An Event-Based Stack For Data Transmission Through Underwater Multimodal Networks

Roberto Francescon, Filippo Campagnaro, Emanuele Coccolo,
Alberto Signori, Federico Guerra, Federico Favaro, Michele Zorzi

***Abstract*—The DESERT Underwater framework (http://desert-underwater.dei.unipd.it/), originally designed for simulating and testing underwater acoustic networks in sea trials, has recently been extended to support real payload data transmission through underwater multimodal networks. Specifically, the new version of the framework is now able to transmit data in real-time through the EvoLogics S2C low-rate and high-rate acoustic modems, the SmartPORT low-cost acoustic underwater modem prototype (AHOI) for IoT applications, as well as Ethernet, surface WiFi, and the BlueComm optical modem. The system can also be tested in lab by employing a simulated channel, and the EvoLogics S2C DMAC Emulator (DMACE).**

***Index Terms*—Underwater multimodal networks, emulation, test-bed, NS-Miracle, AHOI modem, EvoLogics.**

I. Introduction

Underwater networks allow a wide range of submarine applications, such as monitoring of oil spill, remote control for underwater vehicles and coastline protection. In order to verify whether the performance of underwater network protocols obtained through analysis and simulation can effectively be achieved in a real scenario, dedicated real field evaluations must be carried out, in order to deal with the actual capabilities and limitations of the real hardware. Moreover, the best evaluation possible should be performed by transmitting real data instead of synthetic data automatically generated by the network simulator itself, in order to verify whether a certain application can actually be supported with the designed network or not. To achieve this, employing a framework that can easily allow to test a protocol stack both in simulations and in sea trials is essential. Furthermore, the same framework can become a tool to send real data through the underwater network. In the last years, several tools [1]–[4] have been developed to support both simulations and sea trials. Among the others, the SUNSET framework [1], based on the MIRACLE extensions [5] of the network simulator ns2 [6], provides support for simulation, emulation and experimentation. Similarly, the AquaNet framework [2] provides the possibility to switch between simulations and real-life experiments in underwater acoustic networks. The Aqua-Sim [7] simulator, recently ported from ns2 to ns3, is willing to exploit the emulation capabilities of ns3 to be employed for experimentation as well. Also the UAN framework [8] is based on ns3, but it supports only network simulations. The third version of the UnetStack framework [3] provides, not only the capabilities to run a software defined modem in a laptop, but also a set of APIs to develop your real application and therefore send real data through acoustics. The UnetStack software defined modem mode exploits the SubNero [9] acoustic modems features, while its simulation mode models the performance of the SubNero modem with high reliability. The CMRE CCA Framework [4], envisions a modular and multimodal communications architecture, designed to enable the deployment of advanced autonomous underwater solutions making use of smart, adaptive and secure underwater networking strategies. Mainly developed to be a reliable and robust framework for in-field activities, it also provides the possibility to perform network simulator, interacting with the CMRE channel simulator. Another tool mainly developed for in-field activity is the EviNS framework [10], that can either work with S2C EvoLogics modems [11] or with the EvoLogics DMACE real-time emulator [12]. The latter employs a simulated underwater channel, specifically designed to map the EvoLogics modem performance.

The open-source DESERT Underwater [13] framework, available online [14], offers a complete set of underwater networking capabilities, with the support of modular stacks and cross-layer communications. Based on the MIRACLE extensions [5] of the network simulator ns2 [6], it grants the co-existence of multiple modules within each layer of the protocol stack. The development of different rate acoustic and non-acoustic communications, like optical or radio-frequency (RF) technology, requires a tool for simulating and testing general multi-modal networks. Addressing this requirement, the DESERT Underwater framework can support multiple and heterogeneous protocol stacks in each node. This feature enables the possibility to send packets through high frequency and low frequency acoustics as well as optical physical layers. The physical layers can be simulated [13], emulation maps of existing devices (e.g via a Lookup Table) or connected with real modems to perform actual transmissions.

In this work, we present the new design of the drivers used for in-field experiments with the DESERT Underwater framework developed at the University of Padova. First, in Section II, the new structure of the DESERT drivers is presented. This includes the general structure of the interface between the DESERT physical layer and the modems (Section II-A), as well as the specific implementation of the interface of

F. Campagnaro (Corresponding author, email: campagn1@dei.unipd.it), E. Coccolo, A. Signori, F. Guerra, F. Favaro and M. Zorzi are with the Department of Information Engineering, University of Padova, Italy. R. Francescon is with Wireless and More srl, Padova, Italy.

This work has been supported in part by the Italian Ministry of Education, University and Research (MIUR), and ERA-NET Cofound MarTERA (contract 728053). We would like to thank the smartPORT research group from TUHH for providing three acoustic modems and support.

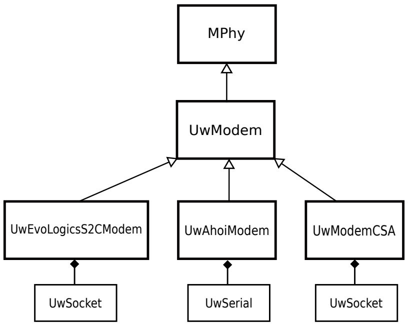

Fig. 1. Structure of the inheritance relationships of `UwModem` class.

two different acoustic modems , i.e., the S2C EvoLogics [11] (Section II-B) and the AHOI acoustic modems [15] (Section II-C)), and of a end-to-end socket-based modem (Section II-D), that can be used either to mock actual transmissions in lab or to transmit data through Ethernet-based modems, such as the Sonardyne BlueComm optical modems [16]. In Section III, we present different use cases and evaluate the drivers performance with an in-lab test campaign.

In Section IV we draw conclusions and illustrate ongoing work.

II. Modem Drivers of DESERT Underwater

The drivers provided by DESERT Underwater are responsible for managing real modems, allowing to transmit and receive real data, as well as configure the devices according to the technical manual provided by the manufacturer. Each modem driver is a ns2 module representing a physical layer in the ns2 architecture.

A. Drivers General Structure

The general structure, presented in Fig. 1, includes the `UwModem` class, whose main task is to make available the basic functionalities common to each modem, such as transmission, reception and configuration. This is achieved through the use of a connector (`UwConnector`): in this context, connector refers to the physical interface connecting the communication hardware to the computer running DESERT Underwater, such as an Ethernet connection (either TCP or UDP) or a serial connection; to this aim, inheriting from `UwConnector`, the classes `UwSocket` and `UwSerial` implement the socket and the serial interfaces, respectively.

In addition to the main thread executing the event scheduler, each modem starts a transmission and a reception thread: these threads are created by the derived class. In the main thread the packets received from the upper layer are stored in a thread-safe transmission queue before being sent through the modem with the transmission thread. Similarly, packets received from the modem by the reception thread are stored in a thread-safe reception queue, periodically checked by the main thread which forwards the received packets to the upper layers. The checking period TO_{RX} can be set in the simulation script. The lower TO_{RX} the quicker the synchronization between the event scheduler and the real packet reception, and the higher the CPU use. For instance, in a laptop with an Intel core i7-8700 with 12 cores, $TO_{RX} = 10$ ms provides an average synchronization delay of 5 ms, and a CPU use (single core) of the 4%.

To correctly parse notifications from the modem and format commands, if any, a dedicated class, called *interpreter*, is implemented in the derived class.

This structure allows for a simple integration of additional drivers for other devices: a new driver for a modem connected through the serial interface would be deriving from `UwModem`, including the connector `UwSerial` and a custom state machine, used to send commands to the device, according to its specifications. Each event involving the modems, such as a packet to be transmitted or a notification from the device, is converted into an ns2 event and queued, along with all other simulator events, allowing to exploit the ns2 events management.

B. S2C EvoLogics Drivers

EvoLogics GmbH [11] manufactures commercial off-the-shelf devices for a wide range of environments and with various performance. These modems employ a technology called S2C (Sweep-Spread Carrier) which effectively tries to compensate multi-path. These devices can be accessed through an Ethernet interface, thanks to a TCP server listening for incoming connections. The S2C devices are used, by the DESERT framework, in *Command Mode*, which allows to transmit and receive data and configure the modem easily, using only a TCP connection.

The EvoLogics devices have two types of packet transmission: *Instant Messages* (IM) and *Burst* transmissions [17]. While IMs must be issued one at at time to not override the ongoing transmission and they exploit the lowest possible bitrate (976 bit/s for all devices with the exception of the S2C M HS, which provides a minimum bitrate of 1952 bit/s), the burst transmissions exploit the full capability of the modem by determining the highest possible bitrate given the channel conditions estimated by the device itself. It works by storing all the issued transmissions in the modem's internal buffer, and then sending them as fast as possible. In our drivers, we implemented a state machine that strictly controls when packets are sent: a packet can be sent to the modem only if we received the confirmation of the previous transmission attempt. This behavior implements in DESERT a stop and wait retransmission policy, that is beneficial to the correct transmission of data and the control of the data link layer, but rapidly deteriorates the performance. Indeed, the design choice of DESERT's drivers focus more on the networking capabilities than on relying on specific abilities of the surrounding hardware to achieve the best performance of a single link.

The state machine for the S2C drivers includes 2 types of states: S_{TX} and S, respectively. S_{TX} represents the possible states for the transmission flow, specifically, `TX_IDLE` and `TX_PENDING`, and S represents the general state of the whole device, i.e., `AVAILABLE`, `BUSY`, `RESET`, `QUIT`, `DEAF`, `NOISE` and `LISTEN`. While the first two states indicate whether the modem is busy for a pending operation or not, the other states are used to apply specific features of the S2C modems. The main task of the state machine is to determine when a new configuration command or a new packet can be sent to the device, as the S2C modems can execute one command at a time, and the next command cannot be issued until the first is not confirmed. They make use of the UwSocket interface.

C. *AHOI Drivers*

The AHOI modems are prototypical cheap acoustic underwater modems developed by the smartPORT group at TUHH (Hamburg) [18] and they are accessed via serial connection. There is a practical interface, written in Python, that allows to use the modems as a *chat-like* application: this and other software, to manage the devices, can be found in the TUHH repositories [19]. The AHOI modem is a cost-effective, high-frequency modem having a maximum bitrate of 260 bit/s (using Hamming encoding) and a maximum range of 150 m. It is intended mainly to be integrated in micro-AUVs and underwater sensor nodes. The modem operates, by default, in the 50-75 kHz band.

Also for the AHOI drivers, we keep track of the general modem state S, that can be `AVAILABLE`, `TRANSMITTING` or `CONFIGURING`, and the transmission flow state S_{TX}, that can assume the values `TX_IDLE` and `TX_WAITING`. They specify if the modem is ready for new commands, is already transmitting or is being configured and they assure a correct transmission of all the packets. Also in this case one command can be issued at a time. They make use of the UwSerial interface.

D. *Client-Server Architecture (CSA) Drivers*

These drivers are an end-to-end socket connection from a host to another remote host. They make use of the UwSocket interface, and a simple two-state machine (AVAILABLE and TRANSMITTING). This driver implementation does not introduce further mechanisms regarding packet retransmissions, but only transmits a single packet as soon as the modem is in the AVAILABLE state. These drivers can be used both to connect two hosts using either TCP or UDP through Ethernet or WiFi links, as well as with modems that make available an end-to-end Ethernet link, such as the Sonardyne BlueComm modems [16].

III. Use Case Examples and Performance Evaluation

In this section we investigate how the introduction of a state machine to control the use of the modems and ensure the synchronization between actual transmissions and the ns2 event scheduler affects the communication performance.

The application layer used to perform the following tests is `uwApplication`: this module enables to transmit through DESERT every kind stream through TCP or UPD sockets: from a chat-like application to the transmission of large files and images. This module is particularly useful as it allows to transmit real data instead of mocked bytes.

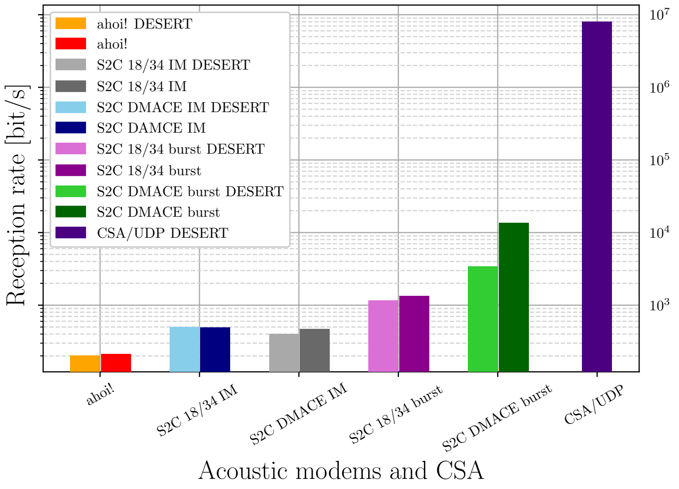

Fig. 2. Averaged bitrate achieved while transmitting an image with various acoustic modems and the UDP end-to-end link.

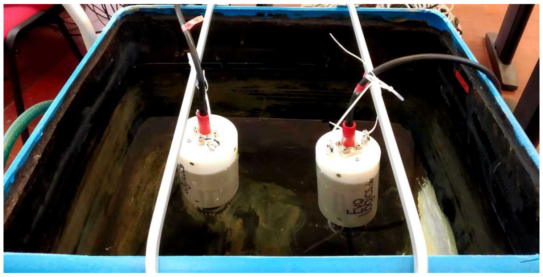

Fig. 3. Setup of the test, for the S2C devices, in the insulated tank

A. *End-to-End Image Transmission*

To evaluate the performance of the DESERT drivers, we set up a transmission of a jpeg image of 2345 bytes from one PC to another by employing different modems, using the devices both through DESERT and directly. In both cases we measured the datarate through the physical layer of the receiver stack R_{RX}.

We tested the S2C modem drivers by employing both the DMACE emulator [12] of the S2C 48/78 modems and two real S2C R 18/34 modems immersed in an insulated tank [20] (Fig. 3). The same tank has been used to test the performance of the drivers developed for the AHOI modems (Fig 4), where the 2 transducers were deployed few centimeters apart.

Fig. 2 shows that the performance loss caused by the use of the DESERT stack in the data transmission with AHOI and S2C modems in IM mode can be assumed to be negligible, specially if we consider the benefits of employing a protocol stack able to provide multimodal and multi-hop networking capabilities [13].

Specifically, with the AHOI modems and the DESERT Underwater protocol stack, R_{RX} = 205 bit/s, while using the transmission tools provided by TUHH [19], R_{RX} = 212 bit/s. The packet size set when employing the AHOI modems was 32 bytes. With the S2C 18/34 IMs through DESERT, instead, we were able to reach R_{RX} = 500 bit/s, the same performance we obtained when transmitting without DESERT. In this case

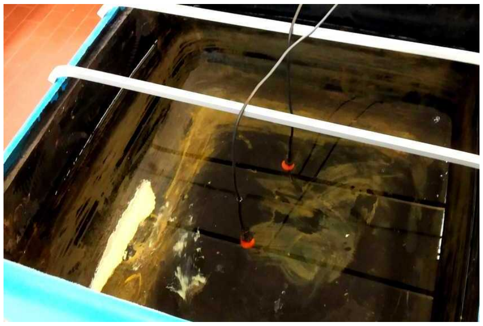

Fig. 4. Setup of the test, for the AHOI modems, in the insulated tank

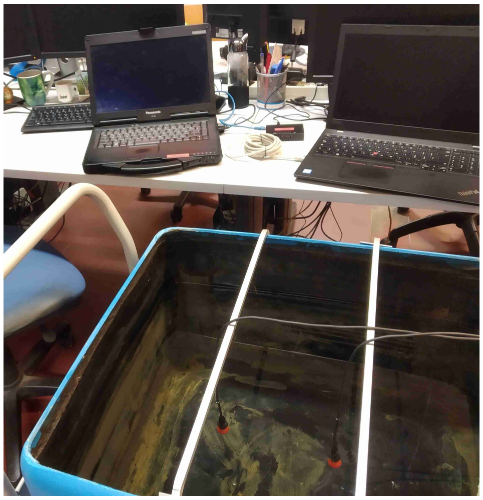

Fig. 5. Setup of the test, for the multimodal network evaluation

the packet size was 64 bytes and the modems were deployed few centimeters apart. Instead, by using the DMACE emulator, with the packet size still equal to 64 bytes, R_{RX} = 400 bit/s with DESERT and 469 bit/s without DESERT. This gap is due to the presence of a VPN to access the DMACE emulator, with a ping delay of 32 ms: the performance of the test without DESERT were not affected by this delay because we were just transmitting consecutive messages through the modem without checking its state. In this case the transmission period was manually tuned to maximize the reception rate.

Using the S2C 18/34 devices for burst transmission, we measured R_{RX} = 1170 bit/s through DESERT and 1340 bit/s when using the modem without DESERT. In this setup the modem bitrate was 3246 bit/s and we transmitted packets of 512 bytes, while, transmitting burst data through the DMACE, with the nodes deployed 5 m apart and a packet size of 824 bytes, we reached R_{RX} = 3.385 kbit/s through DESERT and R_{RX} = 13.504 kbit/s when using the modems directly. The bitrate of the emulated modem was 31.2 kbit/s. In these two cases the performance difference is not negligible anymore, because while the modems themselves buffer the out-coming packets and optimally transmit them, with the DESERT stack we wait for the confirmation of each packet to be correctly received to have a software control of the MAC layer, as explained in Section II-B.

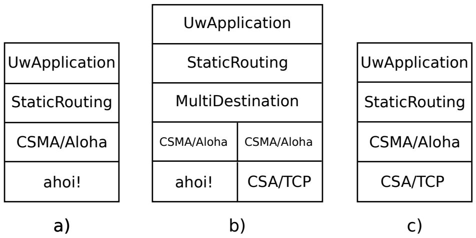

Fig. 6. Stacks of the 3 types of nodes involved in the multimodal data muling test: a) UW nodes, b) buoy node and c) onshore node

To prove that DESERT can also support the datarate transmission of a Bluecomm 200 optical modem (which actual payload datarate varies from 0.25 Mbit/s to 8 Mbit/s), in Fig. 2 we also evaluated the maximum datarate achievable with the CSA drivers when transmitting UDP traffic (indeed, 8 Mbit/s).

B. *Multimodal two-hops underwater sensor network*

We implemented the two-hops network that is illustrated in Fig. 6 in order to test a multimodal network composed by three nodes, where node a) and node b) are connected through the AHOI modem, and node b) and node c) through a CSA/TCP link. Given that the CSA/TCP link is way faster than the AHOI acoustic link, we expect the bitrate to adapt to the latter, around 200 bit/s. The packet transmission from one stack to another, in node b), is performed by the `multi-destination` module [21] that operates between the MAC layer and the Network layer. This setup represents a situation in which an underwater data-collecting node wants to send data to a nearby buoy which is connected via a radio link to the onshore station. Each of the three PCs illustrated in Fig. 5 is running a separate DESERT instance, specifically, the workstation PC is running node a), and the two laptops are running node b) and node c), respectively. The test performed confirmed that the end-to-end bitrate reaches the maximum exploitable bitrate of the AHOI modems, i.e. 205 bit/s.

IV. Conclusions and Future Work

In this paper, we presented the new drivers structure of DESERT Underwater v3. The multimodal capabilities of DESERT allows now to perform a real field deployment of an heterogeneous hybrid network, composed by EvoLogics acoustic modems, the low-cost AHOI acoustic modems, Bluecomm optical modems and surface WiFi links. In this paper we provide an in-lab evaluation of the responsiveness of the framework respect to the direct use of the modems. We quantify the performance loss with each possible hardware configuration, proving that the system can be employed for real data transmission in multihop multimodal underwater networks, with the price of a almost negligible performance loss due to the DESERT processing time. Future work will focus on the evaluation of a complete mutlimodal network composed by both S2C and AHOI modems, as well as wireless surface links in real field tests to perform data gathering from underwater sensors [22] in the context of the RoBoVaaS project [23] and the ability to optimally exploit modem-specific features such the S2C burst data transmission.

References

[1] C. Petrioli, R. Petroccia, and D. Spaccini, "SUNSET version 2.0: Enhanced Framework for Simulation, Emulation and Real-life Testing of Underwater Wireless Sensor Networks," in *Proc. ACM WUWNet*, Kaohsiung, Taiwan, Nov. 2013.

[2] Y. Zhu, S. Le, L. Pu, X. Lu, Z. Peng, and J. Cui, "Towards experimental evaluation of software-defined underwater networked systems," in *Proc. MTS/IEEE Oceans*, San Diego, CA, US, Sep. 2013.

[3] The NUS ARL and SubNero, "UNET–The Underwater NETworks project," Last time accessed: Jul. 2020. [Online]. Available: http://www.unetstack.net

[4] R. Petroccia, G. Zappa, T. Furfaro, J. Alves, and L. D'Amaro, "Development of a software-defined and cognitive communications architecture at CMRE," in *Proc. MTS/IEEE Oceans*, Charleston, SC, US, Oct. 2018.

[5] N. Baldo, M. Miozzo, F. Guerra, M. Rossi, and M. Zorzi, "Miracle: the multi-interface cross-layer extension of ns2," *EURASIP J. of Wireless Commun. and Networking*, 2010. [Online]. Available: http://www.hindawi.com/journals/wcn/aip.761792.html

[6] "The Network Simulator - *ns-2*," Last time accessed: Jul. 2020. [Online]. Available: https://www.isi.edu/nsnam/ns/

[7] R. Martin and R. Z. P. Sanguthevar, "Aqua-Sim Next Generation: A NS-3 Based Simulator for Underwater Sensor Networks," in *Proc. ACM WUWNet*, Halifax, Canada, Nov. 2017.

[8] "ns3 UAN framework," Last time accessed: Jul. 2020. [Online]. Available: https://www.nsnam.org/docs/models/html/uan.html

[9] "Subnero M25M Series Modems," last time accessed: Jul. 2020. [Online]. Available: https://subnero.com/products/modem.html

[10] "EviNS: Framework for Development of Underwater Acoustic Sensor Networks and Positioning Systems," Last time accessed: Jul. 2020. [Online]. Available: https://github.com/okebkal/evins

[11] "EvoLogics Underwater Acoustic Modems," Last time accessed: Jul. 2020. [Online]. Available: https://evologics.de/acoustic-modems

[12] "S2C D-MAC Emulator," Last time accessed: Jul. 2020. [Online]. Available: https://evologics.de/emulator

[13] F. Campagnaro, R. Francescon, F. Guerra, F. Favaro, P. Casari, R. Diamant, and M. Zorzi, "The DESERT underwater framework v2: Improved capabilities and extension tools," in *Proc. Ucomms*, Lerici, Italy, Sep. 2016.

[14] "The DESERT Underwater libraries," Last time accessed: Jul. 2020. [Online]. Available: http://desert-underwater.dei.unipd.it/

[15] C. Renner and A. J. Golkowski, "Acoustic modem for micro auvs: Design and practical evaluation," in *Proc. ACM WUWNet, Shanghai, China*, Oct. 2016.

[16] "BlueComm Underwater Optical Communication," Last time accessed: Jul. 2020. [Online]. Available: https://www.sonardyne.com/product/bluecomm-underwater-optical-communication-system/

[17] O. Kebkal, M. Komar, K. Kebkal, and R. Bannasch, "D-MAC: Media access control architecture for underwater acoustic sensor networks," in *Proc. OES/IEEE Oceans*, Santander, Spain, Jun. 2011.

[18] F. Steinmetz and C. Renner, "Resilience against shipping noise and interference in low-power acoustic underwater communication," in *Proc. MTS/IEEE Oceans*, Seattle, WA, US, Oct. 2019.

[19] TUHH SmartPORT, "AHOI acoustic modem repository," Last time accessed: Jul. 2020. [Online]. Available: https://collaborating.tuhh.de/smartport/public/ahoi

[20] F. Meneghello, F. Campagnaro, R. Diamant, P. Casari, and M. Zorzi, "Design and evaluation of a low-cost acoustic chamber for underwater networking experiments," in *Proceedings of the 11th ACM International Conference on Underwater Networks & Systems*, ser. WUWNet '16. New York, NY, USA: Association for Computing Machinery, October 2016.

[21] W. Rizzo, A. Signori, F. Campagnaro, and M. Zorzi, "AUVs telemetry range extension through a multimodal underwater acoustic network," in *Proc. MTS/IEEE Oceans*, Charleston, SC, US, Oct. 2018.

[22] A. Signori, F. Campagnaro, F. Steinmetz, B.-C. Renner, and M. Zorzi, "Data gathering from a multimodal dense underwater acoustic sensor network deployed in shallow fresh water scenarios," *MDPI Journal of Sensor and Actuator Networks*, vol. 8, no. 4, Nov. 2019.

[23] "Robovaas, robotic vessels as-a-service," Last time accessed: Jul. 2020. [Online]. Available: https://www.martera.eu/projects/robovaas