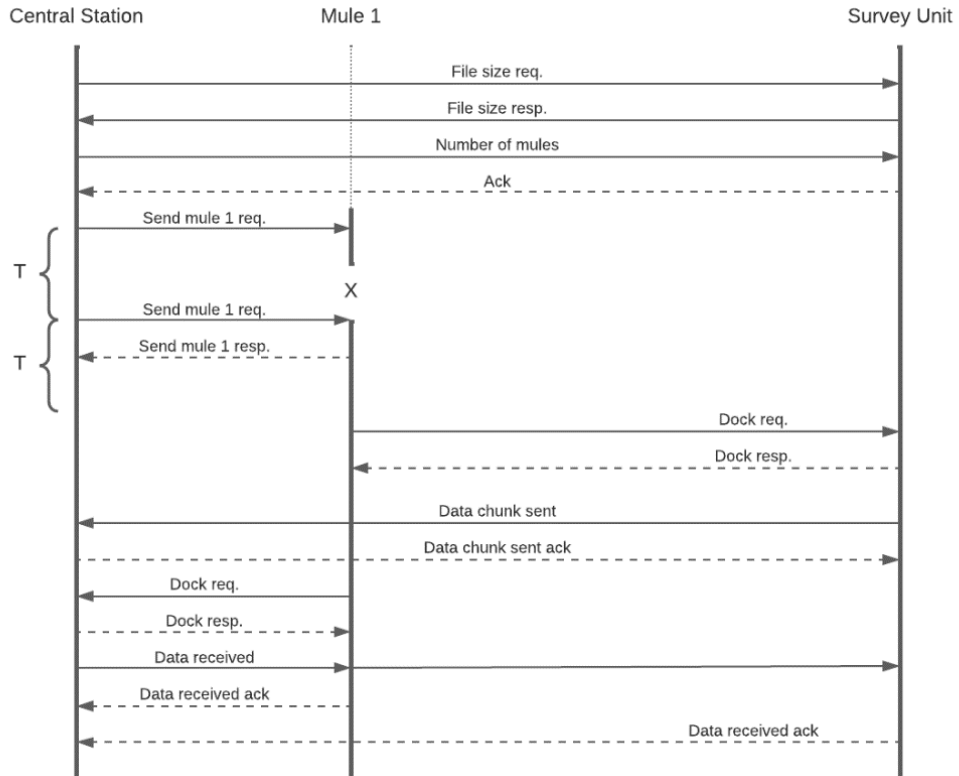


Figure 3.4: Message sequence diagram for 1 Data Mule Unit Failure Example

3.4.2 Simulation with Multiple Data Mule Units

In this scenario, the protocol is demonstrated with two Data Mule Units. The process is similar to the previous approach and the difference is that the Central Station is waiting to receive the data. Only after the data is received from both mules, the nodes are informed with a `data_received()` message and reply with the `Data_received_Ack()` message.

Figure 3.5 presents a successful case using two Data Mules, while Figure 3.6 presents a case when a response fails to be received after a given timeout, and a retransmission mechanism that will retransmit the message one more time. If it fails again, the Central Station Unit will dispatch another mule and follow the previous approach. In this scenario when the docking process is started, every hour the Central Station Unit is sending `Keep_alive` message to be sure that the process is going on correct way, then within a timeout the Data Mule Unit should response with `Keep_alive_ack` message as acknowledge that is still continuing the docking process, if the response failed to be received within that timeout, the Central Station Unit will send `Keep_alive` message again, if there is no response for the second time, the Central Station Unit will dispatch a new Data Mule Unit.

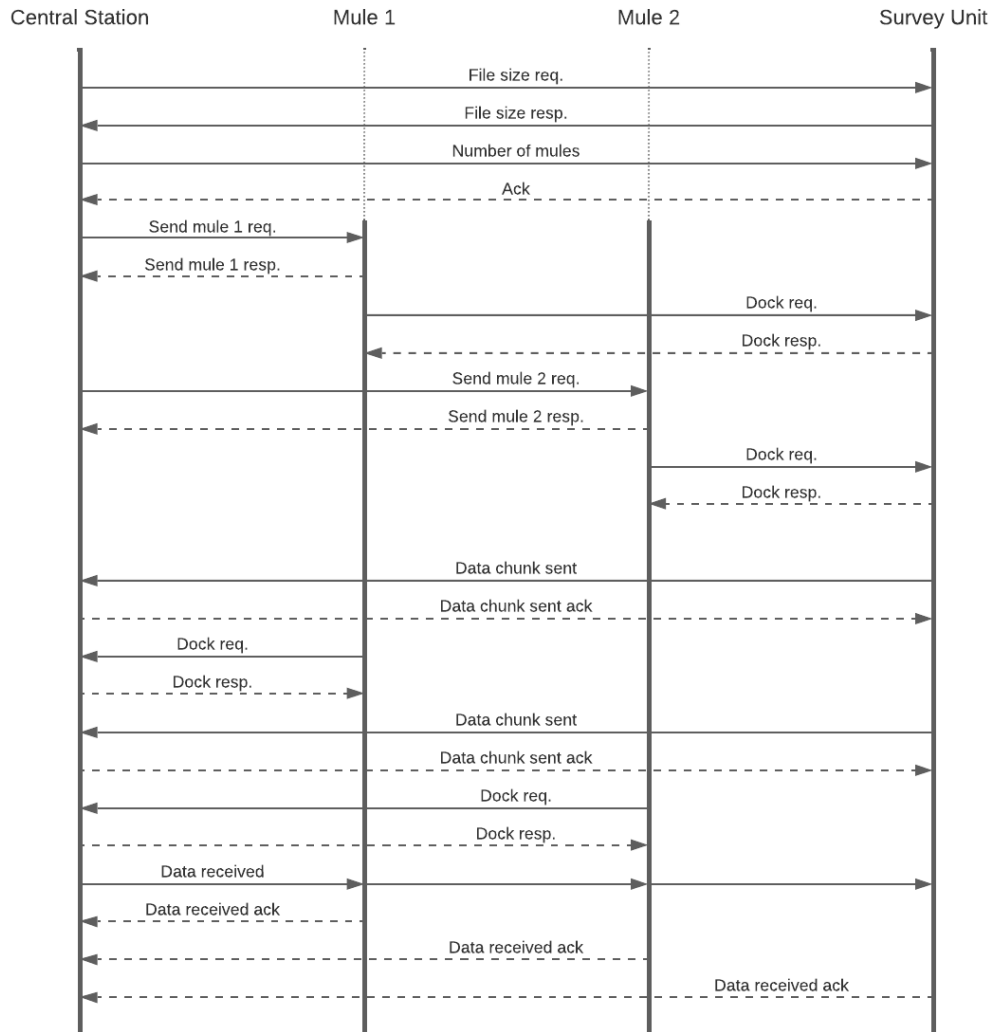


Figure 3.5: Message sequence diagram for 2 data Mules

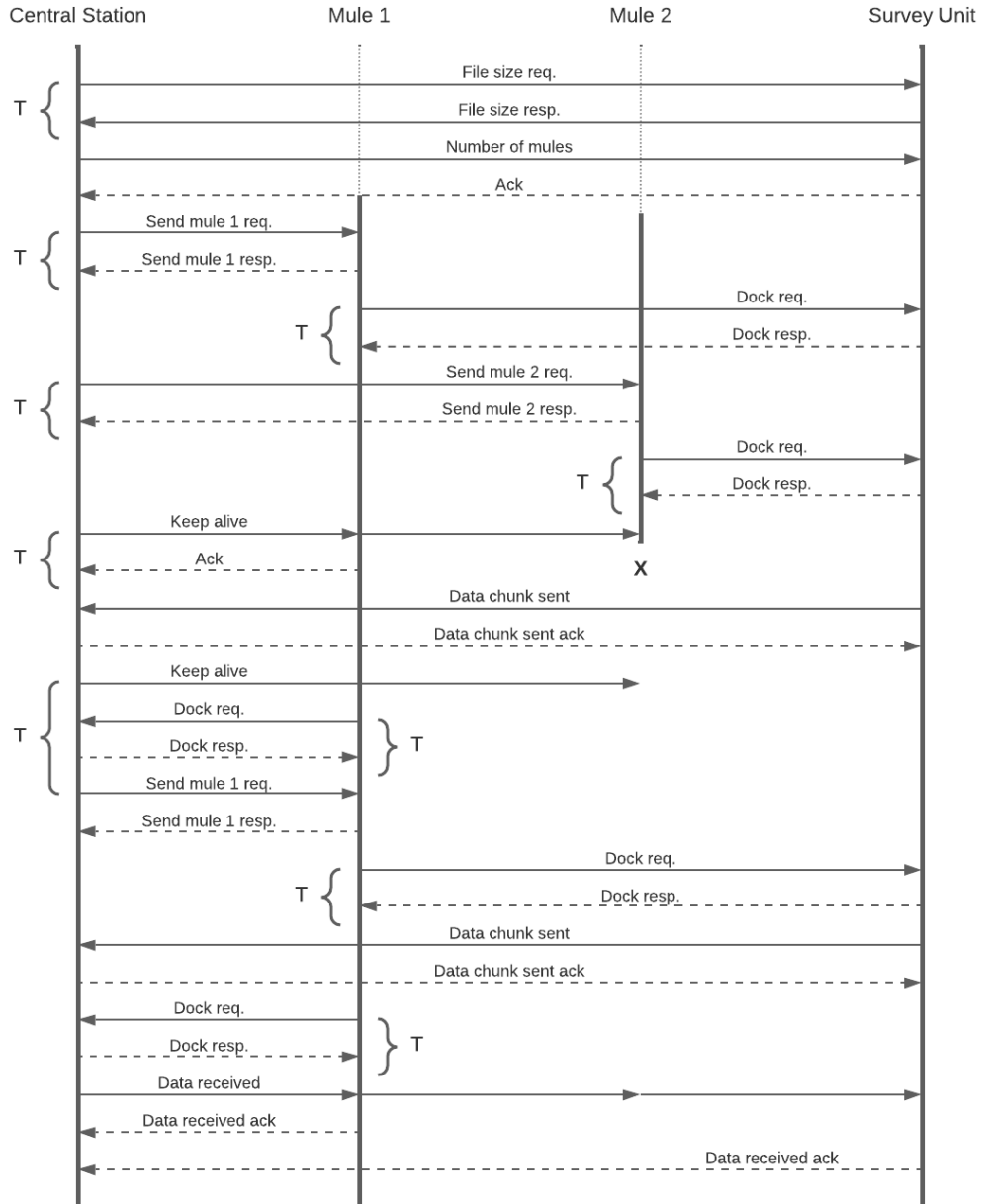


Figure 3.6: Message sequence diagram for 2 Data Mule Units failure

Chapter 4

Implementation and Experimentation Results

This chapter presents the implementation of the proposed solution and how we have tested it in the UnetStack simulator, as well as the explanation of the evaluation metrics.

4.1 Test Scenario and Network Simulation

We present the proposed solution using acoustic modems as four nodes, using two Data mules to transfer the data between the Central station at the surface and the Survey Unit at the sea bottom.

We have implemented the protocol in the UnetStack simulator where The UnetStack project strives to develop technologies that allow us to build communication networks that extend underwater, be it via acoustic, optical, or even wired links. We have written a main script to define the nodes and its location, and the simulator generates the tcp: local host and http: local host as shown in Table 6.

Table 6: Specifications of nodes on UnetStack simulator

Node name	tcp://localhost	http://localhost	Address
Central Station	1101	8081	20
Mule 1	1102	8082	30
Mule 2	1103	8083	40
Survey Unit	1104	8084	50

We have made three cases of simulation: one Data Mule Unit, one Data Mule Unit with failure, two Data Mule Units.

4.1.1 One Data Mule Unit

We have written three different scripts to run on the Central Station, Data Mule1 and the Survey Unit. This scenario is the simplest scenario which we have implemented the messages and run the simulation to

test transferring data between the nodes and we set a state for each message to be sure that the messages are receiving in the correct sequence, in the following Figures 4.1, 4.2, 4.3 we can see the examples of the simulation for different nodes.

```

Node address: 20

Sending File Size Request!
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:2908403641 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:2908403641]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:2935033362]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:2935033362 rssi:87.9 (5 bytes)]
Message is received with Correct destination
The message is: File Size Response!
Sending number of Mules ...
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:2937783641 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:2937783641]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:2949424362]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:2949424362 rssi:87.9 (5 bytes)]
Message is received with Correct destination
The message is: Ack!
Sending Mule 1 Request!
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:2952143641 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:2952143641]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:2959471426]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:2959471426 rssi:102.5 (5 bytes)]
Message is received with Correct destination
The message is: Mule 1 Response!
>
Node name : CentralStation
    
```

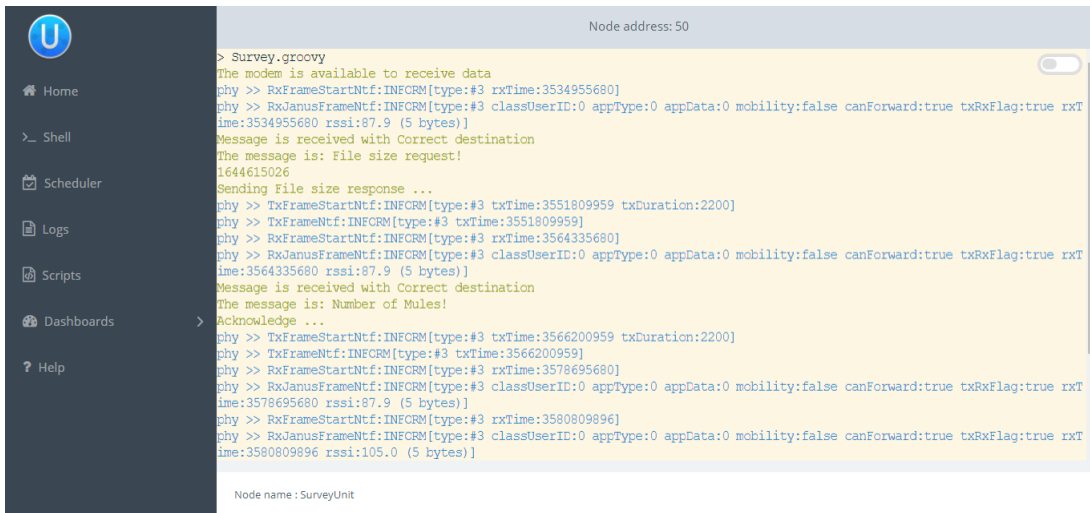
Figure 4.1: Simulation script on Central Station

```

Node address: 30

> mule1.groovy
The modem is available to receive data
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:765092658]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:765092658 rssi:102.5 (5 bytes)]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:786508810]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:786508810 rssi:105.0 (5 bytes)]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:794472658]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:794472658 rssi:102.5 (5 bytes)]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:800899810]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:800899810 rssi:105.0 (5 bytes)]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:808832658]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:808832658 rssi:102.5 (5 bytes)]
Message is received with Correct destination
The message is: Mule 1 Request!
Sending Mule 1 Response ...
Sending Dock Request ...
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:810946873 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:810946873]
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:819179873 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:819179873]
Node name : Mule1
    
```

Figure 4.2: Simulation script on Data Mule Unit 1



```

Node address: 50

> Survey.groovy
The modem is available to receive data
phy >> RxFrameStartNtf:INFCRM[type:#3 txTime:3534955680]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:3534955680 rssi:87.9 (5 bytes)]
Message is received with Correct destination
The message is: File size request!
1644615026
Sending File size response ...
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:3551809959 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:3551809959]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:3564335680]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:3564335680 rssi:87.9 (5 bytes)]
Message is received with Correct destination
The message is: Number of Mules!
Acknowledge ...
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:3566200959 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:3566200959]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:3578695680]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:3578695680 rssi:87.9 (5 bytes)]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:3580809896]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:3580809896 rssi:105.0 (5 bytes)]

Node name : SurveyUnit

```

Figure 4.3: Simulation script on Survey Unit

4.1.2 One Data Mule Unit with failure message

Similar to the previous simulation, we have written the scripts to run on the nodes but in this scenario, we faced a problem which is, The Central Station is not receiving a response from the survey within the given timeout and here we apply the mechanism to overcome this problem by forcing the Central Station Unit to send again the request message then the survey responded, and the process proceed correctly, in Figure 4.4 the Central Station Unit didn't receive the response for the message `File_size_request` within the timeout so, it sends the message again.



```

Node address: 20

> central.groovy
Sending File Size Request!
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:2994124846 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:2994124846]
Sending last message again
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:3004089846 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:3004089846]

Node name : CentralStation

```

Figure 4.4: Central Station Unit repeating the message because of failure message

4.1.3 Two Data Mule Units

In this scenario, we used two Data Mules to exchange the messages between the nodes, the process started by requesting the file size from Central Station to the Survey Unit, after that the Central Station informed the Survey Unit with the number of mules then requested both Data Mules by sending

Send_Mule_request messages to the Mules and when they responded, they began to exchange all the messages between the four nodes.

In Figure 4.5, we can see the Central Station requesting connecting with both Mules.

```

Node address: 20
> central.groovy
Sending File Size Request!
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:2502583528 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:2502583528]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:2510745009]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:2510745009 rssi:97.9 (5 bytes)]
Message is received with Correct destination
The message is: File Size Response!
Sending number of Mules ...
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:2513758528 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:2513758528]
phy >> RxFrameStartNtf:INFCRM[type:#3 rxTime:2521904009]
phy >> RxJanusFrameNtf:INFCRM[type:#3 classUserID:0 appType:0 appData:0 mobility:false canForward:true txRxFlag:true rxTime:2521904009 rssi:97.9 (5 bytes)]
Message is received with Correct destination
The message is: Ack!
Sending Mule 1 Request!
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:2524888528 txDuration:2200]
Sending Mule 2 Request!
phy >> TxFrameNtf:INFCRM[type:#3 txTime:2524888528]
phy >> TxFrameStartNtf:INFCRM[type:#3 txTime:2527143528 txDuration:2200]
phy >> TxFrameNtf:INFCRM[type:#3 txTime:2527143528]
> []
Node name : CentralStation

```

Figure 4.5: Central Station requesting two Data Mule Units

4.2 Simulation Results

In this chapter, we analyze the output of the simulator for different metrics, to evaluate the suitability of using an acoustic channel for controlling the data muling process. We evaluate four different metrics: RSSI, SNR, delay, and frame loss ratio for different distances.

4.2.1 Received Signal Strength Indicator (RSSI)

RSSI is a measurement of how well the modem device can hear a signal from another modem. It's a value that is useful for determining if there is enough signal to get a good connection. The RSSI value is good for making signal strength comparisons between different nodes and it is typically given in dBm.

In Table 7 and Figure 4.6, we can see the comparison of RSSI value for different node at different distance, we notice that when we have longer distance, we have less RSSI value and at distance 8 km between the nodes, we couldn't receive the signal.

Table 7: Received Signal Strength Indicator (RSSI) vs Distance

Distance between Central station and mule or Survey Unit (km)	RSSI at mule or the survey unit (dBm)
0.1	144.7
0.2	138.5
0.5	129.7
1.0	122.4
2.0	113.7
3.0	107.6
4.0	102.5
5.0	97.9
6.0	93.7
7.0	89.8
8.0	Bad Frame

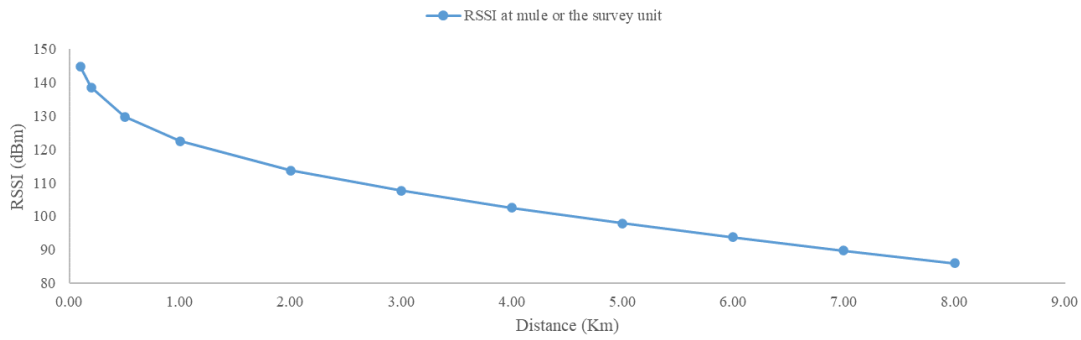


Figure 4.6: Received Signal Strength Indicator vs Distance

4.2.2 Signal to Noise Ratio (SNR)

SNR is a measure that compares the level of a desired signal to the level of background noise, it is expressed in dB, generally a signal with an SNR value of 20 dB or more is recommended for data networks. In UnetStack simulation the noise level is equal to 96 dBm and we can use the RSSI value and the noise to calculate the SNR according to the following equation.

$$SNR = RSSI - Noise$$

In Table 8 and Figure 4.7, we can see the relation between SNR and different distance.

Table 8: Signal to Noise Ratio vs Distance

Distance between Central station and mule or Survey Unit (km)	RSSI at mule or the survey unit (dBm)	SNR at mule or the survey unit (SNR = RSSI-Noise) (dB)
0.1	144.7	48.7
0.2	138.5	42.5
0.5	129.7	33.7
1.0	122.4	26.4
2.0	113.7	17.7
3.0	107.6	11.6
4.0	102.5	6.5
5.0	97.9	1.9
6.0	93.7	-2.3
7.0	89.8	-6.2
8.0	Bad Frame	-10

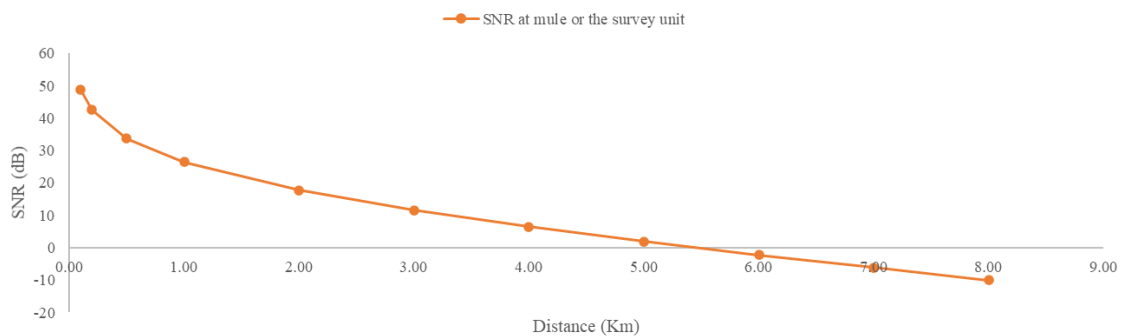


Figure 4.7: Signal to Noise Ratio vs Distance

4.2.3 Delay

This section presents how we calculated the time delay between two nodes. First, we calculate the real time before sending the message. Then, we calculate the real time after receiving the message on the other node. The difference between them is the delay time (1 way) between these two nodes, and the real time after receiving the response on the same node and the difference represents the time delay (2 ways) are calculated. At 8 km we could not exchange the messages because of the high noise level.

In Table 9, Figure 4.8, and Figure 4.9 we see the delay at different distances, 1 way from node to another node and the 2-way delay from a node sending a message to the same node receiving the response.

Table 9: Delays vs Distance

Distance between Central station and mule or Survey Unit (km)	Time at central station – 1 (s)	Time at central station – 2 (s)	Time at mule (s)	Time transfer (1 Way) (s)	Time Transfer (2 way) (s)
0.1	1644238812	1644238817	1644238814	2	5
0.2	1644238931	1644238936	1644238933	2	5
0.5	1644239153	1644239158	1644239156	3	5
1.0	1644237642	1644237648	1644237645	3	6
2.0	1644239211	1644237218	1644237215	4	6
3.0	1644239351	1644239360	1644239355	4	9
5.0	1644239491	1644239503	1644239497	6	12
6.0	1644239589	1644239601	1644239595	6	12
7.0	1644240502	1644240516	1644240509	7	14
8.0	1644240599	X	X	X	X

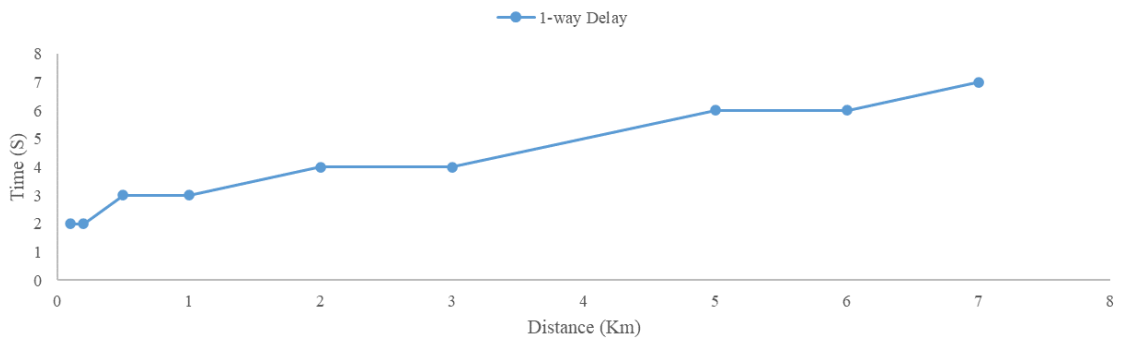


Figure 4.8: 1- way Delay vs Distance

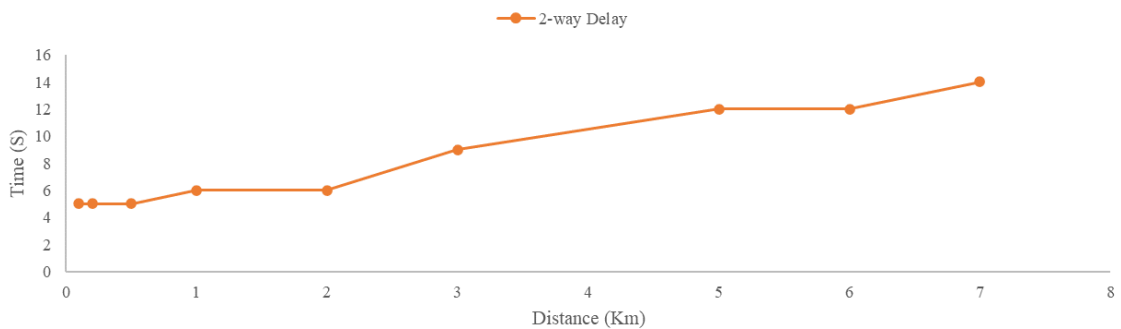


Figure 4.9: 2-way Delay vs Distance

4.2.4 Frame Loss

In this section, we explain the failure performance of exchanging the messages between the nodes at different distances, evaluating the failure percentage moving gradually from 100% connectivity at the maximum possible distance at 5km to 100% failure at 8km. To calculate more accurate percentage values of the frame loss, we have repeated the simulation one hundred times. Table 10 and Figure 4.10 shows the failure percentage with the distance.

Table 10: The failure percentage with the distance

Distance between Central station and mule or Survey Unit (km)	Failure Percentage
0.1	0%
0.2	0%
0.5	0%
1.0	0%
2.0	0%
3.0	0%
4.0	0%
5.0	0%
6.0	1%
7.0	8%
8.0	100%

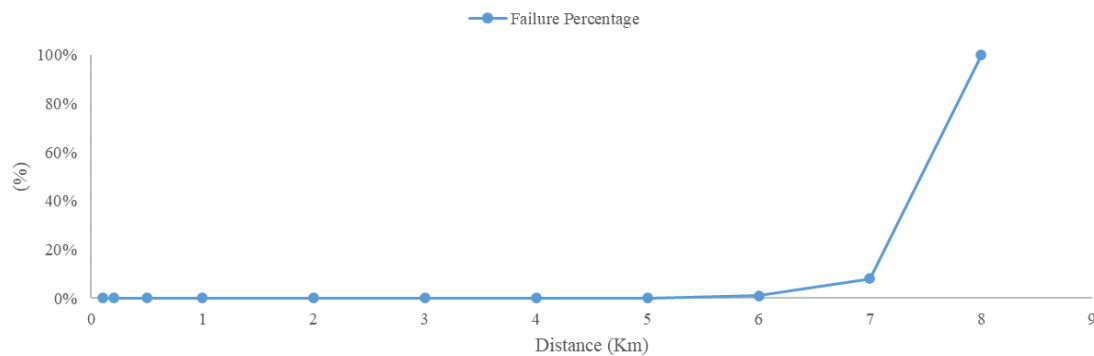


Figure 4.10: The failure percentage with the distance plot

As expected, exchanging the messages according to our protocol that is based on JANUS standard can improve the delay time that, the messages take less time to be transmitted between two nodes, and this delay varies according to the distance, we have noticed based on the simulations we made that, there is a positive correlation between the delay and the distance between the nodes; also we have noticed that there is a negative correlation between the SNR and the distance since the maximum distance that we can send the message with good quality is 5 km, distances which are longer than 5 km we got bad SNR value for them and the frame loss increases gradually until it reaches 100% failure at 8 Km. This was an expected behavior of the protocol and expected results, which prove the success of implementing the proposed protocol.

Chapter 5

Conclusion and Future Work

In recent decades, there has been a significant interest in studying and exploring the oceanic environment. AUVs have been widely used in underwater missions, allowing cost-effective and sustained ocean observations.

Underwater wireless communication is different from terrestrial wireless communication. It is by far more challenging to design underwater wireless links and also transmitters and receivers. An overview and comparison of the three major underwater technologies (radio, acoustic, and optical) indicate that they are not very effective because of their limitations.

To provide a broadband communications link underwater, we assume in this dissertation to use data mules to transfer large amounts of data using a DTN, as defined in the GROW Solution. A protocol to enable the scheduling of a set of Data Mule Units between a Central Station Unit and a Survey Unit was designed and implemented, taking advantage of an out-of-band acoustic channel. This protocol is built upon the JANUS standard and was designed, implemented, and tested through the UnetStack simulator.

The main contribution of the work is to provide an out-of-band multi-node protocol that enables the control and scheduling of the underwater data mules. We were able to validate the protocol for 1 and 2 data mules scenarios, with and without failure messages (retransmission process) and the link was possible up to 5 km. Also, that it was possible to evaluate the suitability of using an acoustic channel for controlling the data muling process and to characterize the wireless link on the metrics that we evaluated during the simulation, they are four different metrics: RSSI, SNR, delay, and frame loss ratio for different distances.

To conclude, we present the future work that can be explored in order to advance the area of underwater wireless communications.

- Apply the retransmission mechanism on several Data Mule Units scenario - In this dissertation we have only performed tests with the retransmission mechanism for the scenario using one data mule, but in future experiments we can consider more mules, for example, using 2 mules and if one of these mules fails to response within the given timeout, the central station can request the other mule.

- Addition of a keep alive message – in future work we can implement the keep_alive message to ensure that, if a mule fails to continue the docking process and instead waiting many hours until the process finishes, the Central Station Unit sends a message to be sure that the mule was not lost or faced any problem.
- Testbed validation - in this dissertation we only have simulated the protocol using UnetStack simulator but in future work, the creation of a testbed with real acoustic modems can help gather more accurate results to validate the real scenario and further developments on the control network.

References

- [1] X. Che, I. Wells, G. Dickers, P. Kear, and X. Gong, "Re-evaluation of RF electromagnetic communication in underwater sensor networks," *IEEE Communications Magazine*, vol. 48, no. 12, pp. 143–151, 2010, doi: 10.1109/MCOM.2010.5673085.
- [2] F. Teixeira, P. Freitas, L. Pessoa, R. Campos, and M. Ricardo, "Evaluation of IEEE 802.11 underwater networks operating at 700 MHz, 2.4 GHz and 5 GHz," in *Proceedings of the 9th ACM Int. Conf. Underw. Networks Syst. WUWNET 2014*, pp. 5–9, 2014, doi: 10.1145/2671490.2674571.
- [3] F. Teixeira, J. Santos, L. Pessoa, M. Pereira, R. Campos, and M. Ricardo, "Evaluation of underwater IEEE 802.11 networks at VHF and UHF Frequency bands using Software Defined Radios," in *Proceedings of the 10th ACM International Conference on Underwater Networks and Systms. WUWNet 2015*, pp. 1–5, 2015, doi: 10.1145/2831296.2831313.
- [4] F. Teixeira, R. Campos, and M. Ricardo, "IEEE 802.11 rate adaptation algorithms in underwater environment," *10th ACM International Conference on Underwater Networks and Systms. WUWNet 2015*, pp. 3–4, 2015, doi: 10.1145/2831296.2831312.
- [5] M. Chitre, R. Bhatnagar, and W. S. Soh, "UnetStack: An agent-based software stack and simulator for underwater networks," *2014 OCEANS. - St. John's, 2014*, doi: 10.1109/OCEANS.2014.7003044.
- [6] J. Potter, J. Alves, D. Green, G. Zappa, I. Nissen, and K. McCoy, "The JANUS underwater communications standard," *2014 Underwater Communications Networking, UComms 2014*, pp. 8–11, 2014, doi: 10.1109/UComms.2014.7017134.
- [7] F. B. Teixeira, N. Moreira, R. Campos, and M. Ricardo, "Data Muling Approach for Long-Range Broadband Underwater Communications," *International Conference on Wireless and Mobile Computing, Networking and Communications*, vol. 2019-October, pp. 4–7, 2019, doi: 10.1109/WiMOB.2019.8923472.
- [8] M. F. Ali, D. N. K. Jayakody, Y. A. Chursin, S. Affes, and S. Dmitry, *Recent Advances and Future Directions on Underwater Wireless Communications*, vol. 27, no. 5. Springer Netherlands, 2020.
- [9] T. C. Wu, Y. C. Chi, H. Y. Wang, C. T. Tsai, and G. R. Lin, "Blue laser diode enables underwater communication at 12.4 gbps," *Scientific Reports*, vol. 7, no. January, pp. 1–10, 2017, doi: 10.1038/srep40480.
- [10] D. Hossen, "A Survey of Acoustic Underwater Communications and Ways of Mitigating Security Challenges," vol. 4, no. June, pp. 43–51, 2016.
- [11] GROW project website. <https://grow.inesctec.pt> [Accessed: 7th February 2021].
- [12] Y. Kularia, S. Kohli, and P. Bhattacharya, "Analyzing Propagation Delay, Transmission Loss and Signal to Noise Ratio in Acoustic Channel for Underwater Wireless Sensor Networks," *1st IEEE Int. Conference on Power Electronics, Intelligent Control and Energy Sys.* 2016, pp. 1–5, 2016.
- [13] N. Moreira "Data Muling for Broadband and Long Range Wireless Underwater Communications," MSc thesis, University of Porto, June 2019. URL: <https://hdl.handle.net/10216/121806>.
- [14] P. Freitas. "Evaluation of Wi-Fi Underwater Networks in Freshwater," MSc thesis, University of Porto, 2014. URL: <https://hdl.handle.net/10216/75691>.
- [15] N. Morozs, W. Gorma, B. T. Henson, L. Shen, P. D. Mitchell, and Y. V. Zakharov, "Channel modeling for underwater acoustic network simulation," *IEEE Access*, vol. 8, no. July, pp. 136151–136175, 2020, doi: 10.1109/ACCESS.2020.3011620.
- [16] H. Luo, K. Wu, R. Ruby, F. Hong, Z. Guo, and L. M. Ni, "Simulation and experimentation platforms for underwater acoustic sensor networks: Advancements and challenges," *ACM Comput. Surv.*, vol. 50, no. 2, 2017, doi: 10.1145/3040990.

- [17] B. R. Chandavarkar and A. V. Gadagkar, "A framework for residual energy model in UnetStack simulator for underwater sensor networks," in *Proceedings of the 2020 International Conference on Computing, Communications and Security ICCCS 2020*, 2020, doi: 10.1109/ICCCS49678.2020.9277035.
- [18] J. Alves *et al.*, "Moving JANUS forward: A look into the future of underwater communications interoperability," *OCEANS. 2016 MTS/IEEE Monterey*, 2016, doi: 10.1109/OCEANS.2016.7761094.
- [19] R. Petroccia, J. Alves, and G. Zappa, "JANUS-based services for operationally relevant underwater applications," *IEEE Journal of Oceanic Engineering*, vol. 42, no. 4, pp. 994–1006, 2017, doi: 10.1109/JOE.2017.2722018.
- [20] JANUS wiki: <http://www.JANUSwiki.org> [Accessed: 22nd February 2022]
- [21] F. Ferreira, R. Petroccia, and J. Alves, "Increasing the operational safety of Autonomous Underwater Vehicles using the JANUS communication standard," *AUV 2018 - 2018 IEEE/OES Autonomous Underwater Vehicle Workshop in Proceedings of the 2018*, doi: 10.1109/AUV.2018.8729757.
- [22] R. Petroccia, G. Cario, M. Lupia, V. Djapic, and C. Petrioli, "First in-field experiments with a 'bilingual' underwater acoustic modem supporting the JANUS standard," *MTS/IEEE OCEANS 2015 - Genova 2015*, doi: 10.1109/OCEANS-Genova.2015.7271740.
- [23] R. Petroccia, J. Alves, and G. Zappa, "Fostering the use of JANUS in operationally-relevant underwater applications," *3rd Underwater Communications and Networking Conference Ucomms 2016*, no. Figure 1, 2016, doi: 10.1109/UComms.2016.7583424.
- [24] M. Chitre, I. Topor, R. Bhatnagar, and V. Pallayil, "Variability in link performance of an underwater acoustic network," *OCEANS 2013 MTS/IEEE Bergen Challenges North. Dimens.*, pp. 0–6, 2013, doi: 10.1109/OCEANS-Bergen.2013.6607953.
- [25] M. Chitre and G. Chua, "Modeling realistic underwater acoustic networks using experimental data," *Conference on Signals, Systems and Computers*, vol. 2015-April, no. node 21, pp. 39–43, 2015, doi: 10.1109/ACSSC.2014.7094392.
- [26] M. Y. I. Zia, J. Poncela, and P. Otero, *State-of-the-Art Underwater Acoustic Communication Modems: Classifications, Analyses and Design Challenges*, no. 0123456789. Springer US, 2020, doi: 10.1007/s11277-020-07431-x
- [27] NATO Standardization Office STANAG 4748 Ed. A Ver. 1" Digital Underwater Signaling Standard for Network Node Discovery & Interoperability 2017 URL: <https://nso.nato.int/nso/nsdd/main/standards?search=ANEP-87>
- [28] Smith, S. M., Park, J. C., & Neel, A. (1997). Peer-to-peer communication protocol for underwater acoustic communication. *OCEANS Conference Record (IEEE)*, 1, 268–272, doi: 10.1109/oceans.1997.634374
- [29] S. Gul, S. Sajjad Haider Zaidi, R. Khan, and A. B. Wala, "Underwater acoustic channel modeling using BELLHOP ray tracing method," *Proc. 2017 14th Int. Bhurban Conf. Appl. Sci. Technol. IBCAST 2017*, pp. 665–670, 2017, doi: 10.1109/IBCAST.2017.7868122.
- [30] F. Guerra, P. Casari, and M. Zorzi, "World ocean simulation system (WOSS): A simulation tool for underwater networks with realistic propagation modeling," *Proc. 4th ACM Int. Work. Underw. Networks, WUWNet '09*, no. November, 2009, doi: 10.1145/1654130.1654134.
- [31] C. Petrioli, R. Petroccia, J. R. Potter, and D. Spaccini, "The SUNSET framework for simulation, emulation and at-sea testing of underwater wireless sensor networks," *Ad Hoc Networks*, vol. 34, pp. 224–238, 2015, doi: 10.1016/j.adhoc.2014.08.012.
- [32] S. H. Oh, M. J. Eom, J. S. Kim, J. W. Han, and K. M. Kim, "Application of an underwater communication channel simulator to performance evaluation," *Int. Conf. Ubiquitous Futur. Networks, ICUFN*, no. 1, pp. 30–31, 2013, doi: 10.1109/ICUFN.2013.6614771.
- [33] P. Xie *et al.*, "Aqua-sim: An NS-2 based simulator for underwater sensor networks," *MTS/IEEE Biloxi - Mar. Technol. Our Futur. Glob. Local Challenges, Ocean. 2009*, pp. 1–7, 2009, doi: 10.23919/oceans.2009.5422081.
- [34] F. Campagnaro *et al.*, "The DESERT underwater framework v2: Improved capabilities and extension tools," *3rd Underw. Commun. Netw. Conf. Ucomms 2016*, 2016, doi: 10.1109/UComms.2016.7583420.