HORIZON 2020

PROJECT DELIVERABLE REPORT

PALAEMON

Introducing advanced ICT
and Mass Evacuation Vessel design
to ship evacuation and rescue systems

D7.3_Deployment of VDES (v2)

MG-2-2-2018
Marine Accident Response

“This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 814962”

Information

Grant Agreement Number	814962	Acronym	PALAEMON	
Full Title	A holistic passenger ship evacuation and rescue ecosystem			
Topic	MG-2-2-2018: Marine Accident Response			
Funding scheme	RIA - Research and Innovation action			
Start Date	1st JUNE 2019	**Duration**	42 months	
Project URL	www.palaemonproject.eu			
EU Project Officer	Georgios CHARALAMPOUS			
Project Coordinator	AIRBUS DEFENCE AND SPACE SAS			
Deliverable	VDES deployment			
Work Package	WP7 – PALAEMON Integrated System and Technology Validation Trials			
Date of Delivery	**Contractual**	M32 + 6	**Actual**	M38
Nature	R - Report	**Dissemination Level**	PU-PUBLIC	
Lead Beneficiary	NTUA			
Responsible Author	WISER/Thales	Email		
		Phone		
Reviewer(s):				
Keywords				

Revision History

Version	Date	Responsible	Description/Remarks/Reason for changes
0.1	2022/05/30	Merlini	Template
0.2	2022/06/06	Merlini	Inclusion of existing partners' contributions
0.3	2022/06/08	Merlini	Internal Review
0.4	2022/07/15	Piantini	New contributions
05	2022/09/18	Frasconi	Version for final review
0.6	2022/11/29	Morelli	Reviewed version
1.0	2022/11/29	Guerri	Submitted version

Contents

List of Figures

List of Tables

Abbreviations

AIS	Automatic Identification System
ASM	Application Specific Message
AWGN	Additive White Gaussian Noise
BB	Base Band
BER	Bit Error Rate
CIR	Channel Impulse Response
COG	Course Over Ground
COTS	Commercial Off-The Shelf
DFB	Data Fusion Bus
DPA	Designated Person Ashore
DRF	Django Rest Framework
DSS	Decision Support System
EIRP	Effective Isotropic Radiated Power
FEC	Forward Error Correction
GMDSS	Global Maritime Distress Safety System
HFACS-Coll	Human Factors Analysis and Classification System for Collision Accidents
HFACS-Ground	Human Factors Analysis and Classification System for Grounding Accidents
HFACS-MA	Human Factors Analysis and Classification System for Maritime Accidents
HFACS-MSS	Human Factors Analysis and Classification System for Machinery Space Fire Accidents
IBS	Integrated Bridge System
IOC	Intergovernmental Oceanographic Commission
IR	Information Retrieval
JTSB	Japan Transportation Safety Board
LLR	Log-Likelihood Ratio
LMS	Least Mean Square
LSB	Least Significant Bit
MAIB	Marine Accident Investigation Branch
MCS	Modulation Coding Scheme
MF	Medium Frequency

MMSI	Maritime Mobile Service Identity
MQTT	MQ Telemetry Transport
MSB	Most Significant Bit
MSI	Maritime Safety Information
NAVTEX	Navigational Telex
NOAA	National Oceanic and Atmospheric
NTSB	National Transportation Safety Board
PAPR	Peak to Average Power Ratio
PER	Packet Error Rate
PIMM	PALAEMON Incident Management module
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
REST	Representational State Transfer
RF	Radio-Frequency
RLS	Recursive Least Square
RSC	Recursive Systematic Convolutional
SDR	Software Defined Radio
SoC	System on Chip
SOG	Speed Over Ground
SOLAS	Safety of Life at Sea
SRRC	Square Root Raised Cosine
SWH	Significant Wave Height
VDES	VHF Data Exchange System
VHF	Very High Frequency
VTS	Vessel Traffic Service
WFT	Weather Forecast Tool

1 Introduction

This deliverable reports the deployment of VDES transceiver prototypes, describing any stage of the development process.

Specifically, in section 2 we report the VDES standard analysis and a list of requirements, affecting the VDES transceiver.

Section 3 presents the propagation channel model, which has been identified for simulation purposes. The proposed channel model is derived from a channel sounding campaign, carried out by General Lighthouse Authorities, UK.

Section 4 presents the architecture of the end-to-end simulator, which is used for performance assessment of the receiver architecture and of the related synchronization and demodulation algorithms.

Section 5 reports the description of the multi-thread framework, which is used to implement the SDR-based VDES waveform, running over Intel x64 architecture.

Section 6 presents the COTS hardware platform, selected for the VDES transceiver deployment. Specifically, this platform is composed by two electronics board, i.e. an RF tuner for VHF bands and an embedded processor based on Xilinx Zynq chipset.

Section 7 introduces the concepts of cross-compiling and porting the VDES software waveform over the COTS hardware platform.

Section 0 presents the interface description between the VDES radio and the other component of PALEMON platform.

2 VDES requirements

This section reports the requirements, which refer to the whole VDES. Requirements in this section are extrapolated from [1].

2.1 VDES Global requirements

REQ-GLO-001: The VDES transceiver shall support the following services: Automatic Identification System; Application-Specific Messages; VDE Terrestrial; VDES sharing options.

The priority and timing of transmissions shall be in accordance with the following service priorities:

- Priority 1 (Highest): AIS transmissions on AIS channels
- Priority 2: specified and approved ASM transmissions on ASM channels
- Priority 3: all other data exchange on VDE channels

REQ-GLO-002: The VDES transceiver shall operate on the following frequencies, when communicating between terrestrial stations (ships or shore stations):

- AIS 1 (channel 2087) and AIS 2 (channel 2088) are AIS channels, in accordance with Recommendation ITU-R M.1371
- ASM 1 (channel 2027) and ASM 2 (channel 2028) are the channels used for application specific messages (ASM)
- VDE1-A lower legs (channels 1024, 1084, 1025, 1085) are ship-to-shore VDE
- VDE1-B upper legs (channels 2024, 2084, 2025, 2085) are shore-to-ship and ship-to-ship VDE.

REQ-GLO-003: The VDES transceiver shall operate on the following frequencies, when communicating between satellite and terrestrial stations:

- AIS 1 (channel 2087) and AIS 2 (channel 2088) are terrestrial AIS channels that are also used as uplinks for receiving AIS messages by satellite
- Long Range AIS using channel 75 and channel 76 are specified channels to be used as uplinks for receiving AIS messages by satellite. SAT Up1 (channel 2027) and SAT Up 2 (channel 2028) are used for receiving ASM by satellite
- SAT Up3 (channels 1024, 1084, 1025, 1085, 1026 and 1086) are used for ship-to-satellite VDE uplinks
- SAT Downlink (channels 2024, 2084, 2025, 2085, 2026 and 2086) are used for satellite-to-ship VDE downlinks.

VDES frequency channels are shown in Figure 1.

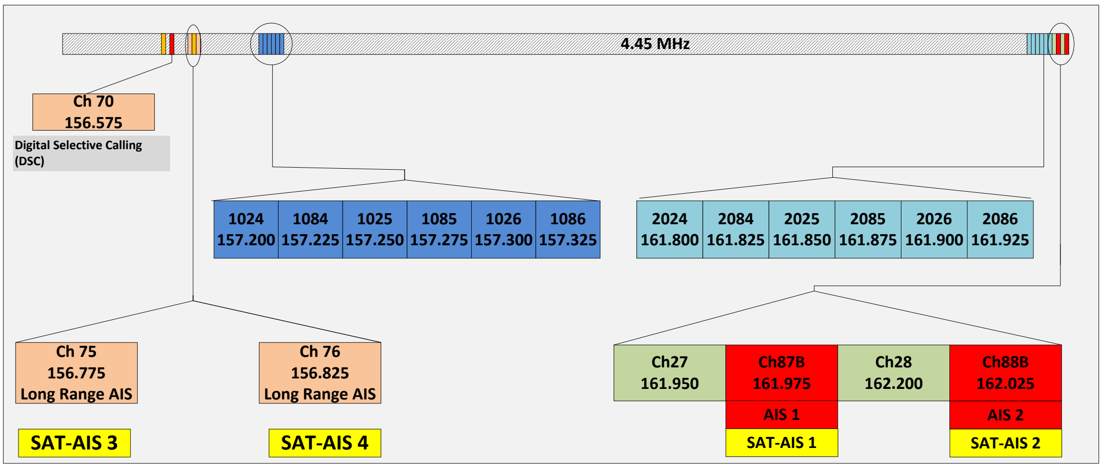


Figure 1 – VDES frequency allocation

2.2 VDES common technical requirements

This section reports the technical requirements, which are common for VDE and ASM channels. These requirements have been extrapolated from Annex 1 of [1].

REQ-COM-001: The timing accuracy of the transmit signal shall be better than 5 ppm.

REQ-COM-002: The timing jitter shall be better than 5% of the symbol interval (peak value).

Example: let us consider the π/4-QPSK, with 25 kHz channel bandwidth, whose symbol rate is 19.2 kbaud. The symbol interval is around 52 µs. The tolerated jitter is ±2.6 µs.

REQ-COM-003: The slot transmission accuracy shall be better than 100 µs peak relative to UTC time reference for ship stations.

REQ-COM-004: The VDES transceiver shall be able to directly or indirectly synchronize to the UTC. A station, which is unable to get direct access to UTC, but has access to the AIS system timing, should get its synchronization from the AIS system.

REQ-COM-005: The VDES transceiver shall meet the requirements reported in Table 1 – Minimum ship EIRP vs elevation angle, in terms of ship EIRP vs elevation angle.

Ship elevation angle	Ship antenna gain	Minimum ship e.i.r.p. with 6 W transmitter
degrees	dBi	dBW
0	3	10.8
10	3	10.8
20	2.5	10.3
30	1	8.8
40	0	7.8
50	−1.5	6.3
60	−3	4.8
70	−4	3.8
80	−10	−2.2
90	−20	−12.2

Table 1 – Minimum ship EIRP vs elevation angle

2.3 VDE only requirements

REQ-VDE-001: The VDES transceiver shall observe the following table of transmitter requirements on VDE channels.

Transmitter parameters	Requirements	Condition
Frequency error	1.5 ppm	Normal
Frequency error	3 ppm	Extreme
Maximum adjacent power levels for 25 kHz channel	Δfc < ±12.5 kHz: 0 dBc ±12.5 kHz < Δfc < ±25 kHz: below the straight line between −25 dBc at ±12.5 kHz and −70 dBc at ±25 kHz ±25 kHz < Δfc < ±62.5 kHz: −70 dBc	
Maximum adjacent power levels for 50 kHz channel	Δfc < ±25 kHz: 0 dBc ±25 kHz < Δfc < ±37.5 kHz: below the straight line between −25 dBc at ±25 kHz and −70 dBc at ±37.5 kHz ±37.5 kHz < Δfc < ±125 kHz: −70 dBc	
Maximum adjacent power levels for 100 kHz channel	Δfc < ±50 kHz: 0 dBc ±50 kHz < Δfc < ±62.5 kHz: below the straight line between −25 dBc at ±50 kHz and −70 dBc at ±62.5 kHz ±62.5 kHz < Δfc < ±250 kHz: −70 dBc	
Spurious emissions	−36 dBm	9 kHz to 1 GHz
	−30 dBm	1 GHz to 4 GHz

Table 2 – Transmitter requirements

Figure 2 shows the VDE transmission mask for each channel bandwidths.

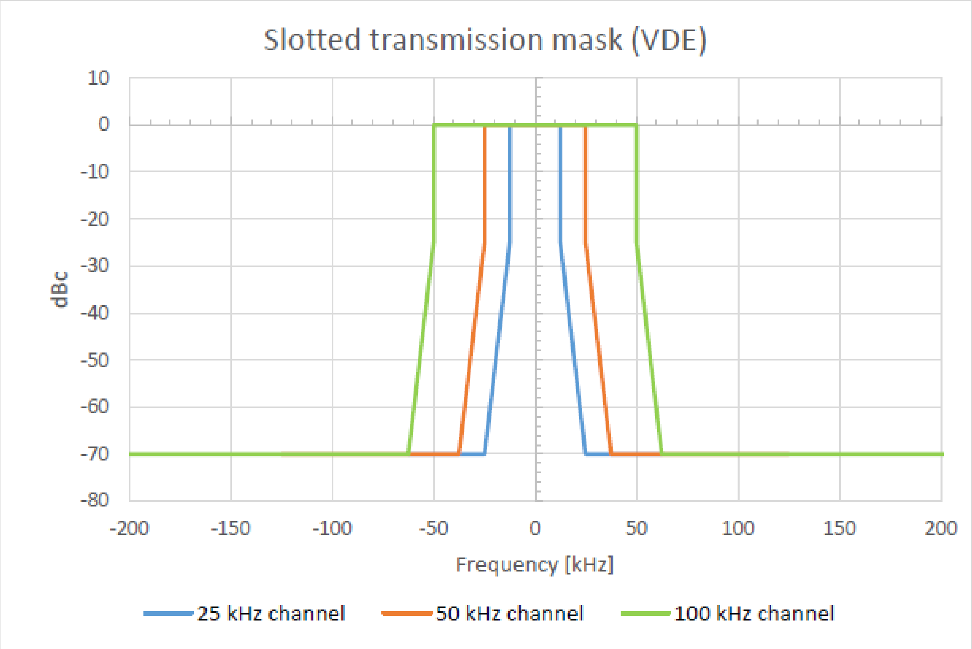


Figure 2 – VDE transmission mask

REQ-VDE-002: The VDES transceiver shall meet the following requirements in terms of sensitivity and Carrier-to-Interference Ratio (CIR), when receiving VDE channels.

Modulation Coding Scheme	25 kHz		50 kHz		100 kHz	
	Sensitivity (dBm)	CIR (dB)	Sensitivity (dBm)	CIR (dB)	Sensitivity (dBm)	CIR (dB)
MCS-1*	–110	8	–107	8	–104	8
MCS-3*	–104	14	–101	14	–98	14
MCS-5*	–102	16	–99	16	–96	16

Table 3 – Sensitivity and interference requirements for VDE channels.

Table 3 assumes a BER of 10^{-6}.

3 Marine propagation channel modelling

This section reports a summary of the channel sounding campaign conducted by the General Lighthouse Authorities (GLA) to examine radio propagation conditions using VHF spectrum for maritime applications. The results of this channel measurements trial are reported in [2]. The analysis of the channel sounding campaign enables to better understand the channel model to include in the end-to-end simulator and the countermeasures to select in the receiver design.

The experiments spanned all channels intended for use in ship-to-shore and shore-to-ship VDES communications. Five scenarios were examined, spanning four of the six IMO Maritime Service Portfolio area categories. The summary of the channel sounding campaign is reported in Table 4.

Scenario	LOS Conditions	Sea State	Rx Power dBm	Additional Path Loss dB	Contributing Factors
Harwich/Felixstowe Harbour	LOS	Slight	-43 to -58	5 to 16	Multipath and shadowing (due to vessel clutter).
Harwich/Felixstowe Harbour Approach	LOS	Moderate to rough	-77	23	Antenna polarisation mismatch induced by rough sea state. Multipath and shadowing.
Ipswich and River Orwell	NLOS	Calm to smooth	-95	39	Multipath and heavy shadowing.
Sunk Deep Water Anchorage	LOS	Moderate to rough	-98	35	Antenna polarisation mismatch induced by rough sea state. Multipath, shadowing (due to vessel clutter).
Gunfleet Sands Wind Farm (North-1)	Near-LOS	Moderate	-84	27	Multipath and shadowing.
Gunfleet Sands Wind Farm (North-2, South)	NLOS	Slight	-95	35	Multipath and heavy shadowing.

Table 4 – Summary of GLA analysed propagation channel

Figure 3 and Figure 4 show the Channel Impulse Response (CIR) for two scenarios, used as references. For both figures the graph on the left represents the amplitude of each channel path, whereas the graph on the right represents the evolution the channel coefficients

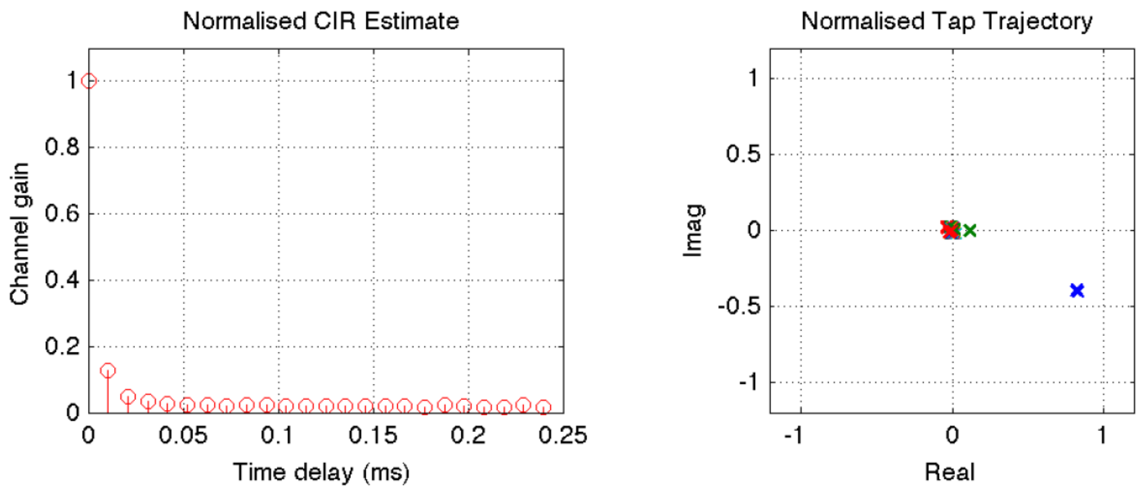


Figure 3 – Measured channel impulse response for VDE1 in the scenario 1.

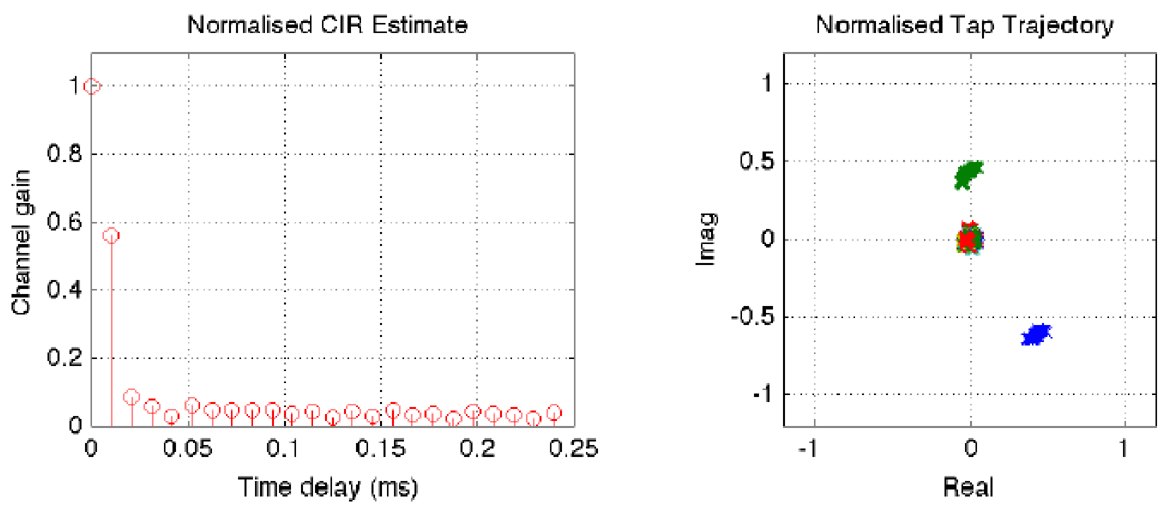


Figure 4 – Measured channel impulse response for VDE1 in the scenario 2 (strong multipath).

Given the CIRs above, it is clear that severe multipath conditions should be envisaged for the VDE transceiver. For this reason, we decided to include an RLS equalizer in the receiver chain. This equalizer is presented in Section 4.2.

4 High level architecture of the VDES transceiver

4.1 Transmitter section architecture

This section presents the VDE transmitter architecture, implemented in the simulator, used for performance assessment. This architecture is depicted in Figure 5. The first block is the Bit generator, which generates pseudo-random bits, which simulates the data payload to insert in each VDE burst. The pseudo-random bits are then conveyed to the turbo encoder, specified in [1].

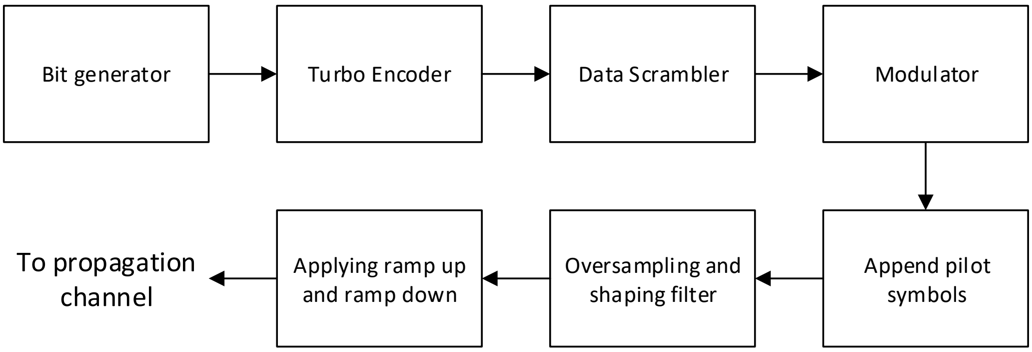


Figure 5 – Digital section of the VDE transmitter architecture

The Turbo encoder, shown in Figure 6, structure follows the specification in the ETSI EN 302 583 standard. The encoder consists of two recursive systematic convolutional (RSC) encoders concatenated in parallel. Each encoder produces 3 output bits per input bit. The first RSC encoder produces the bits X, Y0 and Y1, while the second encoder produces the bits X', Y'0 and Y'1. The π block in Figure 6 represents the interleaving function. The first encoder gets as input a word u of k bits. The second encoder input is denoted by u' and it is a permuted version of the vector u. The input u is the data (including padding and CRC), with MSB of each byte first.

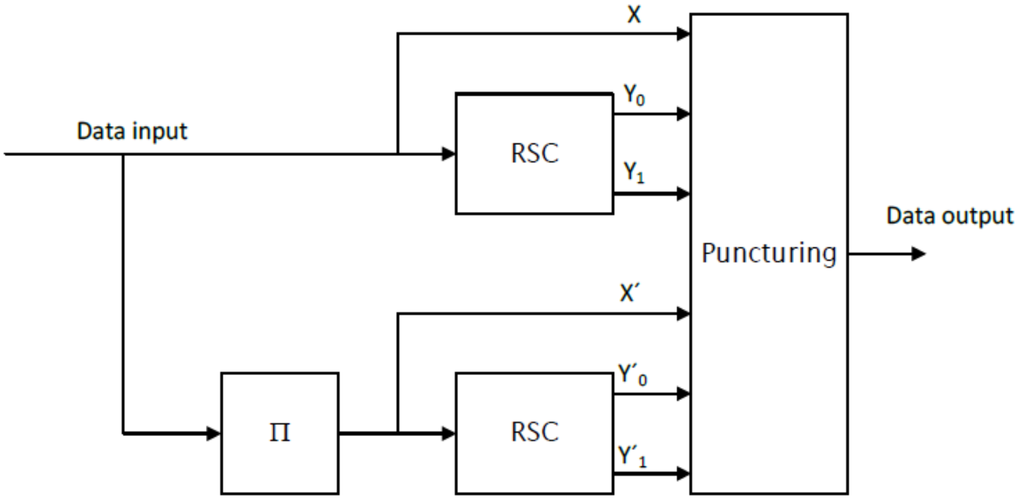


Figure 6 – Turbo encoder structure

The structure of the RSC encoders, the interleaver patterns and the puncturing schemas are presented in [1].

The output of the turbo encoder is then conveyed into the bit scrambles, whose structure is shown hereafter. The bit scrambler uses the polynom:

$$F(x) = 1 + x^{-14} + x^{-15} \quad (1)$$

and the initialization sequence as indicated in the top of. For each transmitted packet, the bit scrambler is re-initialized. The MSB shall be the first output bit.

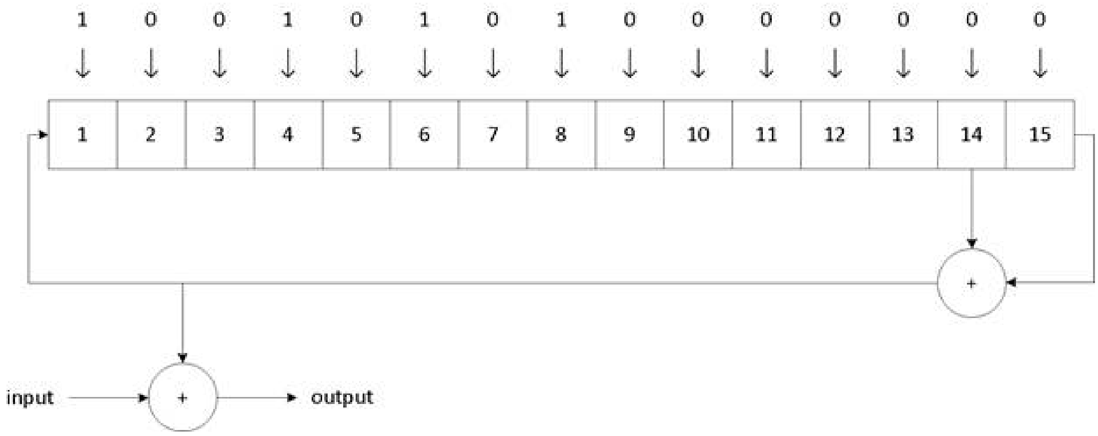


Figure 7 – Bit scrambler structure

The output of bit scrambler is then fed into the modulator. Three different modulation schemas are allowed over the VDE channels, i.e. π/4-QPSK, 8PSK and 16QAM. The bit mapping for each modulation scheme foreseen in the VDES standard are shown in Figure 8, Figure 9 and Figure 10.

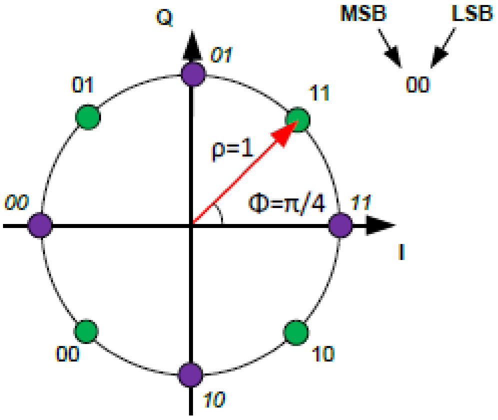


Figure 8 – Bit mapping for π/4-QPSK

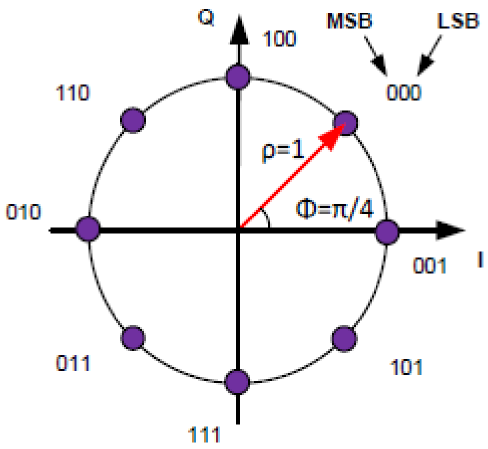


Figure 9 – Bit mapping for 8PSK

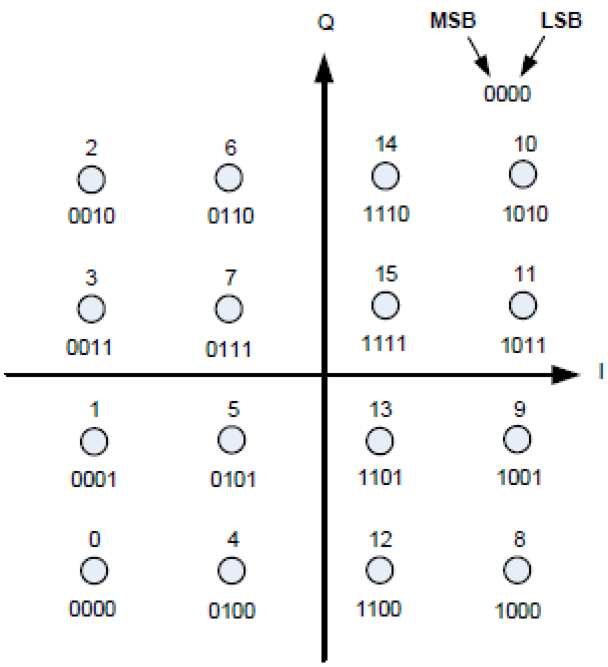


Figure 10 – Bit mapping for 16QAM

The output of the modulator is conveyed into a block, which adds VDE specific symbols, i.e. pilot symbols, link configuration symbols and ramp up and ramp down symbols, so creating the VDE burst of symbols. These parameters are better specified for each VDE transmission mode in Table 5.

PL format #	TER-MCS-1.25	TER-MCS-3.25	TER-MCS-5.25	TER-MCS-1.50	TER-MCS-3.50	TER-MCS-5.50	TER-MCS-1.100	TER-MCS-3.100	TER-MCS-5.100	
Link Config ID	11	12*	13*	14*	15*	16*	17	18*	19	
Channel BW	25			50			100			kHz
Roll off filtering	0,3									
Signal BW	25,0			49,9			99,8			kHz
Symbol rate	19,2			38,4			76,8			ksps
PAPR (example)	3.82	4.4	6.7	3.82	4.4	6.7	3.82	4.4	6.7	dB
Minimum high output average power	12.5	11	6.5	12.5	11	6.5	12.5	11	6.5	Watt
Burst size	1									slot
Guard time	0,83									ms
Burst duration	25,8									ms
Symbols/burst	496			992			1984			symbols
Ramp-up/down	8/8			16/16			32/32			symbols
Ramp-up/down	0.41/0.41									ms
Syncword size	27									symbols
Syncword modulation	PI/4 QPSK (00/11 only)									
Link Config ID size	16 (32,6 block code)									symbols
Link Config ID modulation	PI/4 QPSK									
Net symbols/burst	437			917			1877			symbols
Channel bits	874	1311	1748	1834	2751	3668	3754	5631	7508	bits
Padding + FEC tail***	10+0	3+12	8+12	30+12	51+12	72+12	0+10	243+12	8+12	bits
FEC decoder input symbols	432			896			1872	1792	1872	symbols
FEC decoder input bits	864	1296	1728	1792	2688	3584	3744	5376	7488	bits
FEC output bits	432	972	1296	896	2016	2688	1872	4032	5616	bits
FEC output bytes	54	122	162	112	252	336	234	504	702	bytes
Modulation	PI/4 QPSK	8PSK	16 QAM	PI/4 QPSK	8PSK	16 QAM	PI/4 QPSK	8PSK	16 QAM	
FEC rate	1/2	3/4	3/4	1/2	3/4	3/4	1/2	3/4	3/4	
E_s/N_0 on AWGN	1,0	7,9	10,2	1,0	7,9	10,2	1,0	7,9	10,2	dB
$C/(N_0+I_0)$ threshold	43,8	50,7	53,0	46,8	53,7	56,0	49,9	56,8	59,1	dBHz
Minimum CQI value	[42]						48		84	dBHz

Table 5 – Summary of VDE transmission modes

The VDE burst of symbols is then sent into the Square Root Raised Cosine (SRRC), which performs the shaping of the symbol stream.

Finally, the last block of the VDE transmitter applies the increasing amplitude profile on the ramp up signal and the decreasing amplitude profile on the ramp down signal. The ramp up time from −50 dBc to −1.5 dBc of the power shall controlled rise time and occur in 416 µs. A gradual ramp-up period provides important spectral shaping to reduce energy spread outside the desired signal modulation bandwidth, and reduces interference to other users of the current and adjacent channel.

4.2 Receiver section architecture

The architecture of the VDE receiver section is shown in Figure 11.

The first block is the matched filter. Its peculiarity, compared to the shaping filter present in the transmitter, is that if the impulse response of the shaper filter is h (t), then its will be h (-t). In the frequency domain, this can be seen as follows: a frequency response of the matched filter is the complex conjugate with respect to that of the shaping filter. In the case of the SRRC filter, the time response is even symmetrical, and therefore the adapted filter coincides with the shaping one.

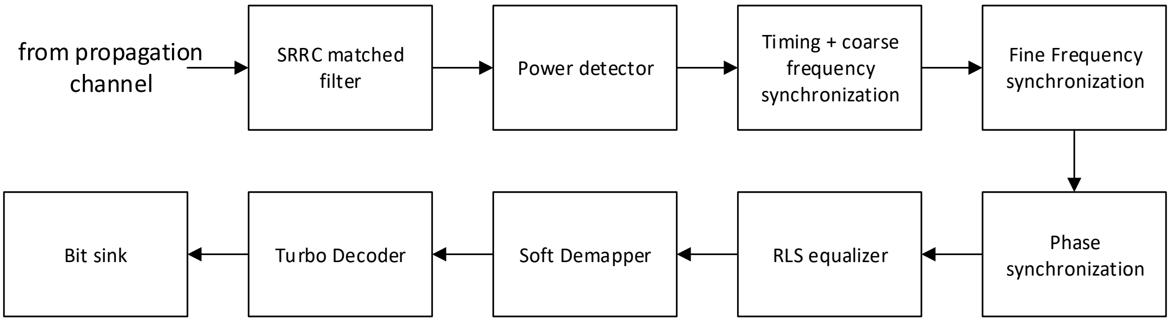


Figure 11 – Digital section of the VDE receiver architecture

The output of the matched filter $x[n]$ is conveyed into the power detector, composed by a sliding window of length equal to N_{SW}. The power detector calculates the average signal strength in the sliding window, according to

$$P_{SW} = \frac{1}{N_{SW}} \sum_{n=1}^{N_{SW}} |x[n]|^2 \quad (2)$$

The measured power is compared with a properly sized threshold. In case the threshold is exceeded, the power detector starts to create a vector of size equal to the VDE signal burst and passes it to the next block. Otherwise, if the threshold is not exceeded, the power detector discards the oldest sample and inserts a new sample before recalculating the power and comparing the value obtained with the threshold. The threshold is sized by calculating the average power of the noise calculated in real-time and scaling this value by a programmable factor, typically 2 or 4.

After detecting a VDE burst, the timing and frequency synchronisms need to be recovered. The first block after the power detector gathers the timing frequency by means of a cross-correlation between the received signal burst and the local replica of the pilot signal. The cross-correlation is insensitive to frequency offsets for the wider bandwidth configurations, i.e. 50 and 100 kHz. This means that for those configurations the timing synchronization obtained from the cross-correlation is reliable even in presence of ± 1 kHz frequency offset. For the narrowband configuration, i.e. 25 kHz a grid search over time and frequency domains should be employed. In other words, the cross-correlation is carried out for an equally spaced set of carrier frequency offsets. The absolute maximum of these cross-correlations gives the timing and the coarse frequency synchronizations.

Now let us observe Figure 12, which shows the missed packet probability in presence of frequency offset. This figure is specific for 25 kHz wide transmissions. As expected, the missed packet probability increases with wider frequency offset. This figure helps to dimension the grid search space over the frequency domain.

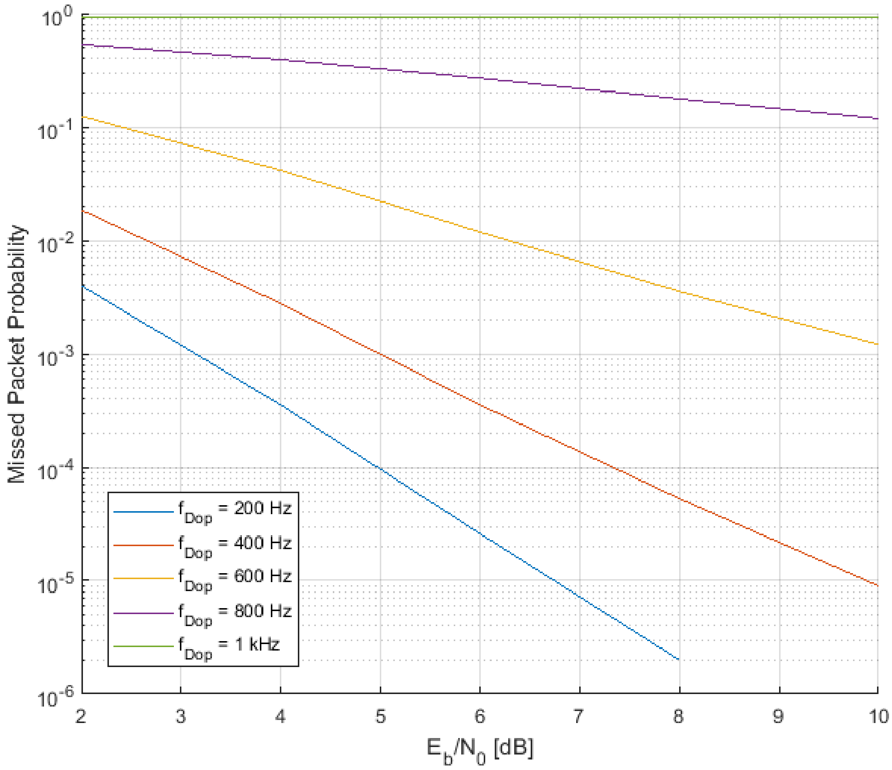


Figure 12 – Missed Packet probability in presence of frequency Doppler.

The following block performs the fine frequency synchronism by means of two algorithms, i.e. Luise&Reggiannini [3] and Viterbi&Viterbi [4]. First of all the L&R algorithm is applied, which enables to reach the Cramer-Rao bound:

$$\sigma_{CR}^2 = \frac{3}{2\pi^2 T_{sym}^2} \frac{1}{\rho N(N^2-1)} \quad (3)$$

where ρ is the signal-to-noise ratio, T_{sym} is the symbol period and N is the number of pilot symbols used for the frequency offset estimation. For the VDE-TER-MCS5.100 the CRB has the statement, shown in Figure 13.

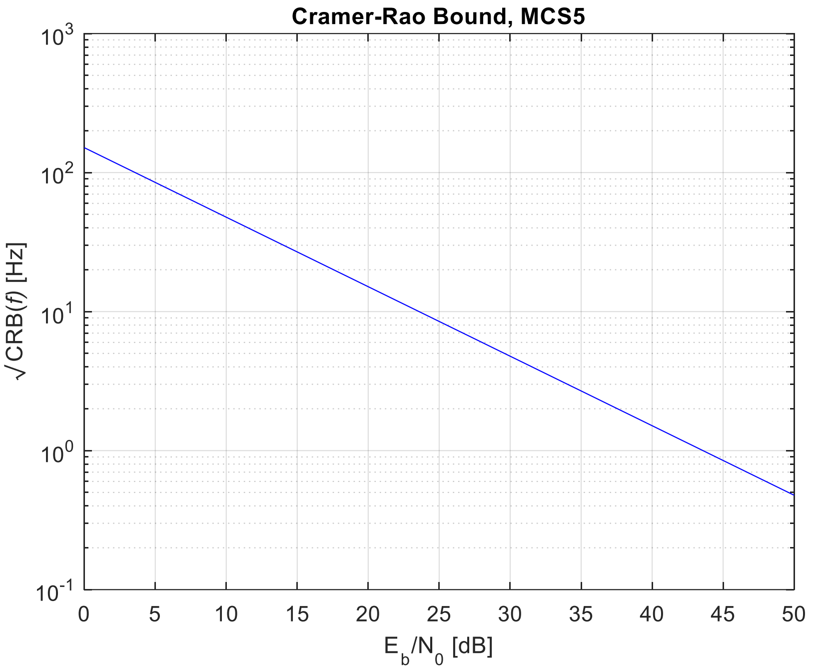


Figure 13 – CRLB: MCS-5.100 for L&R.

The lock range for the L&R algorithm is set as stated below

$$|\Delta f| \leq \left[(P+1)\cdot T_{sym}\right]^{-1} \tag{4}$$

where P is a design parameter and it is normally set equal to $N/2$. For the VDE-TER-MCS5.100 the lock range becomes

$$|\Delta f| \leq 5120\, Hz \tag{5}$$

Figure 13 shows that at $\frac{E_b}{N_0} = 20\, dB$ the frequency error is around 10 Hz. This frequency residual leads to a phase rotation over the burst duration of

$$2\pi f_{res} T_{burst} \sim 100° \tag{6}$$

which cannot be recovered by any phase recovery algorithm. For this reason we introduce an additional frequency recovery algorithm, i.e. the V&V one. The basic idea of this algorithm is to remove modulation from the data symbols and use the whole set of data as pilots.

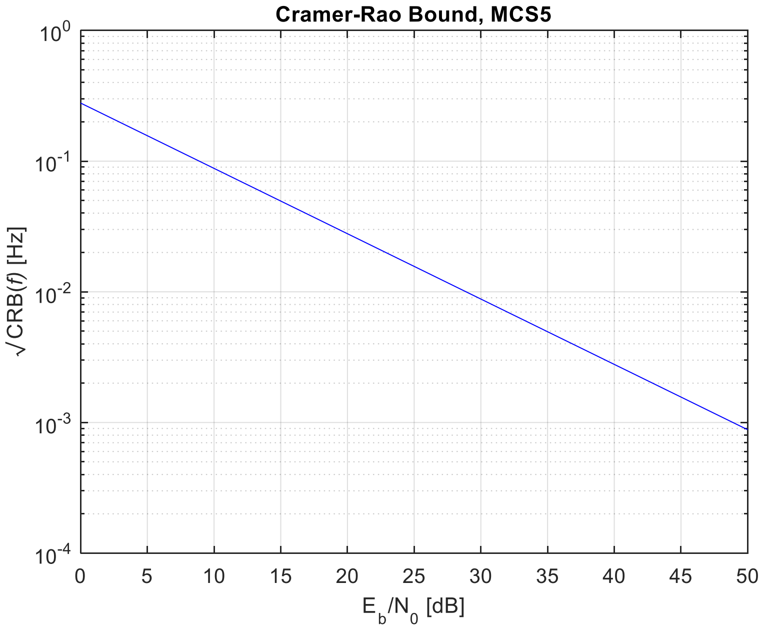


Figure 14 – CRLB: MCS-5.100 for the V&V

Since the number of pilots is now equal to the number of modulation symbols (pilot+data) in a VDE burst, the CRB limit should be recomputed. It is clear that, since a sub-optimum processing (removing data modulation) is applied, the CRB would not be exactly matched as for the L&R algorithm. However, the CRB in Figure 14 can give an idea of the target residual frequency error.

The lock range of this algorithm for the VDE-TER-MCS5.100 is

$$|\Delta f| \leq \left[M \cdot (L+1) \cdot T_{sym}\right]^{-1} \tag{7}$$

where M is the modulation order and L is a design parameter, set equal to 47. The lock range becomes

$$|\Delta f| \leq 133\ Hz \tag{8}$$

Figure 14 shows that at $\frac{E_b}{N_0} = 20\ dB$ the frequency error is around 0.03 Hz. This frequency residual leads to a phase rotation over the burst duration of

$$2\pi f_{res} T_{burst} \sim 0.5° \tag{9}$$

which can be easily recovered by the phase recovery algorithm.

The frequency corrected signal is then conveyed to the phase recovery block, which applies the following correction

$$\begin{cases} \varphi = \measuredangle \sum_{n=1}^{N} r(n) \cdot s^*(n) \\ x_{out}(t) = x_{in}(t) e^{-j\varphi} \end{cases} \tag{10}$$

where N is the number of pilot symbols over which the phase estimation is carried out.

The following block is the equalizer. The need of an equalization block is related to the marine propagation channel, which, in some cases, may be modelled as two rays channel. This type of propagation channel requires equalization. It is worth mentioning that each received burst should be considered as an independent transmission. In fact a stream of burst may be received from different ships or coastal stations. Moreover, even transmission from the same source are separated by time interval over which propagation channel condition may change drastically. For this reason we decided to restart the equalizer taps for any received burst. It is clear that a fast convergence of the equalizer taps is essential, in order to correctly received data after training the taps over the pilot signal. For all these reasons we selected the Recursive Least Square (RLS) equalizer for the VDE receiver. A generic scheme for a digital equalizer is reported in Figure 15.

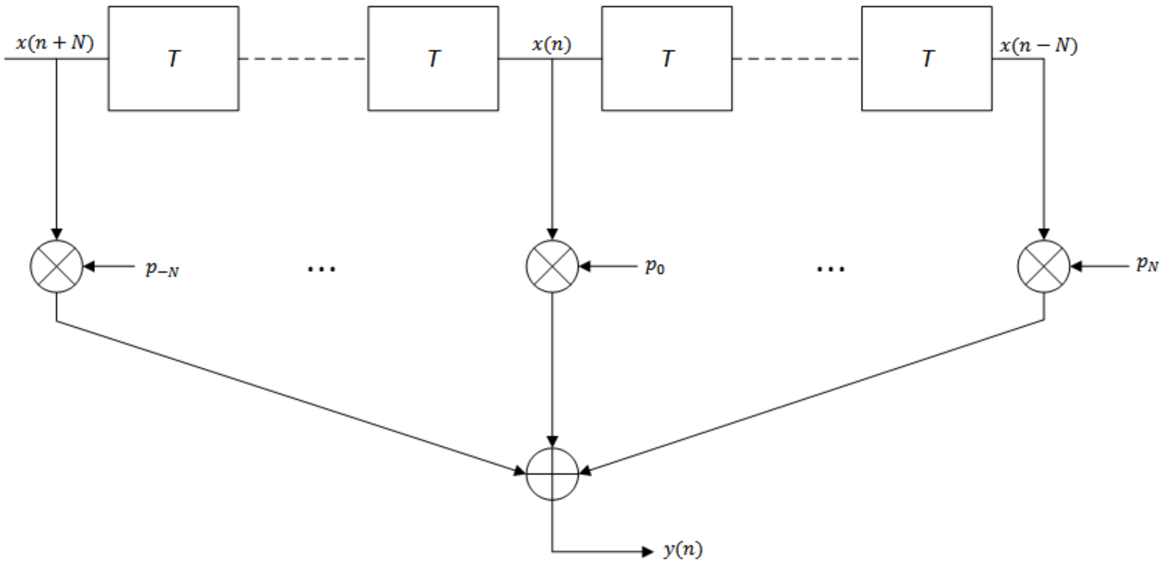


Figure 15 – Digital equalizer scheme

The basic idea of the RLS equalizer is that of minimizing the following cost function

$$WSE(n) = \sum_{k=0}^{n} w^{n-k} \left| \sum_{l=-N}^{N} p_l x(k-l) - c_k \right|^2 \tag{11}$$

where w is the forgetting factor and it should be set within $0 < w < 1$. This minimization is obtained by nulling the derivative of $WSE(n)$ w.r.t. the equalizer taps, i.e.

$$\frac{\partial WSE(n)}{p_l} = 0 \tag{12}$$

By inserting (11) into (12), we obtain

$$\sum_{l=-N}^{N} p_l \sum_{k=0}^{n} w^{n-k} x(k-l) x^*(k-m) = \sum_{k=0}^{n} w^{n-k} c_n x^*(k-m) \tag{13}$$

Now let us define, the vector of equalizer taps at the time instant $n+1$ as

$$\mathbf{p}(n+1) \triangleq [p_{-N}(n+1), \dots p_0(n+1), \dots, p_N(n+1)]^T \tag{14}$$

The element l, m of the matrix $\mathbf{R}(n)$ as

$$[\mathbf{R}(n)]_{l,m} = \sum_{k=0}^{n} w^{n-k} x(k-l) x^*(k-m) \tag{15}$$

and

$$\mathbf{d}(n) \triangleq [d_N(n), \dots d_0(n), \dots, d_{-N}(n)]^T \tag{16}$$

where

$$d_m(n) \triangleq \sum_{k=0}^{n} w^{n-k} c_n x^*(k-m) \tag{17}$$

Hence equation (13) can be rewritten in the matrix form as

$$\mathbf{R}(n)\mathbf{p}(n+1) = \mathbf{d}(n) \tag{18}$$

By observing equation (18), it is easy to understand that RLS equalizer taps at the time $n+1$ can be computed as

$$\mathbf{p}(n+1) = \mathbf{R}^{-1}(n)\mathbf{d}(n) \tag{19}$$

Finally the equation, which states the update of the equalization taps is the following

$$\mathbf{p}(n+1) = \mathbf{p}(n) - \mathbf{R}^{-1}(n)\mathbf{x}^*(n)e(n) \tag{20}$$

where the error $e(n)$ is defined as

$$e(n) = \mathbf{x}^T(n)\mathbf{p}(n) - c_n \tag{21}$$

It is important to observe that the matrix gain $\mathbf{R}^{-1}$ in place of the scalar gain γ, which is normally used for LMS equalizer, makes RLS much faster than LMS, i.e. its convergence time is shorter.

The output of the RLS equalizer is then sent to the soft demapper, which computes the soft information, a.k.a. the Log-Likelihood Ratio (LLR). The definition of LLR is reported hereafter

$$L(u_m) = log\left(\frac{\Pr(u_m = 1|\boldsymbol{y})}{\Pr(u_m = 0|\boldsymbol{y})}\right) \tag{22}$$

where u_m is the m-th bit in the received stream.

The output of the soft demapper is conveyed to the turbo decoder, which extracts the received information bits. The stream of received data bit is compared with the transmit data bit for performance assessment purposes.

5 VDES waveform software implementation

The basic idea for the VDE waveform implementation is that of mapping a single algorithm over a thread. The resulting software architecture is that of multi-threads working in *pipeline*. The concept above relies on the mechanism of data exchange among two consecutive threads, which is depicted in figure below.

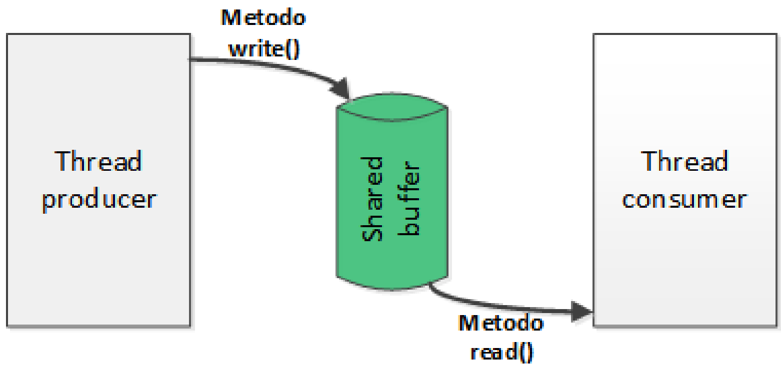


Figure 16 – Concept of thread data exchanges

It is important to note that the shared buffer object must have a read and a write method. In particular, the write method is used by the producer thread, while the read method is used by the consumer thread. It is clear that these methods of reading and writing must be regulated by a mechanism, which checks the validity of the data read / written. To clarify this point we make two examples, always referring to Figure 16. In the first example, we assume that the Thread producer has just finished writing updated data on the shared memory. Let us also assume that the Thread consumer is busy and cannot read the data just written into the shared memory. If the Thread producer were to write new data into the shared memory there would be a loss of data, since the overwritten data has not yet been read by the Thread consumer.

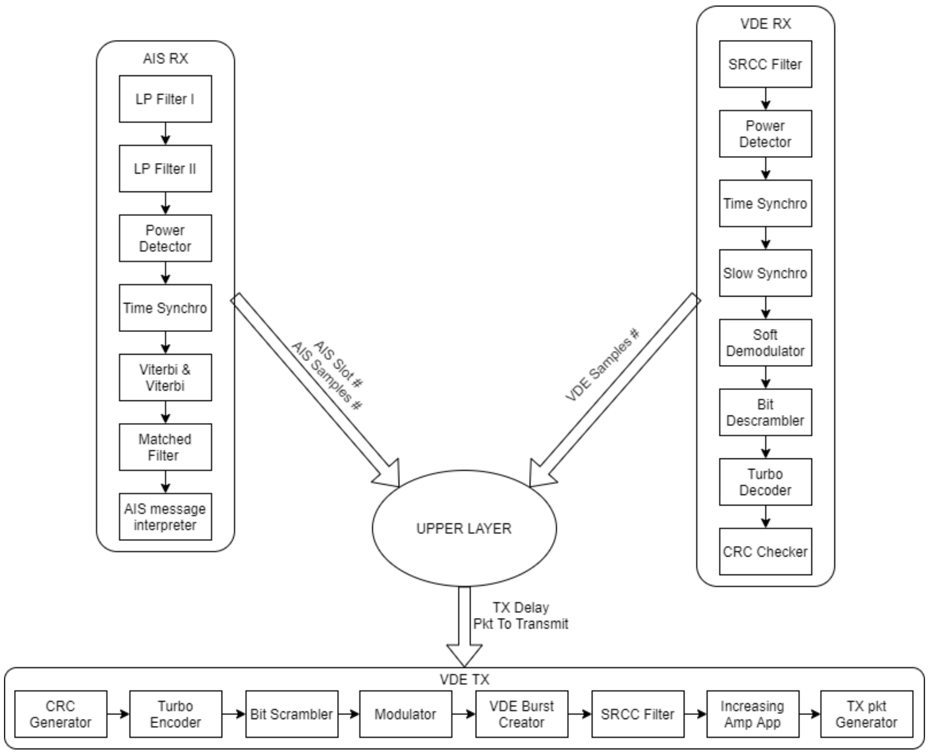


Figure 17 – Multi-thread SW architecture of the VDE transceiver

The following sub-sections describes the blocks (i.e., *threads* in our SDR-based implementation) belonging to the VDES Transceiver.

5.1 VDE TX

The VDE TX is a standard compliant transmitter implemented by means of the following blocks.

1. CRC Generator: the CRC generator thread appends a 32 bits CRC-32 check sequence to the last segment of the datagram.
2. Turbo Encoder: the turbo encoder thread takes the bits sequence from the CRC generator thread and applies to them a turbo code. The encoder consists of two recursive systematic convolutional (RSC) encoders concatenated in parallel. The turbo encoder thread performs also interleaving and rate adaptation by puncturing the encoder output.
3. Bit Scrambler: the bit scrambler thread performs a binary XOR between each element of the input bits sequence and the bit scrambler sequence.
4. Modulator: the modulator thread applies a modulation accordingly to the chosen Link ID at the input bits sequence. The Link IDs 11 and 17 are characterized by a constellation order equal to 4, whilst the Link ID 15 and the Link ID 19 are characterized by a constellation order equal to 8 and 16 respectively. The modulator thread then accordingly to the selected Link ID and therefore accordingly to the constellation order applies to its input bits sequence a π-4QPK, 8PSK or 16QAM modulation respectively.
5. VDE Burst Creator: the VDE burst creator thread creates the VDE burst (shown in Figure 18) to send over the air. In particular, it attaches to the head of its input data symbols, the ramp up symbols, the synchronization word symbols and the Link Configuration symbols. To the tail of the data symbols it also attaches the ramp down symbols sequence and the guard symbols.
6. SRCC Filter: this thread implements a Square-Root Raised Cosine pulse shap. One of the main drawbacks of all the signal waveforms is that although they can very well control the power emissions within the bandwidth of interest, they send relatively high amounts of power out of this one. A practical way of reducing the side-lobes of the spectrum of the navigation signals could be to use a SRRC filter since this has a limited bandwidth. Therefore this thread filters in the frequency domain the symbols received from the VDE burst creator thread passes them to the next thread in the transmission chain.
7. Increasing Amp Application: this thread simply increases and normalizes the power of the ramp up and ramp down symbols.

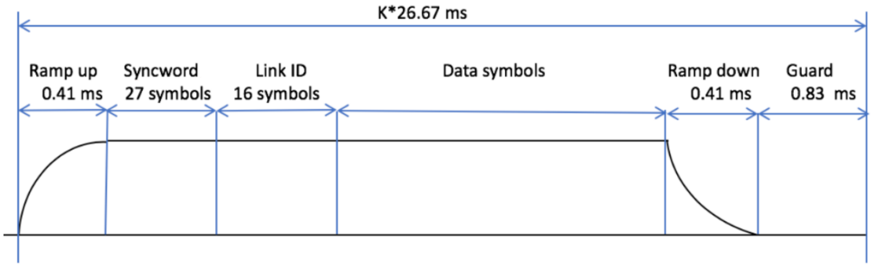


Figure 18 VDE-TER General Packet Format

8. TX Pkt Generator: the TX pkt generator thread performs a simple but crucial role. Due the lack of a GPSDO chipset on the VDES hardware platform able to timestamp the received samples at the receiver front end, tx pkt generator sends continuously burst of zeros to the transmitter front end to maintain the time synchronization once the UTC time slot has been derived by the AIS receiver: the signal is not transmitted, but the synchronism is maintained. If a useful VDE burst has to be transmitted, this thread reading in an asynchronous way from the previous thread is able to understand to stop the transmission of zeros and start the transmission of the useful data coming from the previous thread of the TX chain.

5.2 VDE RX

The VDE RX is a standard-compliant SDR-based receiver deployed accordingly to the VDE TX above described and implemented by means of the following blocks.

1. SRRC Filter: this thread implements a filter matched to the SRCC filter used in the VDE TX chain.
2. Power Detector: this thread aims to detect the useful received signal. It first calculates the noise floor and the accordingly to such measurements identifies the signal and passes it to the time synchro thread deleting the possible presence of noise between useful bursts. The power detector thread forwards to the next thread a queue containing in the first byte a flag indicating the presence of useful signal, in the second byte the number of processed samples and then the useful signal if present. If no signal is detected, the flag and the useful signal bytes are set to zero. In this way the following threads are able to understand if they have to process the input queues or not.
3. Time Synchro: the time synchro thread measures the frequency offset and the time delay of the received signal. It first calculates the cross-correlation between the synchronism sequence and the received signals in the frequency domain, then it looks for the cross-correlation peak, identifies the frequency offset and time delay of the received signal. Finally, the received signal is corrected by means of a correction factor calculated accordingly to the above mentioned parameters and then it is passes to the slow synchro thread.
4. Slow Synchro: the slow synchro thread aims to perform a fine frequency offset recover to minimize the Doppler effect on the received signal. It first extracts from the received signal the pilot symbols and then applies to these symbols the Luise-Reggiannini algorithm to perform a first coarse frequency offset measurement. Then it applies the Viterbi and Viterbi algorithm to the pilots and data symbols to perform a fine frequency offset measurement. The offset is finally applied to the received signal and it is pass to the soft modulator thread.
5. Soft Demodulator: the soft modulator thread extracts the original information-bearing signal. The information content from the modulated carrier wave is recovered here.
6. Bit Descrambler: the bit descrambler thread simply applies the inverse scrambling operation used in the transmission phase by the bit scrambler.
7. Turbo Decoder: the turbo decoder thread performs a turbo decoding and the hard decision on the information block, before passing the bits sequence to the last block of the receiving chain.

8. CRC Checker: the CRC checker thread checks if the received bits sequence is corrupted. If received CRC is right the bits are passed to the upper layer that accordingly to the received message performs the necessary actions.

5.3 The Upper Layer

The upper layer thread firstly manages the UTC time synchronization. It in fact extracts from particular AIS messages the slot number and the processed samples related to such message. The AIS containing the slot number are the "Position Report" message and "Base Station Report" message.
In the initialization phase, the upper layer thread knowing the AIS slot number and the AIS related processed samples, calculates the delay in terms of samples that the VDE TX will have to wait before a transmission, so that it will be synchronized with the UTC time. Such delay is then passed to the TX Pkt Generator thread (by means of a protected value) that will send zeros until the synchronization is reached. From this moment on, the upper layer thread will remain synchronized due to the fact that the VDE TX by means of the TX Pkt generator thread sends continuously null packets of duration equal to a TDMA slot (i.e. 26.67 ms) if no useful data has to be sent over the VDE channels.
The upper layer also manages the received message from the VDE RX, and based on that carries out the right action (i.e reply to received message or store the received information).
The tasks of the upper layer are:

- VDE transmission delay in terms of samples to get UTC synchronization;
- Time slot calculation accordingly to the AIS slot number and AIS processed samples (through the *"time_slot_calculation()"* method);
- Announcement Message preparation (through the "*resource_allocation_pkt_generator()"* and *"resource_request_pkt_generator()"* methods);
- Announcement Message interpretation (through the "*vde_as_interpreter()"* method);
- Data Message Preparation (through the "*weather_info_pkt_generator()*" and "*evacuation_plan_pkt_generator()*" methods);
- Data Message interpretation (through the "*vde_data_interpreter()"* method);

6 COTS hardware platform specifications

In this section it is described the hardware prototype of the VDES system with a description of the COTS boards composing the solution.

6.1 VDES - The Software Defined Radio (SDR) Solution

The introduction of VDES channels in the marine band requires an evolution in radio design. Following intensive research and consultations by THALIT and WISER, a decision to employ a Software Defined Radio (SDR) solution was agreed.

As SDR hardware platform the partners selected a combination of radio and carrier boards able to optimize the development of an optimised SDR based solution implementing a complete high performance VDES solution, whilst retaining the flexibility to support specific OEM/ODM needs and future evolution of the standard.

This architecture consists of a Base-Band (BB) System-on-Chip (SoC) paired with Radio Frequency (RF) Soc. The BB SoC contains Field Programmable Gate Array (FPGA) fabric and ARM dual-core Cortex A9 processor.

For the development of the prototype the Xilinx Zynq®-7000 ZC706 containing the 7020 FPGA SoC is chosen, which provides a low-cost and well supported back-end for the signal processing functionalities. On the RF programmable transceiver SoC, initial evaluation took place using the ADI EVAL-ADRV9002 RF board containing the ADRV9002 RF SoC.

The VDES SDR hardware prototype is composed by the following boards:

- Analog Device EVAL-ADRV9002 evaluation board as radio front-end
- Xilinx Zynq®-7000 SoC ZC706 Evaluation Kit as carrier board

In Figure 19 is represented a picture showing the Xilinx ZC706 connected to the ADI EVAL-ADRV9002 through a FMC (FPGA Mezzanine Card) interface.

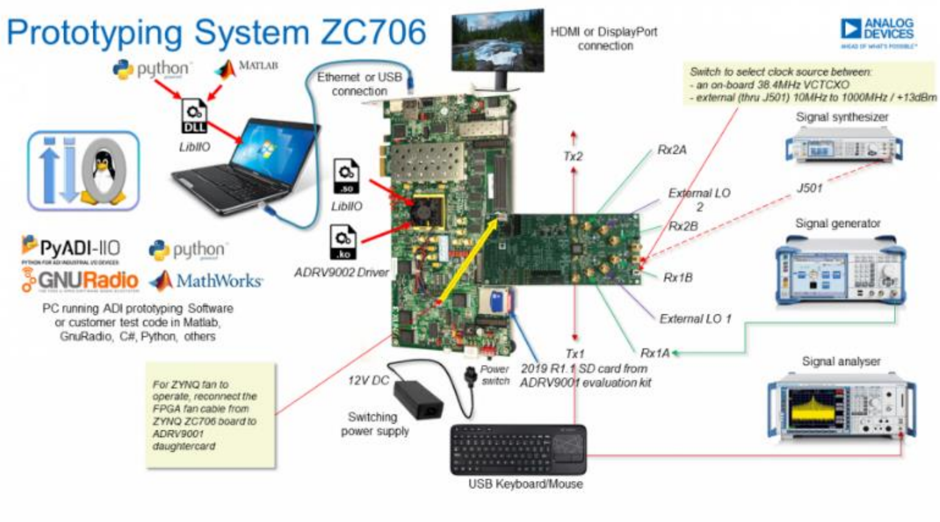


Figure 19: ADRV9002 + Zynq ZC706 boards.

6.2 RF front-end section

This architecture consists of a Base-Band (BB) System-on-Chip (SoC) paired with Radio Frequency (RF) Soc. The BB SoC contains Field Programmable Gate Array (FPGA) fabric and ARM dual-core Cortex A9 processor.

For initial development, the Avnet Zedboard containing the Xilinx Zynq 7020 FPGA SoC is chosen, which provides a low-cost and well supported back-end for the signal processing functionalities. On the RF programmable transceiver SoC, initial evaluation took place using the Lime Micro Myriad RF containing the LMS6002D RF SoC.

6.2.1 EVAL-ADRV9002

The EVAL-ADRV9002 evaluation boards are radio cards designed to showcase the ADRV9002, dual-channel Narrow/Wide-band RF transceiver. The radio cards provide a 2x2 transceiver platform for device evaluation. All peripherals necessary for the radio card to operate include high efficiency power circuit board, and a high-performance clocking/MCS distribution included on the radio board. Connecting one of the radio cards with the FPGA motherboard through the FMC connector form a complete evaluation platform for ADRV9002.

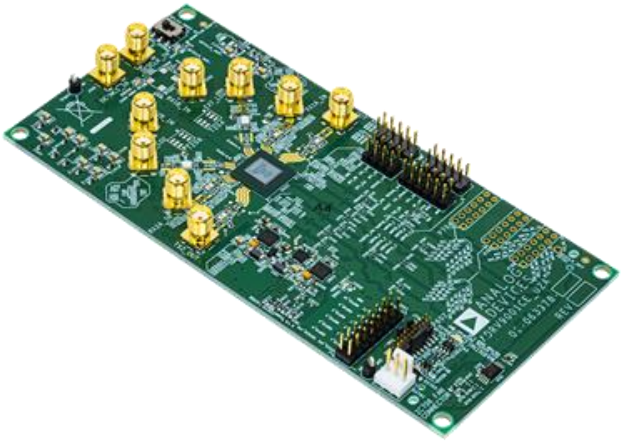


Figure 20: ADI EVAL-ADRV9002 evaluation board.

6.2.2 ADI ADRV9002

The ADRV9002 is a highly integrated, RF transceiver that has dual-channel transmitters, dual-channel receivers, integrated synthesizers, and digital signal processing functions. ADRV9002 transceiver with minimum number of external components can be utilized to build complete RF-to-bits signal chain that can serve as RF front end in VDES type applications.

The IC delivers a versatile combination of high performance and low power consumption required by battery powered radio equipment and can operate in both FDD and TDD modes. The ADRV9002 operates from 30 MHz to 6000 MHz and covers the UHF, VHF, licensed and unlicensed cellular bands, and industrial, scientific, and medical (ISM) bands. The IC can support both narrowband and wideband standards up to 40 MHz bandwidth on both receive and transmit.

The transceiver consists of direct conversion signal paths with state-of-the-art noise figure and linearity. Each complete receiver and transmitter subsystem includes dc offset correction, quadrature error correction, and programmable digital filters, which eliminate the need for these functions in the digital baseband. In addition, several auxiliary functions such as auxiliary analog-to-digital converters (ADCs), auxiliary digital-to-analog converters (DACs), and general-purpose input/outputs (GPIOs) are integrated to provide additional monitoring and control capability.

The fully integrated phase-locked loops (PLLs) provide high performance, low power, fractional-N frequency synthesis for the transmitter, receiver, and clock sections. Careful design and layout techniques provide the isolation required in high performance personal radio applications.

All voltage-controlled oscillator (VCO) and loop filter components are integrated to minimize the external component count. The local oscillators (LOs) have flexible configuration options and include fast lock modes. The transceiver includes low power sleep and monitor modes to save power and extend the battery life of portable devices while monitoring communication.

The fully integrated, low power digital predistortion (DPD) is optimized for both narrowband and wideband signals and enables linearization of high efficiency power amplifiers.

The ADRV9002 core can be powered directly from 1.0 V, 1.3 V, and 1.8 V regulators and is controlled via a standard 4-wire serial port. Other voltage supplies are used to provide proper digital interface levels and to optimize receiver, transmitter, and auxiliary converter performance.

High data rate and low data rate interfaces are supported using configurable complementary metal-oxide semiconductors (CMOS) or low voltage differential signaling (LVDS) serial synchronous interface (SSI) choice.

6.3 Embedded processor section

The EVAL-ADRV9002 evaluation board is by definition a “FPGA mezzanine card” (FMC) that means it needs a carrier to plug into. For developing the VDES radio we selected as carrier the Xilinx Zynq®-7000 SoC ZC706 evaluation board.

6.3.1 Xilinx ZC706 Evaluation Board

The Zynq®-7000 SoC ZC706 Evaluation Kit includes all the basic components of hardware, design tools, IP, and pre-verified reference designs including a targeted design, enabling a complete embedded processing platform and transceiver-based designs including PCIe. The included pre-verified reference designs and industry-standard FPGA Mezzanine Connectors (FMC) allow scaling and customization with daughter cards like the ADI EVAL-ADRV902.

The evaluation board comes with the following key features:

- Optimized for quickly prototyping embedded applications using Zynq-7000 SoCs
- Hardware, design tools, IP, and pre-verified reference designs
- Demonstrates a embedded design, targeting video pipeline
- Advanced memory interface with
- 1GB DDR3 Component Memory
- 1GB DDR3 SODIM Memory
- Enabling serial connectivity with PCIe Gen2x4, SFP+ and SMA Pairs, USB OTG, UART, IIC
- Supports embedded processing with Dual ARM Cortex-A9 core processors
- Develop networking applications with 10-100-1000 Mbps Ethernet (RGMII)
- Implement Video display applications with HDMI out
- Expand I/O with the FPGA Mezzanine Card (FMC) interface

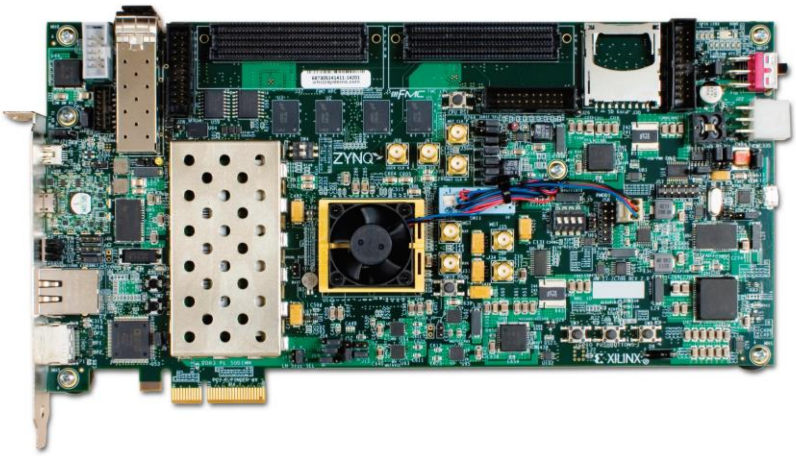


Figure 21: Xilinx ZC706 evaluation board.

For more details, please refer to the ZC706 board user manual[1].

6.3.2 Zynq-7000 XC7Z045 SoC

The ZC706 evaluation board is populated with the Zynq-7000 XC7Z045-2FFG900C SoC, which consists of an integrated processing system (PS) and programmable logic (PL), on a single die. The high-level block diagram is shown in Figure 22.

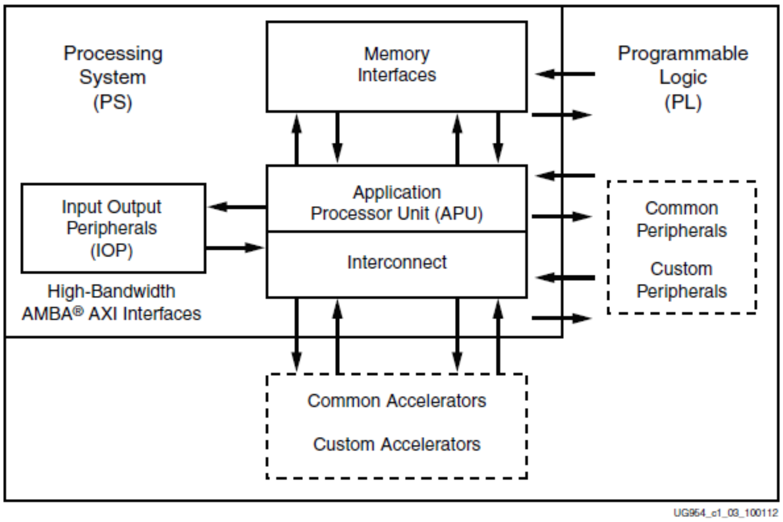


Figure 22: Zynq-7000 XC7Z045 High Level Block Diagram.

The PS integrates two ARM® Cortex™-A9 MPCore™ application processors, AMBA® interconnect, internal memories, external memory interfaces, and peripherals including USB, Ethernet, SPI, SD/SDIO, I2C, CAN, UART, and GPIO. The PS runs independently of the PL and boots at power-up or reset.

A system level block diagram is shown in Figure 23.

---

1 https://www.xilinx.com/support/documentation/boards_and_kits/zc706/ug954-zc706-eval-board-xc7z045-ap-soc.pdf

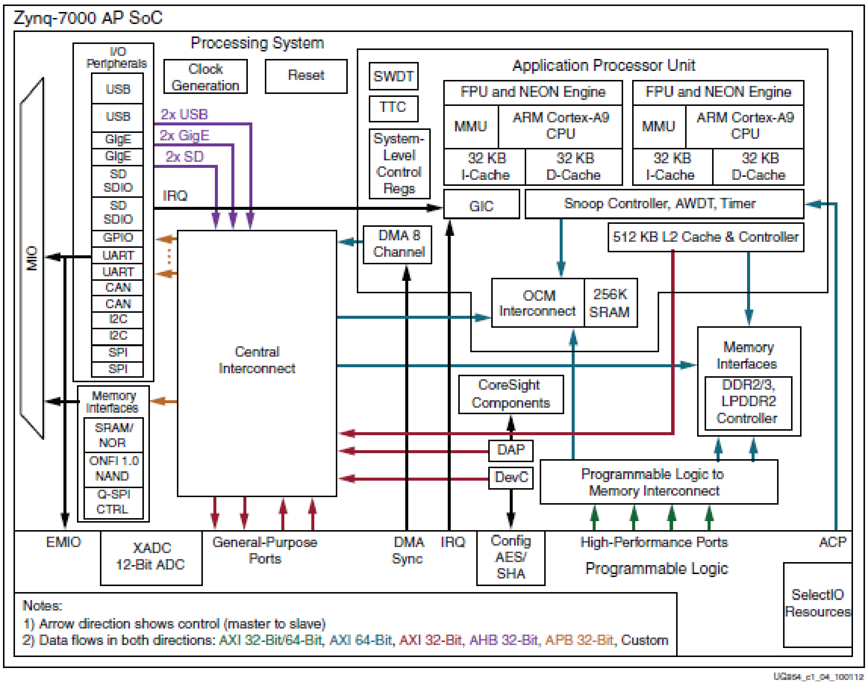


Figure 23: Zynq-7000 Block Diagram.

For additional information on Zynq-7000 SoC devices, see Zynq-7000 SoC Overview[2] and Zynq-7000 SoC Technical Reference Manual[3].

7 Porting over the selected embedded processor

The porting of the VDES transceiver on the selected embedded processor (see section 6.3) will follow an incremental development approach as described below.

As a rule of thumb, for both the Transmitter and Receiver sections, the high level software architecture of the ported VDES transceiver will match that of the VDES simulator used for performance assessment (see section 4).

Thus, the ported VDES transceiver software structure, in terms of components, interfaces, protocols and data exchanged at runtime in the transmission and reception chains, will remain unchanged with respect to the VDES simulator unless adaptations will be mandatory due to performance and/or optimizations constraints induced by the capabilities of selected embedded processor.

In particular, the VDES simulator code, initially implemented in MATLAB and then refactored in C++ to run on a Linux PC, will be cross-compiled for the selected embedded processor with the required adaptations in terms of used APIs and libraries. This process will rely on the SDK, designed by the manufacturer and accompanying the selected COTS hardware platform (see section 6).

In a first step, the ported VDES transceiver code will thus execute at runtime in the Linux User Space. Depending on the achieved performances, measured first on a laboratory test bench, in regard of the requirements of the VDES standard (see section 2), it will be then assessed if

[2] www.xilinx.com/support/documentation/data_sheets/ds190-Zynq-7000-Overview.pdf
[3] www.xilinx.com/support/documentation/user_guides/ug585-Zynq-7000-TRM.pdf

it is necessary to move some of the ported VDES transceiver components to the Linux Kernel Space, by refactoring them in C, and/or to the FPGA of the selected COTS hardware platform (see section 5.2), by refactoring them in VHDL.

In addition, the ported VDES transceiver code will be adapted, with respect to the VDES simulator code, in order to interface the RF front end section of the selected COTS hardware platform (see section 6). This interface will rely on the C APIs, designed by the manufacturer, to initialize, configure, program, and control the RF front end both in transmission and reception.

During the development , we have seen that the time execution of Turbo Encoder block (see section 4.1) and Turbo Decoder block (see section 4.2) was too large for VDES timing. Than we have implemented this two blocks on FPGA. Furthermore on microprocessor part we have implemented a linux divice driver to interface with new fpga module.

7.1 Hardware implementation of turbo encoder/decoder

To obtain a flexible system it is preferable to implement each component of the waveform in a software module.

The FEC system encoder is a part that doesn't require a lot of resource in term of complexity and computing time.

The decoder instead is a complex component, if it is implemented over a GPP it needs a long processing time because this device type has a limited parallelism. In VDES platform the processing time of a software turbo decoder (computed by PS) is too long respect to the reception slots.

To reduce the decoder times the encoder/decoder subsystems are moved to the Programmable Logic (PL) part of Zynq-7000.

The encoding/decoding functions are implemented by a couple of IP cores (turbo encoder and turbo decoder).

These IP are integrated into an AXI peripheral and connected to PS of Zynq-7000.

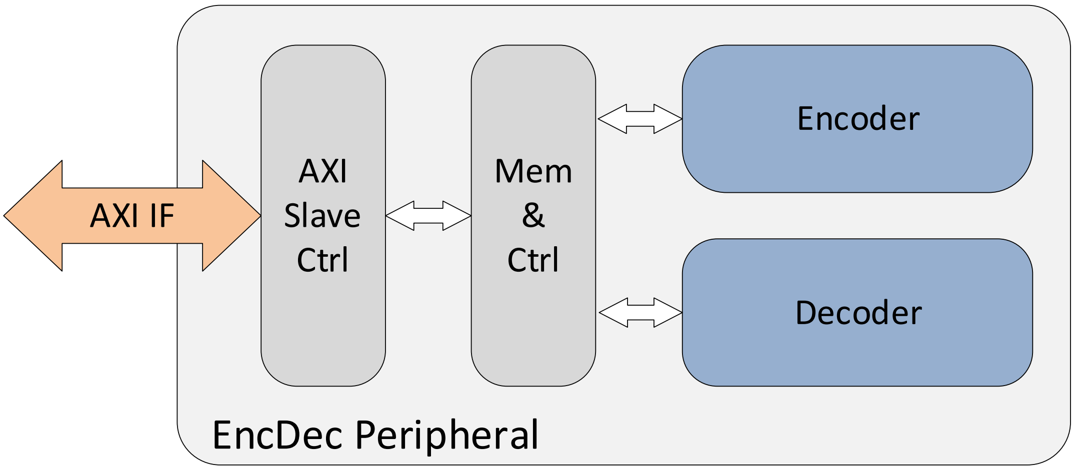


Figure 24 - Encoder/decoder peripheral

The waveform software accesses to the HW FEC through a specific address space that allows to control the peripheral and the data exchange.

The address space is partitioned according to Table 6.

Start address	End address	Name	Description
0x0000	0x7FFF	REG_CTRL	Register bank to control the peripheral
0x8000	0x9FFF	DATA_ENC_IN	RAM containing the input data of encoder
0xA000	0xBFFF	DATA_ENC_OUT	RAM containing the output data of encoder
0xC000	0xDFFF	DATA_DEC_IN	RAM containing the input data of decoder
0xE000	0xFFFF	DATA_DEC_OUT	RAM containing the output data of decoder

Table 6 - Peripheral memory partitions

7.1.1 Registers description

The REG_CTRL section contains 6 registers that control the peripheral.

Each register has a width of 32 bit.

REG_VERSION register (0x0000)

This is a Read Only (RO) register that contains the peripheral version. It has the following structure:

Bit range	Name
31…24	MAJOR
23…16	MINOR
15…8	REVISION
7…0	BUILD_NUMBER

Table 7 - REG_VERSION structure

REG_ENABLE register (0x0004)

It is a Read Write (RW) register that controls the peripheral activation state.

Bit range	Name	Reset value
0	ENABLE	0

Table 8 - REG_ENABLE structure

If ENABLE bit is false the peripheral is inactive and locked in the reset state, all other registers are forced to the reset values.

If ENABLE is true the peripheral is enabled, all others registers are operative.

REG_ENC_CTRL register (0x0008)

This is a RW register that controls the encoder subsystem.

Bit range	Name	Reset value
24	INT_ACK	0
16	INT_EN	0
4…0	MODE	0

Table 9 - REG_ENC_CTRL structure

The MODE field sets the link configuration ID of the next encoding process. Writing this field, the encoder starts.

If INT_EN is true the encoder interrupt system is enabled otherwise it is disabled.

If a true value is written to the INT_ACK bit the encoder interrupt flag is cleared.

REG_ENC_STATE register (0x000C)

This is a RO register that provides the current encoder state.

Bit range	Name	Reset value
16	INT_FLAG	0
2…0	STATE	0

Table 10 - REG_ENC_STATE structure

The STATE field contains the encoder state, it can assume the following values:

0 DISABLED when the peripheral is disabled (see §0);
1 IDLE when the previous encoding process is done and the output is available in output data memory (DATA_ENC_OUT), the encoder is inactive and ready for a new encoding;
2 LOAD when the encoder is reading the input data memory (DATA_ENC_IN);
3 BUSY when the encoder is working;
4 ERROR when a bad encoder mode is written into the MODE field, the encoder is ready for a new elaboration.

If the encoder raises an interrupt event the INT_FLAG goes high. To reset this interrupt flag is necessary to write INT_ACK bit of REG_ENC_CTRL.

REG_DEC_CTRL register (0x0010)

This is a RW register that controls the decoder subsystem.

Bit range	Name	Reset value
24	INT_ACK	0
16	INT_EN	0
11…8	ITER	0
4…0	MODE	0

Table 11 - REG_DEC_CTRL structure

The MODE field sets the link configuration ID of the next decoding process.

The ITER field sets the iterations number of the next decoding process.

Writing the MODE and ITER fields at the same time starts the decoder.

If INT_EN is true the decoder interrupt system is enabled otherwise it is disabled.

If a true value is written to the INT_ACK bit the decoder interrupt flag is cleared.

REG_DEC_STATE register (0x0014)

This is a RO register that provides the current decoder state.

Bit range	Name	Reset value
16	INT_FLAG	0
2…0	STATE	0

Table 12 - REG_DEC_STATE structure

The STATE field contains the decoder state, it can assume the following values:

0 DISABLED when the peripheral is disabled (see §0);

1 IDLE when the previous decoding process is done and the output is available in output data memory (DATA_DEC_OUT), the decoder is inactive and ready for a new decoding;

2 LOAD when the decoder is reading the input data memory (DATA_DEC_IN);

3 BUSY when the decoder is working;

4 ERROR when a bad decoder mode is written into the MODE field, the decoder is ready for a new elaboration.

If the decoder raises an interrupt event the INT_FLAG goes high. To reset this interrupt flag is necessary to write INT_ACK bit of REG_DEC_CTRL.

DATA_ENC_IN memory (0x8000 – 9FFF)

This memory contains the encoder input data, is written by the PS and read by the encoder.

Each byte contains one useful bit (carried in the LSB).

DATA_ENC_OUT memory (0xA000 – BFFF)

This memory contains the encoder output data, is written by the encoder and read by the PS.

Each byte contains 8 useful bits (MSB first).

DATA_DEC_IN memory (0xC000 – DFFF)

This memory contains the decoder input data, is written by the PS and read by the decoder.

Each byte contains one useful soft-bit (2's complement 8-bit word).

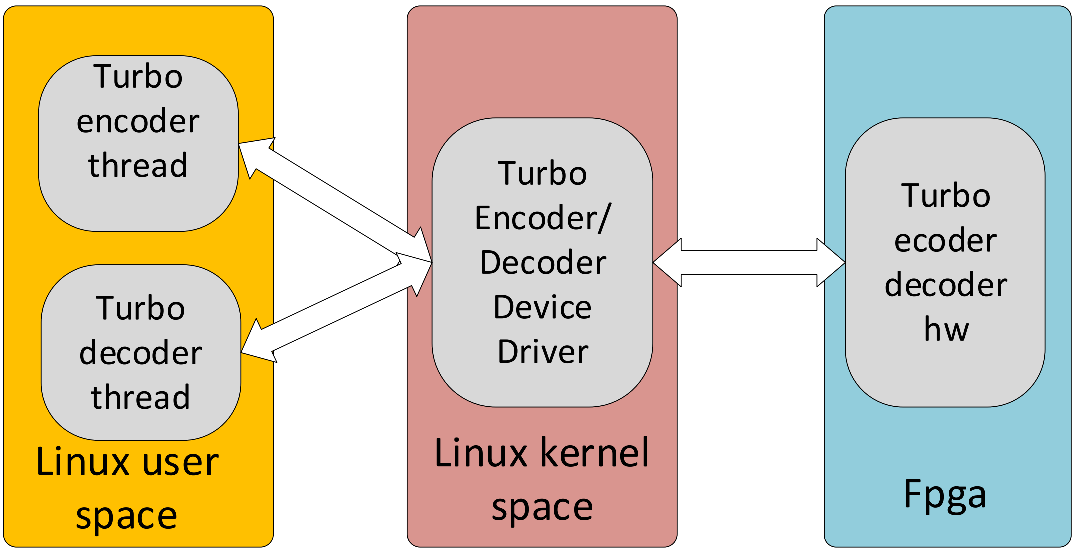
DATA_DEC_OUT memory (0xE000 – FFFF)

This memory contains the decoder output data, is written by the decoder and read by the PS.

Each byte contains 8 useful bits (MSB first).

7.2 Linux device driver implementation for turbo encoder/decoder on fpga

7.2.1 Block diagram



7.2.2 Device driver description

Linux device driver implement the following IOCTL functions:

- WR_ENCODER
  It is used to write the fpga RAM containing the input data to encoder
- RD_END_ENCODER
  It is used to read the end of encoder operation
- RD_ENCODER
  It is used to read the fpga RAM containing the output data encoded
- WR_DECODER
  It is used to write the fpga RAM containing the input data to decoder
- RD_END_DECODER
  It is used to read the end of decoder operation
- RD_DECODER
  It is used to read the fpga RAM containing the output data decoded
- RD_VERSION
  It is used to read the fpga turbo module version
- WR_MODE
  It is used to write the VDES mode

- WR_NUM_ITERATION
  It is used to write the NUM_ITERATION parameter

7.2.3 Encoder Thread operazion description

- Initialization sequence
  Read Version calling RD_VERSION ioctl function
  Set VDES mode calling WR_MODE ioctl function

- Runtime sequence
  Write the input data to encoder calling WR_ENCODER ioctl function
  Wait the end of encoder calling RD_END_ENCODER ioctl function
  Read the output data encoded calling RD_ENCODER ioctl function

7.2.4 Decoder Thread operazion description

- Initialization sequence
  Read Version calling RD_VERSION ioctl function
  Set VDES mode calling WR_MODE ioctl function
  Set number iteration parameter calling WR_NUM_ITERATION ioctl function

- Runtime sequence
  - Write the input data to decoder calling WR_DECODER ioctl function
  - Wait the end of decoder calling RD_END_DECODER ioctl function
  - Read the output data decoded calling RD_DECODER ioctl function

8 Interfaces

The purpose of this Chapter is to describe the communication model, protocol and messages used by the VDES gateway software developed in Thales Italia as a gateway between the VDES radio and the shore and ship ICT Data Fusion Bus (DFB) systems use in the Palaemon financed project. The purpose of the VDES gateway module is to allow the bidirectional data transfer between shore and ship components for demonstration purposes in the PALAEMON funded project. It is also a component of the Weather Forecast Tool (WFT) deliverable (see (AAVV, 2021) D3.6 Development of PALAEMOM Weather Forecast Tool (v2)).

8.1 Introduction

8.1.1 Purpose & Scope

This document contains a detailed description of the communication channels used by the two VDES Gateway applications running on Shore and Ship side of the demonstrator used for the following use cases inside the Palaemon project:

- AIS data extraction use case
- Weather info use case
- Evacuation info use case

This document contains also the detailed description of the interfaces with the external components the two VDES Gateway application.

In the remaining of this document we refer to the two VDES Gateway applications with the names VDESGW_Shore and VDESGW_Ship.

8.1.2 Use Case 1: AIS position extraction

This use case is used as a demonstration of the capability of the VDES radio to interface and extract information from the AIS radio system. As a sample of the data that can be extracted from the AIS system, the **ship position** is used. The data are acquired by both VDES radio on shore and ship side, and it is provided to VDES GW. VDES GW forwards the data to ICT DFB on both sides (see figure). Moreover, this information will be kept on the Ship ICT Data Fusion Bus for tracking purposes. To cite an example, at the end of every voyage, a report will be carried, abridging all the information generated by the PALAEMON components (alongside the shipboard legacy systems that are connected to our platform).

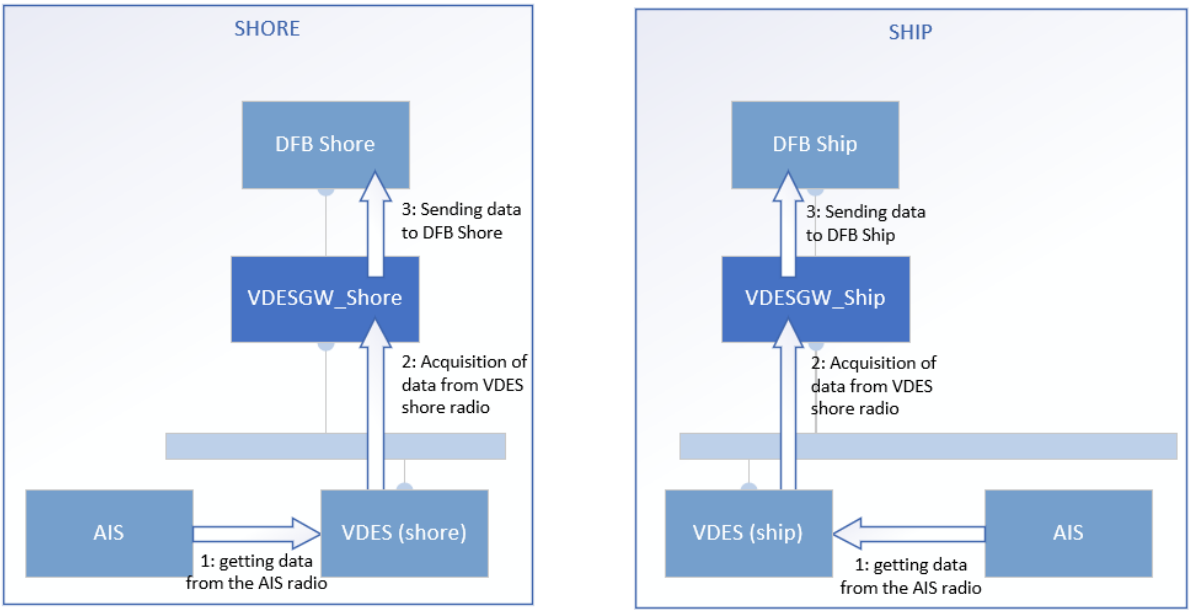


Figure 25 AIS data extraction use case

- **Step 1**: The VDES Ship and Shore radios interfaces the AIS system and extracts the ship position data. The details of this interface are outside the scope of this document, refer to [5], chapter 3. The AIS messages used to extract the position are the number 1, 2 and 3 (Position reports).
- **Step 2**: The VDES Ship and Shore radio publishes the received data in the LAN and the VDES GW applications acquire it. The received data are stored by the application for usage in other scenarios.
- **Step 3**: the VDES GW applications publish the data to the ICT DFB systems.

8.1.3 Use Case 2: Weather info use case

In this use case the VDESGW_Shore application polls the Danaos Weather data service to get the current forecast of weather condition (next three hours) in the current position of the ship, and send this weather data directly to DFB shore and to indirectly to DFB Ship using the VDES channel:

- **Step 1**: If the VDESGW_Shore application has a valid and fresh position of the ship, obtained from the Use Case 1, it polls periodically the Danaos Weather data service to get the current weather condition in the position where the ship is placed.
- **Step 2**: The VDESGW_Shore application sends the weather data to the ICT DFB Shore
- **Step 3**: The VDESGW_Shore application sends to the VDES Shore radio the request to send the Weather data to the Ship
- **Step 4**: The VDES Shore radio sends the data to the VDES Ship radio
- **Step 5**: The VDES Ship radio publish the received data towards the VDESGW_Ship application
- **Step 6**: the VDESGW_Ship application sends the weather data to the ICT DFB Ship

For additional details, please refer to [6] D3.6 Development of PALAEMON Weather Forecast Tool (v2).

8.1.4 Use Case 3: Evacuation use case

This use case demonstrates the transfer of data between ICS Ship and VDESGW_Shore (and possibly the ICT DFB Shore) using the VDES radio channel. The “Ship Evacuation command” is used as a sample of data to be transferred:

- **Step 1**: the ICT DFB Ship component publishes an “evacuation coordinator” topic; the topic is received from the VDESGW_Ship application.
- **Step 2**: The VDESGW_Ship application sends to the VDES Ship radio the request to send the evacuation command to the Shore
- **Step 3**: The VDES Ship radio sends the data to the VDES Shore radio
- **Step 4**: The VDES Shore radio publishes the received data towards the VDESGW_Shore application for log purposes
- **Step 5:** If the evacuation status contained in the evacuation coordinator message indicates that an evacuation is in progress, the VDESGW_Ship starts to periodically send a “mayday message” to the VDES Ship radio. The mayday message is forwarded from the ship to the ICT shore final recipient following the usual sequence
- **Step 6:** If the evacuation status contained in the evacuation coordinator message indicates that an evacuation is **NOT** in progress, the VDESGW_Ship stops to send a “mayday message” to the VDES Ship radio. The mayday message is then no more sent to the Shore side (VDES, VDESGW, ICT)
-

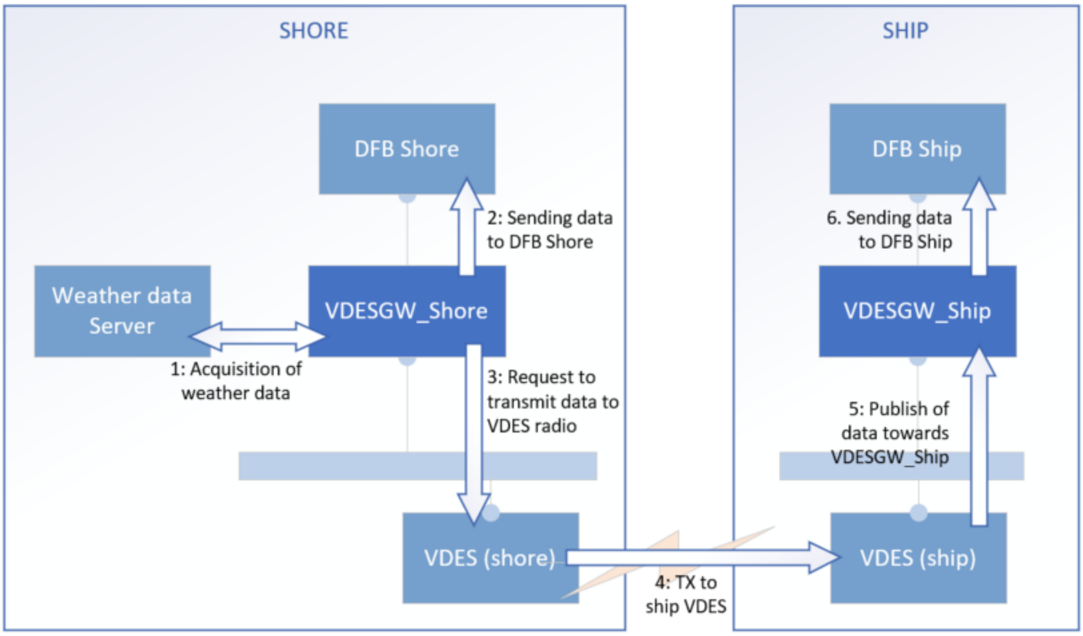


Figure 26 Weather info use case

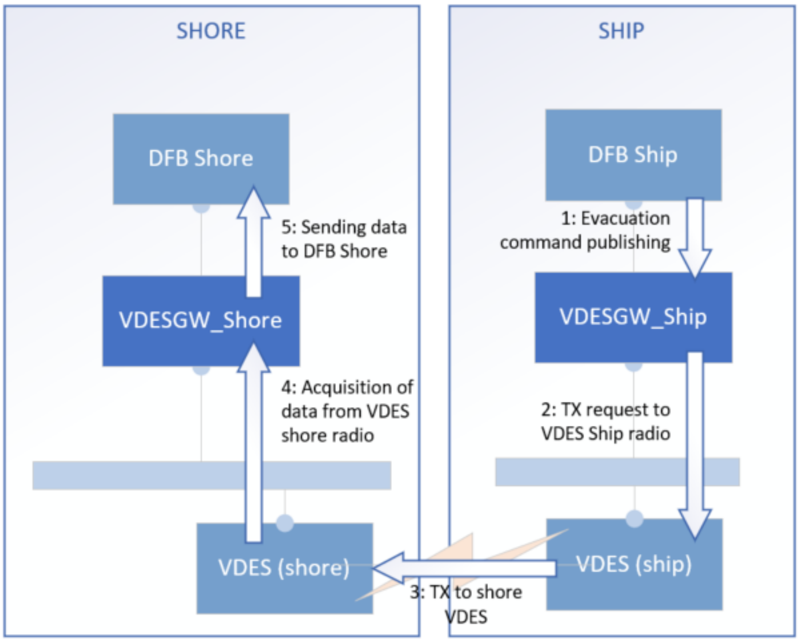


Figure 27 Evacuation info use case

Additionally, the VDES GW application shall manage the diagnostic communication with the DFB systems. This includes component query at startup and heartbeat polling during normal operating conditions:

- The VDES GW application shall answer the initial query performed by DFB system and shall identify themselves as the VDES GW component
- The VDES GW application shall answer the the periodic “heartbeat” request performed by DFB system to confirm that the communication is correctly working

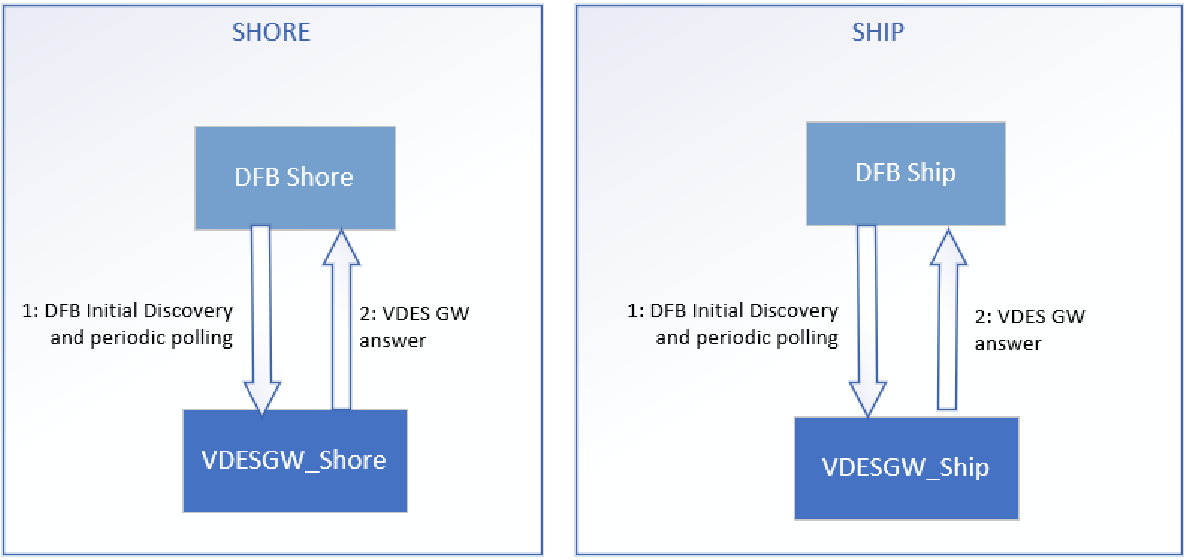


Figure 28 Discovery and heartbeat messages exchanged between DFB and VDES GW applications

8.2 Interfaces design

The top-level architecture and use cases outlined in previous chapters highlight the presence of three main interfaces that the VDESGW applications (Shore and Ship) shall manage:

- Interfaces with the ICT DFB systems. Named "1a" (shore side) and "1b" (ship side) in the next figure
- Interfaces with the VDES radios. Named "2a" (shore side) and "2b" (ship side)
- Interface with the Danaos Weather Data service, named "3".

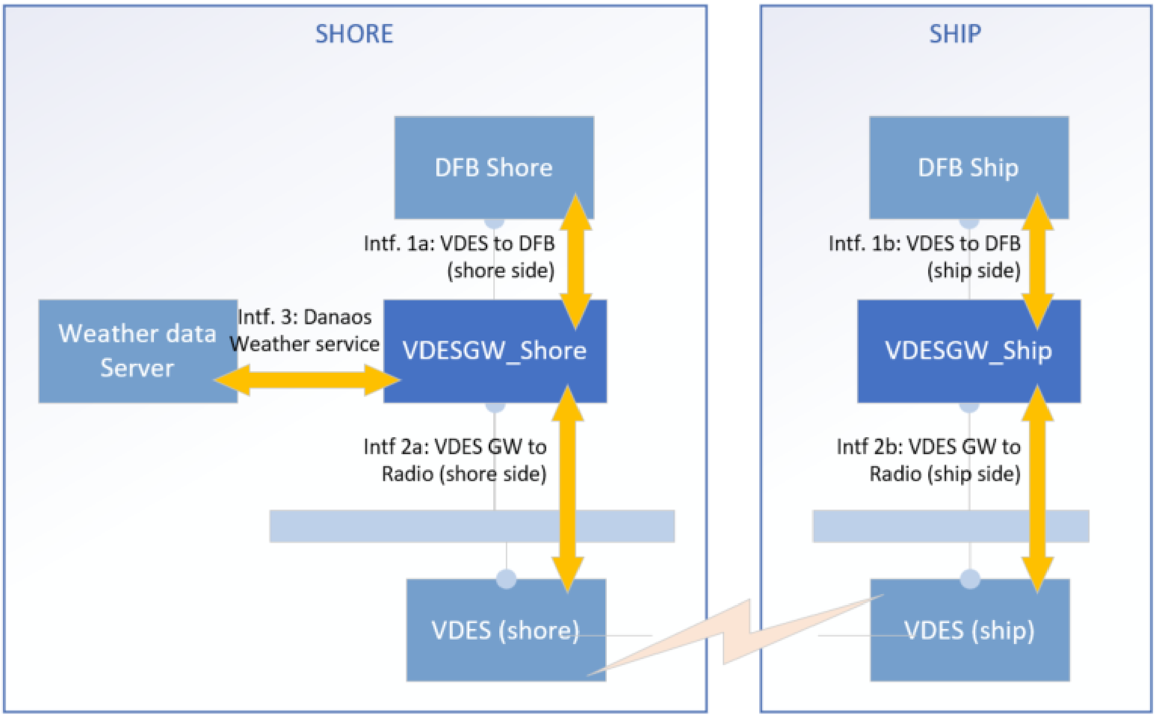


Figure 29 Interfaces of the VDESGW applications, preliminary

8.2.1 Interface with the ICT DFB systems / PALAEMON Data Fusion Bus

The ICT DFB systems stream and receive the data via the Kafka [7] messaging system technology.

Apache Kafka is an open-source project for a distributed publish-subscribe messaging system rethought as a distributed commit log. Kafka stores messages in topics that are partitioned and replicated across multiple brokers in a cluster. Producers send messages to topics from which consumers read. Kafka can monitor operational data, aggregating statistics from distributed applications to produce centralized data feeds. It also works well for log aggregation, with low latency and convenient support for multiple data sources.

On both Shore and Ship side the Kafka clusters are provided by ITML. To access the Kafka clusters the SSL security mode can be used, with certificates provided by ITML. An example of a configuration using the `librdkafka` library is the following:

```
# Kafka
bootstrap.servers=<KAFKA BROKER endpoint (Shore)>
security.protocol=ssl
ssl.key.location=<path_to_certs>/signed.key
ssl.ca.location=<path_to_certs>/trusted_authority.cert
ssl.certificate.location=<path_to_certs>/signed.pem
group.id=thales-consumer-group
```

The VDESGW_Shore application will spawn a Kafka client what will connect to the ITML Kafka cluster to exchange data with the ICT DFB Shore system.

The VDESGW_Ship application will spawn a Kafka client what will connect to the Kafka ship cluster to exchange data with the ICT DFB Ship system.

It is assumed that, apart from the cluster configuration and topics carried, the interfaces on shore and ship side are equivalent, so there is no need to distinguish them. In the following of this document, we refer to this interface as **"Intf. 1 – VDES/DFB"**.

8.2.2 Interface with the VDES Radio

The VDES radio sends and receives data using a publish/subscribe mechanism based on MQTT.

MQTT is a lightweight, publish-subscribe network protocol that transports messages between devices. The protocol usually runs over TCP/IP, however, any network protocol that provides ordered, lossless, bi-directional connections can support MQTT. It is designed for connections with remote locations where resource constraints exist, or the network bandwidth is limited. The protocol is an open OASIS standard and an ISO recommendation (ISO/IEC 20922).

In both shore and ship side an MQTT broker will be available. Both VDES radio and Gateway applications will instance an MQTT client and exchange topics and data with the mediation of the MQTT broker.

It is assumed that apart from the configuration and data carried, the interfaces on shore and ship side are equivalent, so there is no need to distinguish them. In the following of this document we refer to this interface as **"Intf. 2 – VDES GW/RADIO"**.

8.2.3 Interface with DANAOS' Weather Service API

The Danaos weather data service is a weather data proxy that can be polled to obtain current or historic maritime weather condition over the globe accessing National Oceanic and Atmospheric (NOAA) data[4].

The weather information are published using a REST interface and can be accessed at the URL http://<API_ENDP>/weatherservice and a GET request to get the weather condition of a specific point of the globe at a given data have the following format:

http://<API_ENDP>/weatherservice/<timestamp>/<lat>/<lon>

Where:

- `<API_ENDP>`: the endpoint of the DANAOS weather service server
- `<timestamp>`: timestamp in the format `YYYY-MM-DD%20HH:MM:SS`. Example of a valid timestamp: `2021-04-30%2014:10:59`
- `<lat>`: the latitude of the point, in the Decimal Degree[5] format (the server accepts latitude floating point values in the range [-86.5,86.5])
- `<long>:` the longitude of the point in the Decimal Degree format (the server accepts any floating point value)

Example of a valid URL:

https://<API_ENDPOINT>/weatherservice/20220408 13:37:59 /43.806/11.180

In the following of this document we refer to this interface as **"Intf. 3 – VDES/DWS"**.

The format of the packet generated by the Danaos Weather REST service is detailed in chapter 8.3.7.1.

[4] https://www.noaa.gov/weather
[5] https://en.wikipedia.org/wiki/Decimal_degrees

8.2.4 Interfaces final view

Summing up, the following figure shows the interfaces that shall be managed by the VDESGW applications on both Shore and Ship side, adding also the middleware used for the communication (Kafka, MQTT).

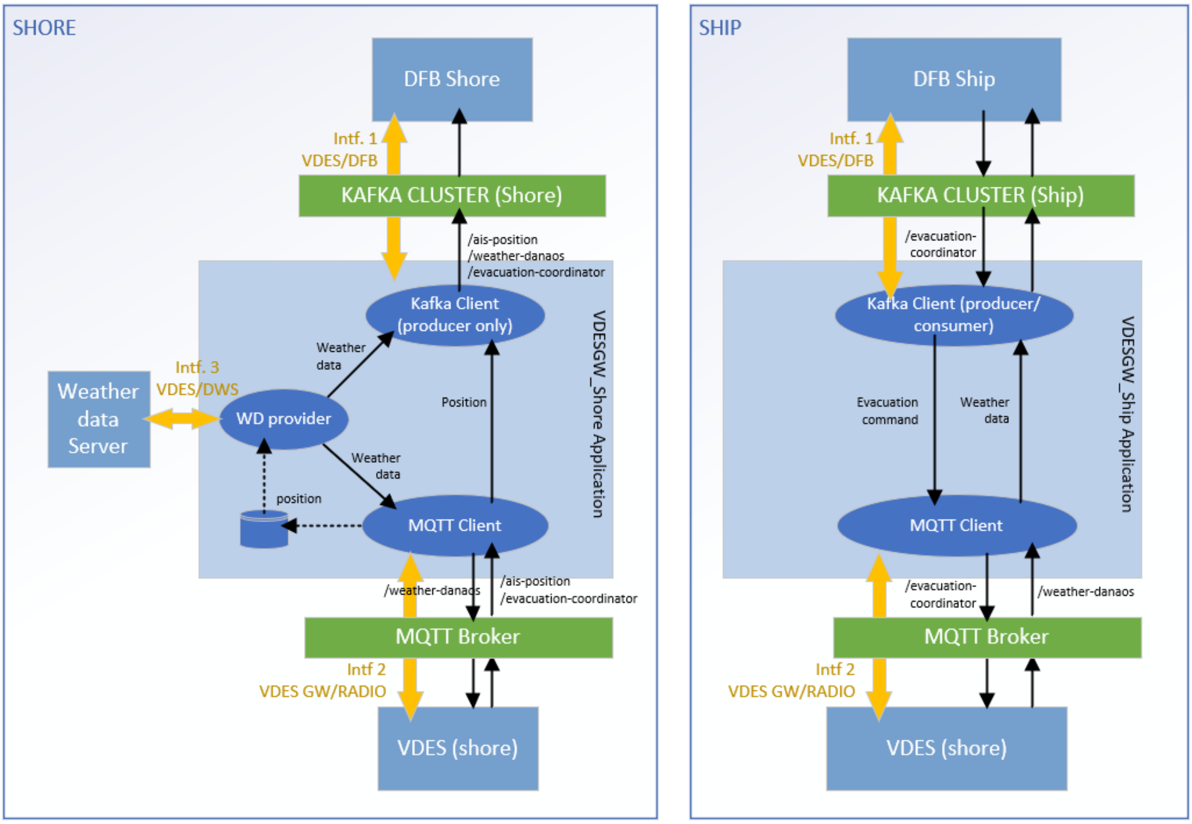


Figure 30 Detailed interfaces and architectural sketch of the VDESGW applications

8.3 Use Cases Analysis

In this section, the use case listed in the introduction are analyzed in detail. Then, a final chapter with the detailed definition of topic content is present.

8.3.1 Use Case 1: AIS position extraction

This scenario is equivalent on Ship and Shore Side. Only the Shore side is described in details.

- **Step 1**: The VDES Shore radio interfaces the AIS system and extract the ship position data. (Details outside this document scope).
- **Step 2a**: The VDES Shore radio publish the received data to the MQTT broker using the `/ais-position` topic.
- **Step 2b**: The VDESGW_Shore application receives the `/ais-position` topic from the MQTT broker.
- **Step 3**: The VDESGW_Shore radio publishes the received data to the Shore Kafka cluster using the `/ais-position` topic.

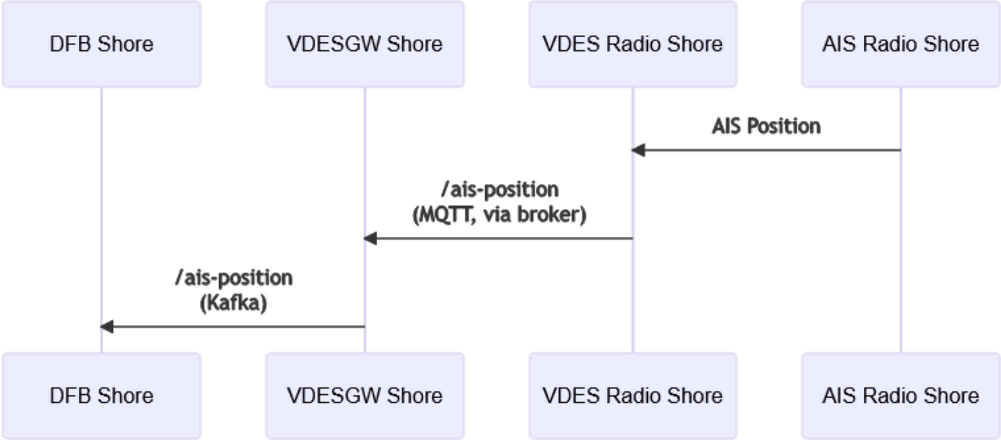


Figure 31 Sequence diagram of the AIS position use case (shore side)

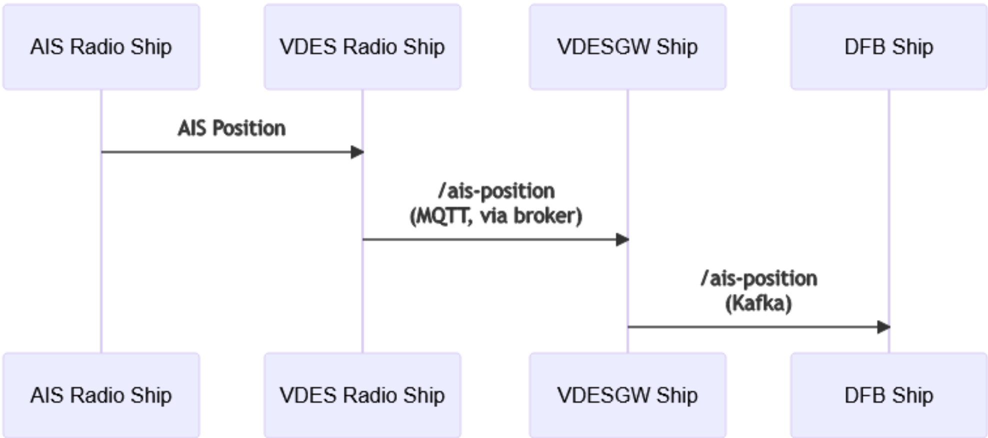


Figure 32 Sequence diagram of the AIS position use case (ship side)

8.3.2 Use Case 2: Weather Info Use Case

- **Step 1**: If the VDESGW_Shore application has a valid and fresh position of the ship, obtained from the Use Case 1, it polls periodically the Danaos Weather data service using its rest interface (see note below for consideration about polling frequency).
- **Step 2**: When new fresh data are available, the VDESGW_Shore application publish the received data to the Shore Kafka cluster using the `/weather-service` topic.
- **Step 3a**: When new fresh data are available, the VDESGW_Shore application publish the received data to the MQTT broker using the `/weather-service` topic.
- **Step 3b**: The VDES Shore radio receives the `/weather-service` topic from the MQTT broker.
- **Step 4**: The VDES Shore radio transmit the data to the VDES Ship radio using the VDES radio interface. (Details outside this document scope).
- **Step 5a**: The VDES Ship radio publishes the received data to the MQTT broker using the `/weather-service` topic.
- **Step 5b**: The VDESGW_Ship application receives the `/weather-service` topic from the MQTT broker.

- **Step 6**: When new fresh data are available, the VDESGW_Ship application publishes the received data to the Shore Kafka cluster using the `/weather-service` topic.

The diagram is split in two images to improve readability.

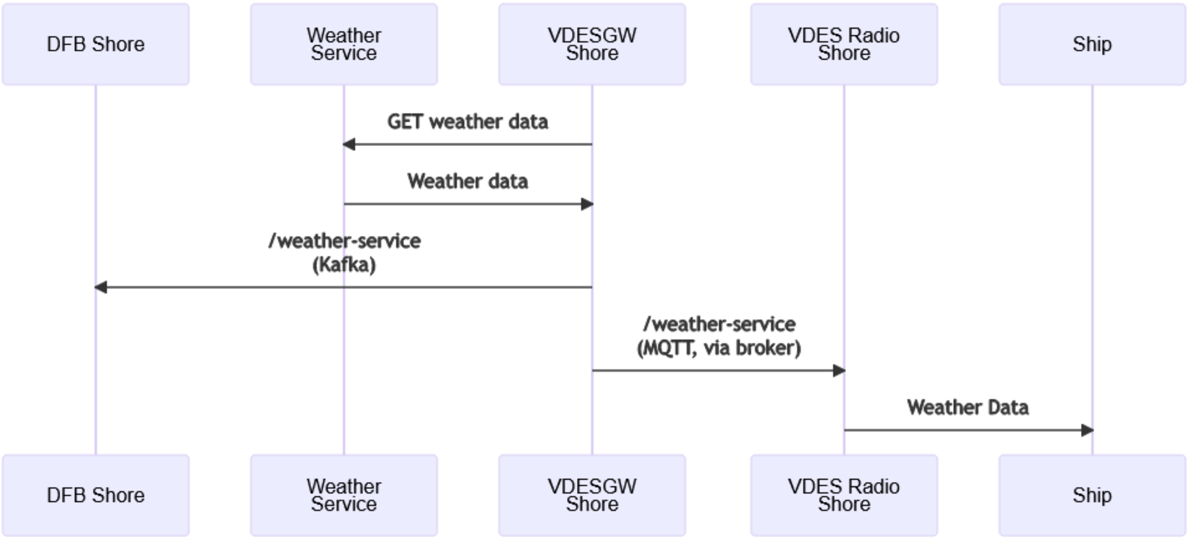


Figure 33 Sequence diagram of the Weather Info use case (shore side)

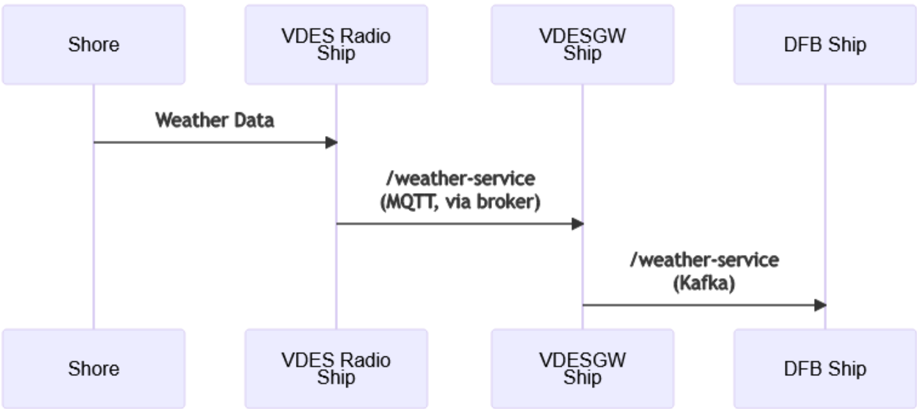


Figure 34 Sequence diagram of the Weather Info use case (ship side)

NOTE about the Weather data polling.

The Danaos Weather Data REST service, provides updates on weather data conditions every 3 hours (see [6] D3.6 Development of PALAEMON Weather Forecast Tool (v2)). The time at which the last weather condition was collected, is available in the `Timestamp` field of the REST packet generated by the Danaos weather service (see 8.3.7.1). Using this field, the VDESGW_Shore application shall try to execute as few requests as possible, booking the next query only when new data is likely to be available.

Example.

- At start-up (e.g., 12:42:00 of 1/4/2022) the VDESGW_Shore application requires the weather data at the Danaos server.
- Danaos server answers with a packet containing a timestamp of e.g., 12:00:00 o'clock of 1/4/2022. The VDESGW_Shore application can propagate these data.
- The next fresh data from Danaos server will be likely available at 15:00 (12:00 + 3 hours). Until this time, it is useless to poll the Danaos server, so the VDESGW_Shore application does nothing (in terms of weather service interface).
- At 15:00 o'clock the VDESGW_Shore application triggers a new GET request to the server. If new data are available, the application propagate the data, otherwise it will retry in (e.g.) a minute until new data will be provided.
- After new data are received the new Timestamp received is used to synchronize the next poll 3 hours later.

Another reason to perform a weather data polling is a change in position of the ship. To avoid useless polling due to small ship movements a large threshold, (e.g. 10 Km) shall be used by VDESGW_Shore application to decide to update the data due to ship position change.

8.3.3 Use Case 3: Evacuation Use Case

- **Step 1**: When a change in evacuation status is generated onboard, the ICT DFB Ship publishes an `/evacuation-coordinator` topic to the Ship Kafka cluster. This topic is consumed by the VDESGW_Ship application.
- **Step 2a**: When new fresh data are available, the VDESGW_Ship application publishes the received data to the MQTT broker using the `/evacuation-coordinator` topic.
- **Step 2b**: The VDES Ship radio receives the `/evacuation-coordinator` topic from the MQTT broker.
- **Step 3**: The VDES Ship radio transmits the data to the VDES Shore radio using the VDES radio interface. (Details outside this document scope). This radio transmission will be made so that surrounding vessels and ports can overhear the transmission and react in consequence.
- **Step 4a**: The VDES Shore radio publishes the received data to the MQTT broker using the `/evacuation-coordinator` topic.
- **Step 4b**: The VDESGW_Shore application receives the `/evacuation-coordinator` topic from the MQTT broker. The transmission of the `/evacuation-coordinator` topic from the ship to the shore is done only for demonstration purposes. The data are not forwarded to the ICT shore.
- **Step 5a:** If the evacuation status contained in the evacuation coordinator message indicates that an evacuation is in progress, the VDESGW_Ship starts to periodically send a "mayday message" to the VDES Ship radio
- **Step 5b:** The VDES Ship radio sends the data to the VDES Shore radio
- **Step 5c**: The VDES Shore radio publishes the received data towards the VDESGW_Shore application

- **Step 5d**: the VDESGW_Shore application sends the mayday message to the ICT DFB Shore application
- **Step 6:** If the evacuation status contained in the evacuation coordinator message indicates that an evacuation is **NOT** in progress, the VDESGW_Ship stops to send a “mayday message” to the VDES Ship radio. The mayday message is then no more sent to the Shore side (VDES, VDESGW, ICT)

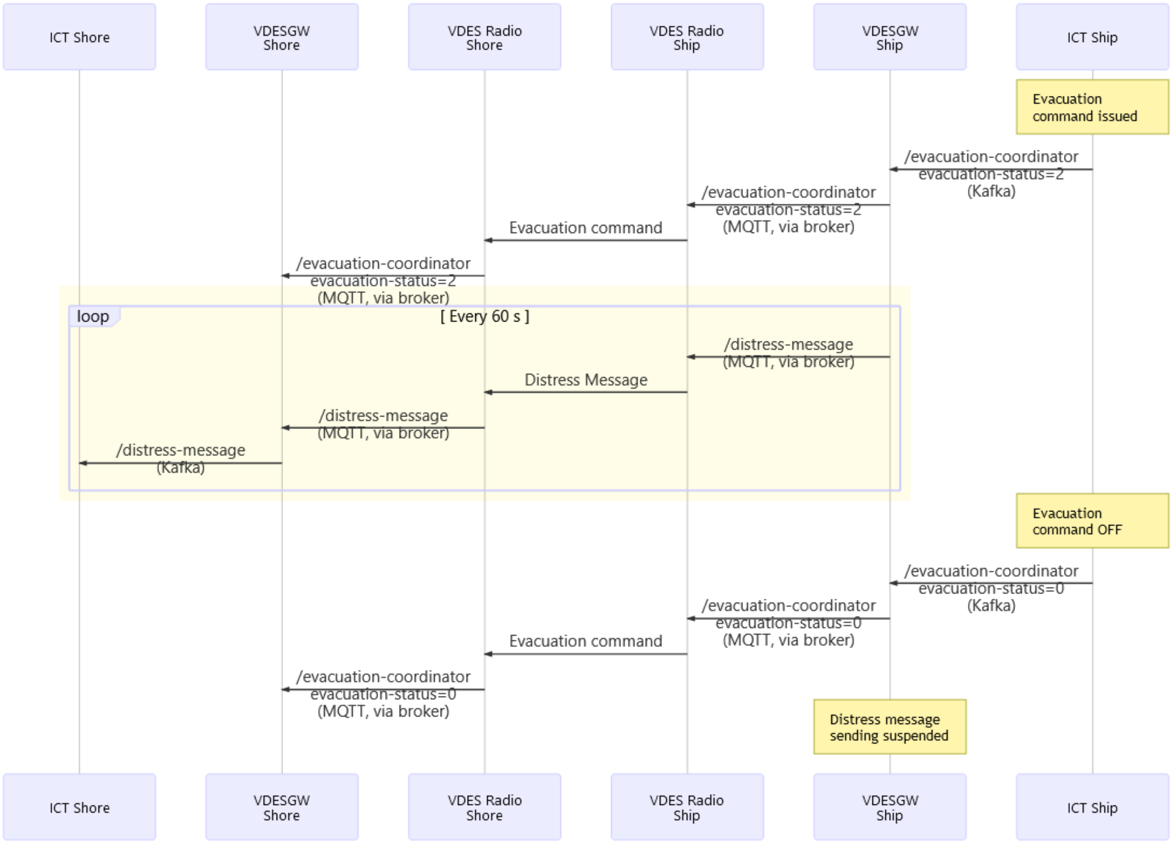


Figure 35 Sequence diagram of the Evacuation command + mayday message use case

The VDES GW applications uses the information present in the `/evacuation-coordinator` topic to update its internal operational mode used in identification and heartbeat messages (see below).

8.3.4 DFB identification and heartbeat messages

As stated above the Shore and Ship DFB implements identification and heartbeat polling messages to identify software components running on the system and to monitor the correct functioning of the communication interfaces and of the software modules.

8.3.5 Resource Discovery

The DFB collect the list of the devices and software component running in the system using a “resource discovery” message:

- The DFB (Shore and Ship) generates `/resource-discovery-request` topic to collect the complete list of devices operative in the system

- The VDES GW (Shore and Ship) shall reply to a `/resource-discovery-request` topic with a `/resource-discovery-response` topic

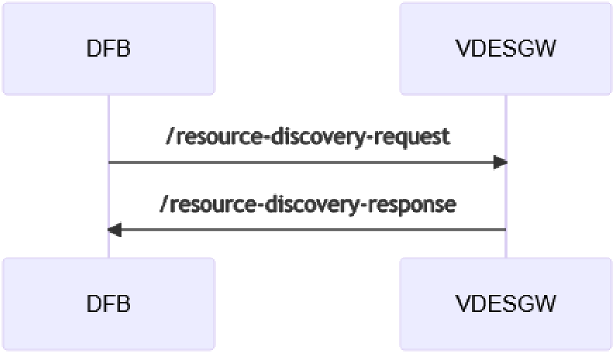


Figure 36 Resource Discovery performed by DFB applications

8.3.6 Resource Polling

The DFB periodically performs a polling of all the collected devices and applications using a “heartbeat” message:

- The DFB (Shore and Ship) generates periodically `/heartbeat-request` topic towards the VDES GW application (Shore and Ship respectively) to monitor the status of the VDES GW application and the correct working of the communication.
- The VDES GW (Shore and Ship) shall reply to an `/heartbeat-request` topic with an `/heartbeat-response` topic

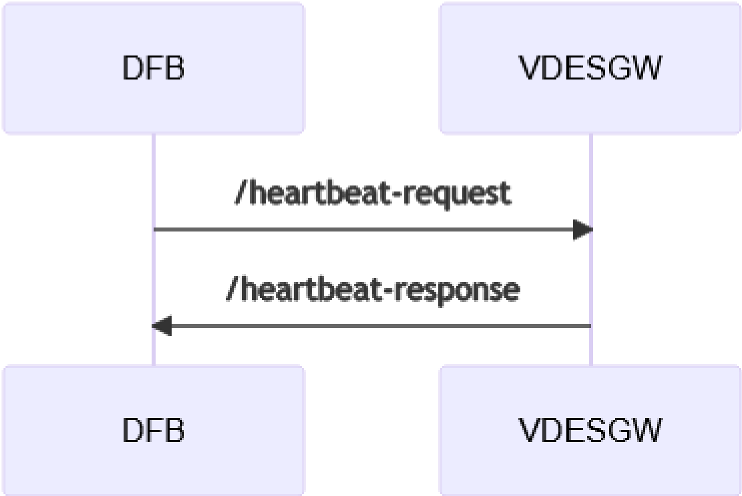


Figure 37 Heartbeat polling performed by DFB applications

8.3.7 Message content definition

The following topic/messages have been identified:

- `/ais-position` topic: contains the information about position of the ship as collected by the VDES Ship radio
- `/weather-danaos` topic: the weather condition at the position of the ship at a given time (e.g. the current time).
- `/evacuation-coordinator` topic: the topic carrying an evacuation event with related information
- `/resource-discovery-request` topic: generated by the DFB applications to collect the list of devices and applications active in the system
- `/resource-discovery-response` topic: generated by the VDES GW applications as an answer to a `/resource-discovery-request` message
- `/heartbeat-request` topic: generated by the DFB applications to poll the status of devices and applications active in the system
- `/heartbeat-response` topic: topic: generated by the VDES GW applications as an answer to a `/heartbeat-request` message

Additionally, the packet generated by the Danaos Weather Service is described here.

8.3.7.1 Danaos weather data packet

Table 13 The format of the packets generated by the Danaos Weather REST service

#	Field	Description
1	Timestamp	The timestamp to which the weather data were collected
2	combDirectionDegrees	combDirectionDegrees
3	combPeriod	combPeriod
4	combSWHMeters	combSWHMeters
5	currentsDirectionDegrees	The currents direction of the current (0-359, 0=N, 90=E, etc.)
6	currentsSpeedKnots	The currents speed in knots
7	hum%	The percentage of humidity
8	iceCover	The ice coverage (from 0.0 = no ice to 1.0 = full ice coverage)
9	lat	The latitude of the geographical position to which the weather data refers. In decimal degree format.
10	lon	The longitude of the geographical position to which the weather data refers. In decimal degree format.
11	mslhPa	mslhPa
12	sea	Boolean, `true` if position is in sea, `false` otherwise
13	swellDirectionMeters	The swell direction

#	Field	Description
14	swellPeriod	The swell period (unit of measure unknown)
15	swellSWHMeters	Significant swell height in meters
16	tempCelciousDegrees	The temperature in Celsius degrees
17	visKm	visKm
18	wavesDirectionDegrees	The waves direction (0-359, 0=N, 90=E, etc.)
19	wavesPeriod	The waves period in seconds
20	wavesSWHMeters	Significant waves height in meters
21	windDirectionDegrees	The wind direction (0-359, 0=N, 90=E, etc.)
22	windSpeedKnots	The wind speed in knots

Example:


```
{
    "Timestamp": "2020-01-01 12:00:00",
    "combDirectionDegrees": 24.98,
    "combPeriod": 2.35,
    "combSWHMeters": 0.28,
    "currentsDirectionDegrees": 45.0,
    "currentsSpeedKnots": 0.05832,
    "hum%": 66.3,
    "iceCover": 0.0,
    "lat": 44.123,
    "lon": -10.456,
    "mslhPa": 1032.54,
    "sea": true,
    "swellDirectionMeters": 68.44,
    "swellPeriod": 2.85,
    "swellSWHMeters": 0.14,
    "tempCelciousDegrees": 11.18,
    "visKm": 24.135,
    "wavesDirectionDegrees": 15.3,
    "wavesPeriod": 2.19,
    "wavesSWHMeters": 0.24,
    "windDirectionDegrees": 339.98,
    "windSpeedKnots": 9.93384
}
```


8.3.7.2 `/ais-position` topic

Table 14 /ais-position topic data content

#	Field	Description
1	user_id	Unique identifier such as MMSI number
2	nav_status	Navigational Status. See table below for possible values
3	rate_turn_rotais	Rate of turn ROT_{ais} as taken from the AIS data
4	sog_available	Boolean, **true** if Speed Over Ground (SOG) is available, **false** otherwise
5	sog	Speed over ground in knots. This field is set to 0 and shall be ignored if sog_available is **false**
6	pos_accuracy	The position accuracy (PA) flag: 1 = high (≤ 10 m) 0 = low (>10 m, default)
7	longitude	Longitude in decimal degrees format[6] and range (-180 to 180)
8	latitude	Latitude in decimal degrees format and range (-90 to 90)
9	cog_available	Boolean, **true** if Course Over Ground (COG) is available, **false** otherwise
10	cog	Course Over Ground, if available. Possible values from 0.0 to 359.0. This field is set to 0 and shall be ignored if cog_available is **false**
11	th_available	Boolean, **true** if True Heading is available, **false** otherwise
12	true_heading	True Heading value. Possible values from 0 to 359. This field is set to 0 and shall be ignored if th_available is **false**
13	timestamp	UTC Timestamp when the report was received from the VDES Ship radio. Format: 'YYYY-MM-DDTHH:MM:SS.mmm'
14	manoeuvre_ind	Special Maneuver Indicator. See table below for possible values

*Table 15 Possible values for the **nav_status** field of the **/ais-position** topic*

Value	Meaning
0	Under way using engine
1	At anchor
2	Not under command
3	Restricted manoeuvrability
4	Constrained by her draught
5	Moored
6	Aground
7	Engaged in fishing

[6] https://en.wikipedia.org/wiki/Decimal_degrees

8	Under way sailing
9	Reserved
10	Reserved
11	Power-driven vessel towing astern
12	Power-driven vessel pushing
13	Reserved
14	AIS-SART (active), MOB-AIS, EPIRB-AIS
15	Undefined (default)

Table 16 Possible values for the ***`manoeuvre_ind`*** *field of the* ***`/ais-position`*** *topic*

Value	Meaning
0	Not available (default)
1	Not engaged in special manoeuvre
2	Engaged in special manoeuvre

Example:

```
{
    "user_id": "1234",
    "nav_status": 1,
    "rate_turn_rotais": -126,
    "sog_available": true,
    "sog": 0.0,
    "pos_accuracy": 1,
    "longitude": 44.10444384277289,
    "latitude": 9.829416212827747,
    "cog_available": true,
    "cog": 120,
    "th_available": true,
    "true_heading":123,
    "timestamp": "2022-03-22T15:45:59.123",
    "manoeuvre_ind": 1
}
```

8.3.7.3 `/weather-danaos` topic

The weather Danaos topic contains the same information collected from the Danaos weather REST service. The names of some fields are changed for uniformity with the other topics.

Table 17 `/weather-danaos` topic data content

#	Field	Description
1	Timestamp	The UTC timestamp to which the weather data were collected. Format: 'YYYY-MM-DDTHH:MM:SS.mmm'
2	combDirectionDegrees	?
3	combPeriod	?
4	combSWHMeters	?
5	currentsDirectionDegrees	The currents direction of the current (0-359, 0=N, 90=E, etc.)
6	currentsSpeedKnots	The currents speed in knots
7	hum%	The percentage of humidity
8	iceCover	The ice coverage (from 0.0 = no ice to 1.0 = full ice coverage)
8	latitude	The latitude of the geographical position to which the weather data refers. In decimal degree format.
8	longitude	The longitude of the geographical position to which the weather data refers. In decimal degree format.
9	mslhPa	?
10	sea	Boolean, `true` if position is in sea, `false` otherwise
11	swellDirectionMeters	The swell direction
12	swellPeriod	The swell period (unit of measure unknown)
13	swellSWHMeters	Significant swell height in meters
14	tempCelciousDegrees	The temperature in Celsius degrees
15	visKm	?
16	wavesDirectionDegrees	The waves direction
17	wavesPeriod	The waves period
18	wavesSWHMeters	Significant waves height in meters
19	windDirectionDegrees	The wind direction (0-359, 0=N, 90=E, etc.)
20	windSpeedKnots	The wind speed in knots

Example:


```
{
    "timestamp": "2020-01-01 12:00:00",
    "comb_direction_degrees": 24.98,
    "comb_period": 2.35,
    "comb_swh_meters": 0.28,
    "currents_direction_degrees": 45.0,
```

```
    "currents_speed_knots": 0.05832,
    "humidity_percentage": 66.3,
    "ice_cover": 0.0,
    "longitude": 44.10444384277289,
    "latitude": 9.829416212827747,
    "mslh_pa": 1032.54,
    "sea": true,
    "swell_direction_meters": 68.44,
    "swell_period": 2.85,
    "swell_swh_meters": 0.14,
    "temp_celsius_degrees": 11.18,
    "vis_km": 24.135,
    "waves_direction_degrees": 15.3,
    "waves_period": 2.19,
    "waves_swh_meters": 0.24,
    "wind_direction_degrees": 339.98,
    "wind_speed_knots": 9.93384
}
```


8.3.7.4 `/evacuation-coordinator` topic

For the sake of illustration, Figure 38 represent the evolution of states during an evacuation process. Moreover, the picture also reflects the actors that trigger the status switch. The reader might refer to D2.6 and D2.7 for a more in-depth description.

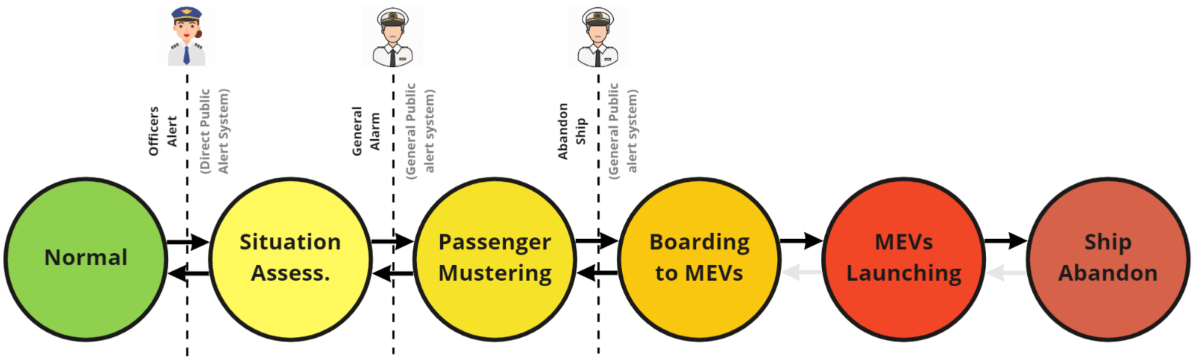


Figure 38. Evacuation status chain

Table 18 `/evacuation-coordinator` topic data content

#	Field	Description
1	originator	The originator of the command (e.g. “evacuation-coordinator”)
2	evacuation-status	Integer, the status of the evacuation, see table below for the possible values
3	timestamp	The timestamp of the command. Format: 'YYYY-MM-DDTHH:MM:SS.mmm'

Table 19 Possible values for the ***`evacuation_status`*** *field of the* ***`/evacuation-coordinator`*** *topic*

Value	Meaning
0	Normal status (no evacuation)
1	Situation assessment
2	Passenger Mustering
3	Boarding to MEVs
4	MEVs launching
5	Ship abandon

Example:

```
{
    "originator": "evacuation-coordinator",
    "evacuation_status": 1,
    "timestamp": "2022-03-21T23:14:59.393523"
}
```

8.3.7.5 `/mayday-message` topic

The mayday message contains information about the ship and surrounding environmental condition when an evacuation command has been generates

Table 20 ***`/mayday-message`*** *topic data content*

#	Field	Description
1	via	The communication channel used to send the message (VDES)
2	ship_name	The name of the ship
3	ship_position	Position of the ship (latitude, longitude)
4	timestamp	The timestamp of the evacuation command event
5	incident_type	The incident type
6	assistance_required	The type of assistance that it is required (if any, "none" is the default value)
7	number_passenger	The number of the passenger registered onboard
8	injured_people	The number of injured people
9	crew_on_board	The number or crew person registered onboard
10	weather_condition	A description of the weather condition at the ship position
11	damage_extent	The extension of the damage in percent of the total ship extension

Example:

```
{
```

```
    "via" : "VDES",
    "ship_name" : "Costa Concordia",
    "ship_position" : [
        42.3717,
        10.92602
    ],
    "timestamp" : "2021-01-26T14:25:00.606610+00:00",
    "incident_type" : "Contact - Breach - Blackout",
    "assistance_required" : "None",
    "number_passengers" : 4229,
    "injured_people" : 0,
    "crew_on_board" : 1023,
    "weather_conditions" : "ROUGH SEA - NE 4; WIND 7 KNOTS E; VISIBILITY CLOUDY",
    "damage_extent" : "75%"
}
```


8.3.7.6 `/resource-discovery-request` topic

This message has the same format of the `/evacuation-coordinator` topic

8.3.7.7 `/resource-discovery-response` topic

Table 21 `/resource-discovery-response` topic data content

#	Field	Description
1	timestamp	The timestamp of the command. Format: 'YYYY-MM-DDTHH:MM:SS.mmm'
2	component_id	The id of the component. For the VDES GW application (both Shore and Ship side), this field shall be sent to "**vdes-gw**"
3	operation_mode	The operation mode of the component. This field shall reflect the evacuation status received in the `/evacuation-coordinator`, `/resource-discovery-request` and `/heartbeat-request` topics. If the evacuation status is set to 0 (Normal Status) the operation_mode shall be set to 0, if the evacuation status is different from 0, the operation_mode shall be set to 1.

Example:


```
{
    "timestamp": "2022-03-21T23:14:59.393523"
    "component_id": "vdes-gw",
```

```
    “operation_mode”: 1,
}
```

8.3.7.8 `/heartbeat-request` topic

This message has the same format of the `/evacuation-coordinator` topic

8.3.7.9 `/heartbeat-response` topic

This message has the same format of the `/resource-discovery-response` topic

9 References

[1] IALA, *Guideline G1139: The technical specification of VDES*, Edition 3, June 2019.

[2] J. Safar *et al.*, “VDES Channel Sounding Campaign,” Trial report, April 2014.

[3] M. Luise and R. Reggiannini, “Carrier frequency recovery in all-digital modems for burst-mode transmissions,” IEEE Transactions on Communications 43 (2/3/4), pp. 1169-1178, 1995.

[4] A. J. Viterbi and A. M. Viterbi, “Nonlinear Estimation of PSK-Modulated Carrier Phase with Application to Burst Digital Transmission,” IEEE Transaction on Information theory, vol. 29, n. 4, July 1983.

[5] AAVV. (2021). D 3.6 Development of PALAEMOM Weather Forecast Tool (v2).

[6] Bejeck, B. (2018). Kafka Streams in Action: Real-time apps and microservices with the Kafka Streams API. In *Simon and Schuster.*

[7] ITU-R. (2014). *Reccomendation ITU-R M.1371-5 Technical characteristics for an automatic identification system using time division multiple access in the VHF maritime mobile frequency band*

ANNEX A – VDES Demonstration Test Procedure

The DEMO of the VDES (VHF Data Exchange System) took place on October 28th at Thales headquarters in Sesto Fiorentino. Hereafter the Test Procedure followed during the DEMO is reported.

All tests have been performed and the results have been as per related expectations.

1 Setup and Material

In order to perform the test described in this section, it is required the following material:

- 3 testbed nodes (named node-1, node-2 and node-3) containing each:
  - ZC706 evaluation board
  - ADRV9002 RF board
  - RF splitter
  - Power Delivery
- 3 Linux notebook (named pc-1, pc-2 and pc-3)
- 3 VGA/HDMI monitor
- 5 Eth Cables
- 3 variable attenuators
- 2 Banten Whip Antennas (RX)
- 2 Protel Antennas (TX)
- 1 Banten Whip Antennas (TX)
- 1 Network switch connected to internet
- 1 Agilent N9340B Spectrum Analyzer

1.1 Setup Ship Installation

The setup of the ship installation required the execution of following steps:

- Connect VDES ship testbed (node-2) to the network switch via LAN cable;
- Connect Ship PC (pc-2) to the network switch;
- Connect Protel Antenna to the Attenuator;
- Set Attenuator to -10 dB of attenuation;
- Connect the Attenuator to VDES Ship Testbed (node-2) RF output port;
- Connect Banten Whip Antenna to VDES Ship testbed (node-2) RF input port (splitter);

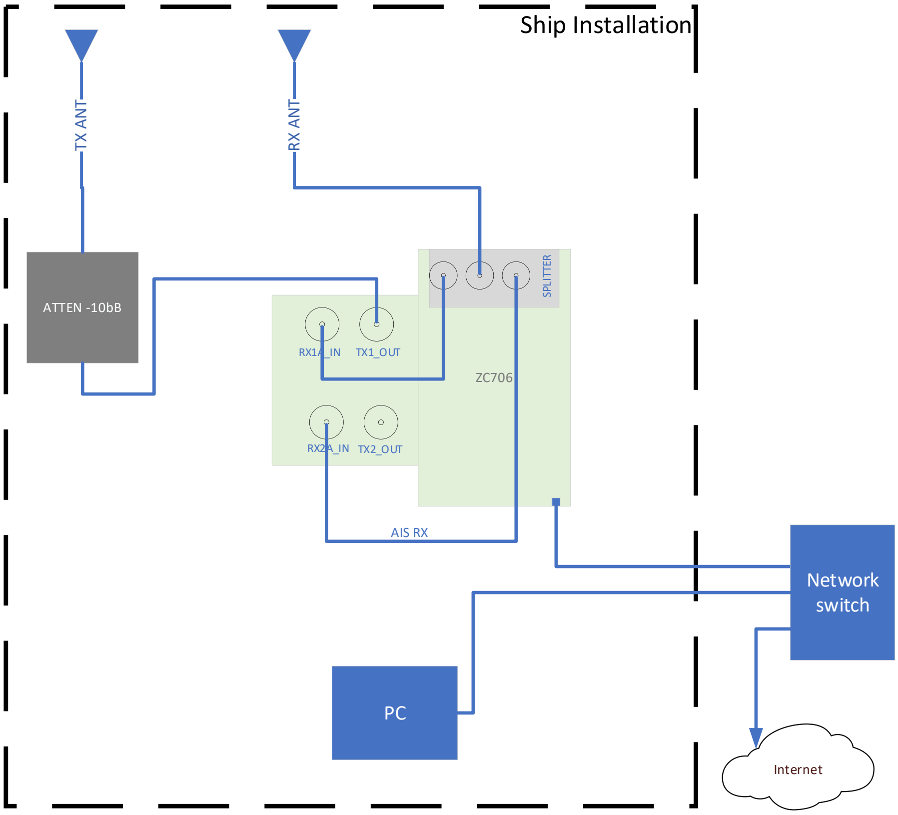


Figure 39- Ship installation

1.2 Setup Shore Installation

The setup of the shore installation required the execution of following steps:

- Connect VDES shore testbed (node-3) to the network switch via LAN cable;
- Connect Ship PC (pc-3) to the network switch;
- Connect Protel Antenna to the Attenuator;
- Set Attenuator to -10 dB of attenuation;
- Connect the Attenuator to VDES Shore Testbed (node-3) RF output port;
- Connect Banten Whip Antenna to VDES Shore testbed (node-3) RF input port (splitter);

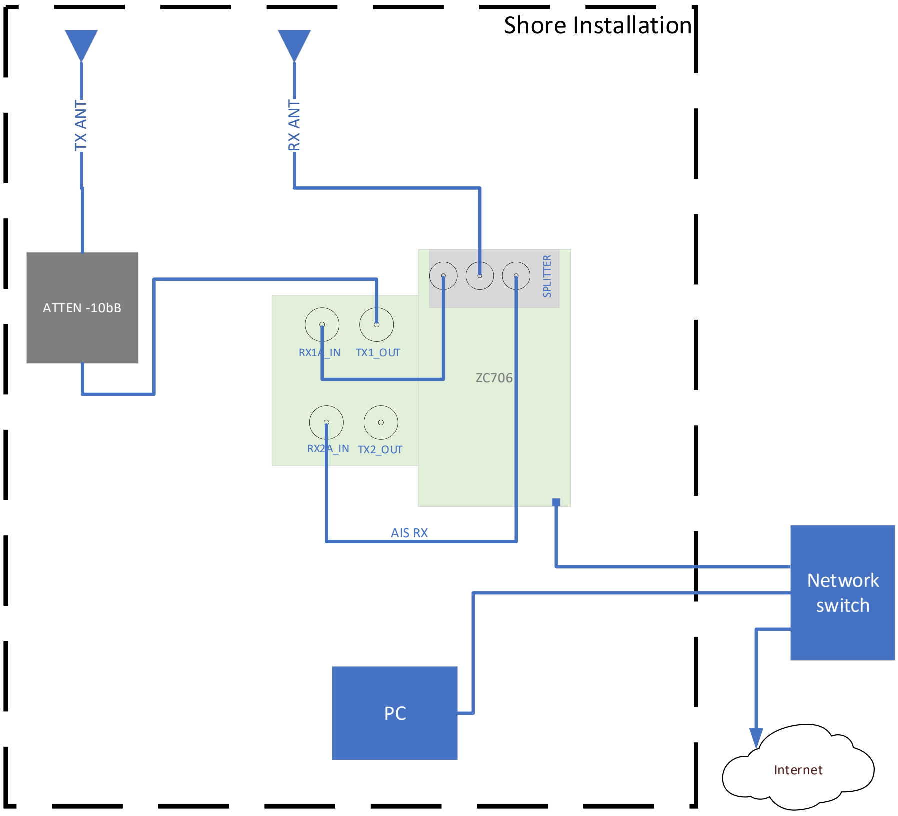


Figure 40- Shore installation

1.3 Setup AIS Emulator

The setup of the AIS Emulator installation required the execution of following steps:

- Connect VDES AIS Emulator testbed (node1) to the network switch via LAN cable;
- Connect AIS Emulator PC (pc-1) to the network switch;
- Connect Banten Whip Antenna to VDES Shore testbed (node-1) RF output port;

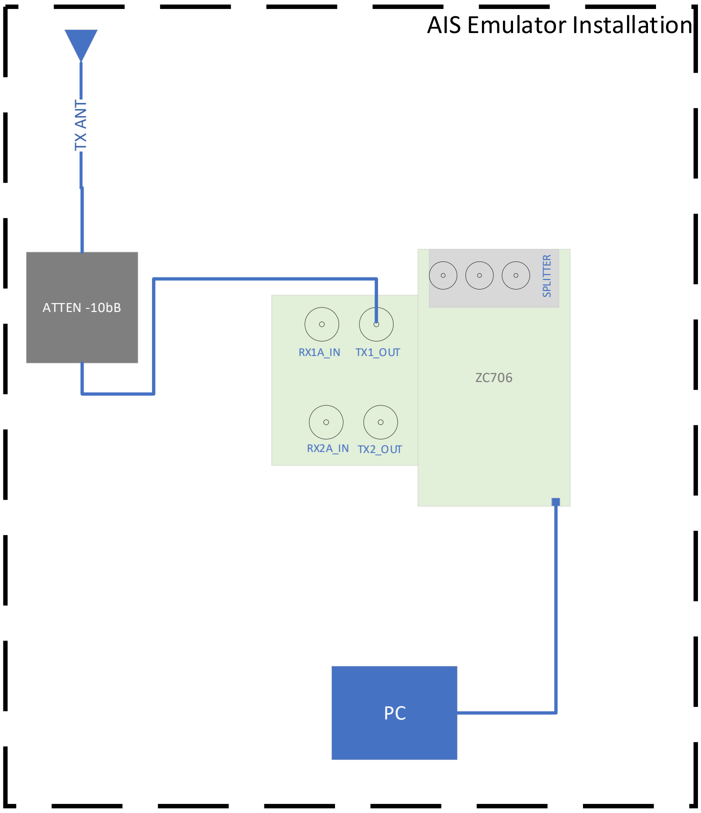


Figure 41 – AIS Emulator installation

2 Test List

This chapter includes paragraphs to describe each test.

2.1 [T01] Boxes, Antennas and Installation Accessories Visual Inspection test

2.1.1 Test Short Description

This Test consists mainly in a Visual Inspection conducted on all boxes for containment of ZC706, ADRV 900 and installation accessories for mounting and connecting the Tx and Rx antennas, aiming at verifying that all the systems are assembled according to LINE-UP Tx-Rx DiagramRev02.pdf and internally wired according to Box for ZC706 LayoutRev06.pdf and Box for LNA LayoutRev01.pdf.

2.1.2 Test Pre-conditions

All the boxes containing the cards, the amplifiers, the filters, Tx and Rx antennas and the installation accessories for the VDES Test must be correctly installed according to the LINE-UP Tx-Rx DiagramRev02.pdf. Where required, the power supplies of the boxes must be correctly connected to the local power network.

2.1.3 Test input Data

The data necessary to implement the test are represented by the presence and correct installation of all the materials listed above.

2.1.4 Test Steps Description

During the verification phase, the presence and correct installation of all the materials listed in the table must be checked reporting in the appropriate column OK if the material is present and correctly installed and NOK vice versa.

2.1.5 Test Outputs

The Test Outputs will be represented by the correctly compiled Test Material list.

2.1.6 Test Results

The test will be considered passed if all the materials on the list are present and correctly installed. In the event that any material is missing, appropriate explanations must be provided reporting a footnote to the Test Material List.

2.2 [T02] AIS Synchronization test

2.2.1 Test Short Description

This objective of AIS Synchronization test is to verify the capability of the Shore and ship VDES radio to synchronize with the AIS radio system.
The AIS signal will be generated by the 3rd node, connected to a PC running an AIS sample file.

2.2.2 Test Pre-conditions

Complete the setup of Ship Installation, Shore Installation and AIS Emulator.

2.2.3 Input Data

No input.

2.2.4 Steps description

To perform this test, it is necessary to execute the following operations in order:

1. Switch on Node-1
2. Switch on Node-2
3. Switch on Node-3
4. Switch on Spectrum Analyzer
5. On PC-1:
    a. Connect via SSH PC-1 to Node-1
6. On PC-2:
    a. Connect via SSH PC-2 to Node-2
    b. Open terminal with mosquitto-sub ais-position

   c. Open terminal with mosquitto-sub mayday
   d. Open terminal for VDES_GW (not started)
   e. Open ais map
7. On PC-3:
   a. Connect via SSH PC-2 to Noed-3
   b. Open terminal with mosquitto-sub weather
   c. Open terminal for VDES_GW (not started)
8. On PC-2 (SSH) start VDES SHORE
9. On PC-3 (SSH) start VDES SHIP
10. Wait RF calibration on both
11. Press any key on both PC-2 and PC-3
12. Wait end of noise measure
13. Start AIS emulator on PC-1
14. Wait AIS sync and training

2.2.5 Test Outputs

Shore and ship logs.

2.2.6 Test Results

The test will be considered OK if the Shore and Ship PC(s) show message of AIS synchronization on the terminals.

2.3 [T03] AIS Position Retrieve test

2.3.1 Test Short Description

This objective of AIS position test is to verify the capability of the Shore VDES radio to interface and extract information from the AIS radio system (ship position), and the subsequent delivery of those messages to the DFB on shore side.
The AIS signal will be generated by the 3rd node, connected to a PC running an AIS sample file.

2.3.2 Test Pre-conditions

Complete the [T02] AIS Synchronization test.

2.3.3 Test input Data

No input data.

2.3.4 Test Steps Description

To perform this test, it is necessary to execute the following operations in order:

1. Complete [T02] AIS Synchronization test procedure
2. Check ais map

2.3.5 Test Outputs

Reception of the /ais-position topic by the Shore PC (pc-2) and show it on map.

2.3.6 Test Results

The test will be considered OK if the Sore PC receives correctly the AIS position generated by the AIS Emulator.

2.4 [T04] Weather Data Polling and Transmission test

2.4.1 Test Short Description

The objective of this test is to verify the correct reception of weather condition messages related to the specific ship position (acquired from the previous test case) by the shore VDES system and the correct transmission of these messages to the ship via VDES radio channel. Once these messages arrive at the ship's VDES, they shall be made available to the DFB Ship.

2.4.2 Test Pre-conditions

Complete [T03] AIS Position Retrieve test procedure

Components that shall be reachable from the network:

- The Danaos Weather Forecast server shall be up and running and reachable via the internet by the Shore PC (PC-2)

This environment assumes that the last two PC(s) are available and that they contain the executables correctly built, with all the configuration and simulation files properly prepared.

2.4.3 Test input Data

No input data.

2.4.4 Test Steps Description

To perform the test the following steps must be performed:

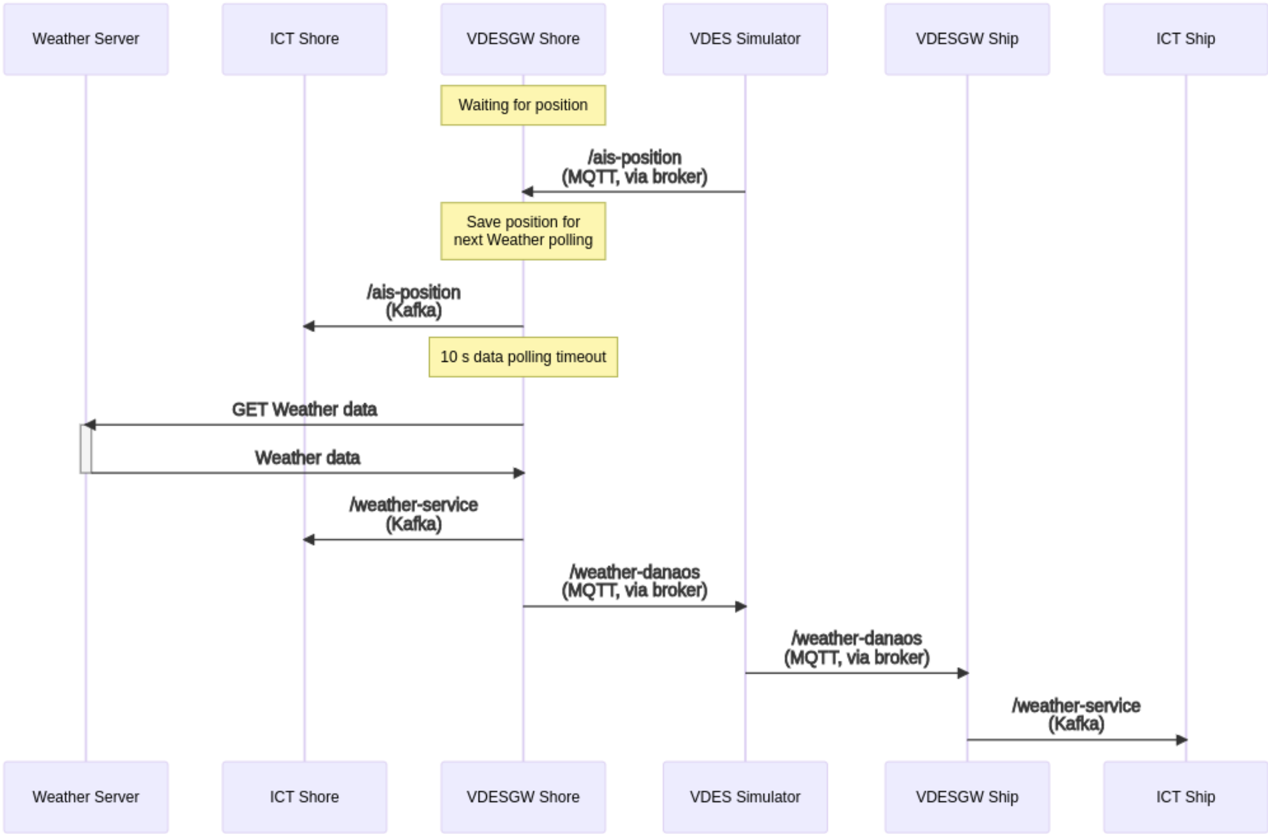
1. Complete [T03] AIS Position Retrieve test procedure
2. Switch on VDES GW Shore on PC-2
3. Check mosquitto-sub on PC-3

At the startup, the `shore` VDES-GW application does not have a valid know ship position, and for this reason it disables the polling to the Danaos Weather service and enter a special "10 second" polling to check if a valid position is received.

After this, the scenario proceeds as following:

- After the VDES has generated the first `/ais-position` topic on the shore side, the VDES-GW `shore` application saves the location to use it on the next useful "10 second" polling
- So, within a maximum of 10 second, the `shore` VDES-GW application requires the current weather data in the given ship position to the Danaos weather server

- The Danaos weather server provides the current weather condition to the **shore** VDES-GW application at the given ship position
- The **shore** VDES-GW application provides the weather data to the ICT-Shore Kafka cluster
- The **shore** VDES-GW application publishes the weather data to the MQTT shore channel using the **/weather-danaos** topic
- The VDES shore radio receives the **/weather-danaos** topic and forward the data to the ship side through the radio channel. On the **ship** side the VDES radio publishes the **/weather-danaos** topic on the MQTT ship channel
- The **ship** VDES-GW application provides the weather data to the ICT-Ship Kafka cluster



NOTE: To monitor the generation of the **/weather-danaos** topic on ship side, the following command can be entered:

```
$ mosquitto_sub -h localhost -t /weather-danaos
```

Additionally, the VDES Gateway Ship application logs shall be checked to verify that the **/weather-danaos** topic is received, and that the **/weather-service** topic is sent to the DFB Kafka cluster. On the DFB cluster side collect the evidence of the reception of the **/weather-service** topic.

2.4.5 Test Outputs

Check how many weathers condition messages have arrived on the ship.

2.4.6 Test Results

The test will be considered OK if less than 5 % of the total number of messages were lost, FAIL otherwise.

2.5 [T05] Evacuation coordinator test

2.5.1 Test Short Description

The objective of this test is to verify the exchange of messages for synchronization (resource request and allocation) between VDES ship and shore radios, the correct reception of evacuation messages by the VDES ship (received from DFB ship) and the proper transmission of these messages to the shore via VDES radio channel. If an evacuation is in progress, the ship gateway application starts to send “mayday messages”. Once these messages arrive at the shore's VDES, they shall be made available to the DFB Shore

2.5.2 Test Pre-conditions

Complete [T04] Weather Data Polling and Transmission test.

2.5.3 Test input Data

Evacuation coordinator message

2.5.4 Test Steps Description

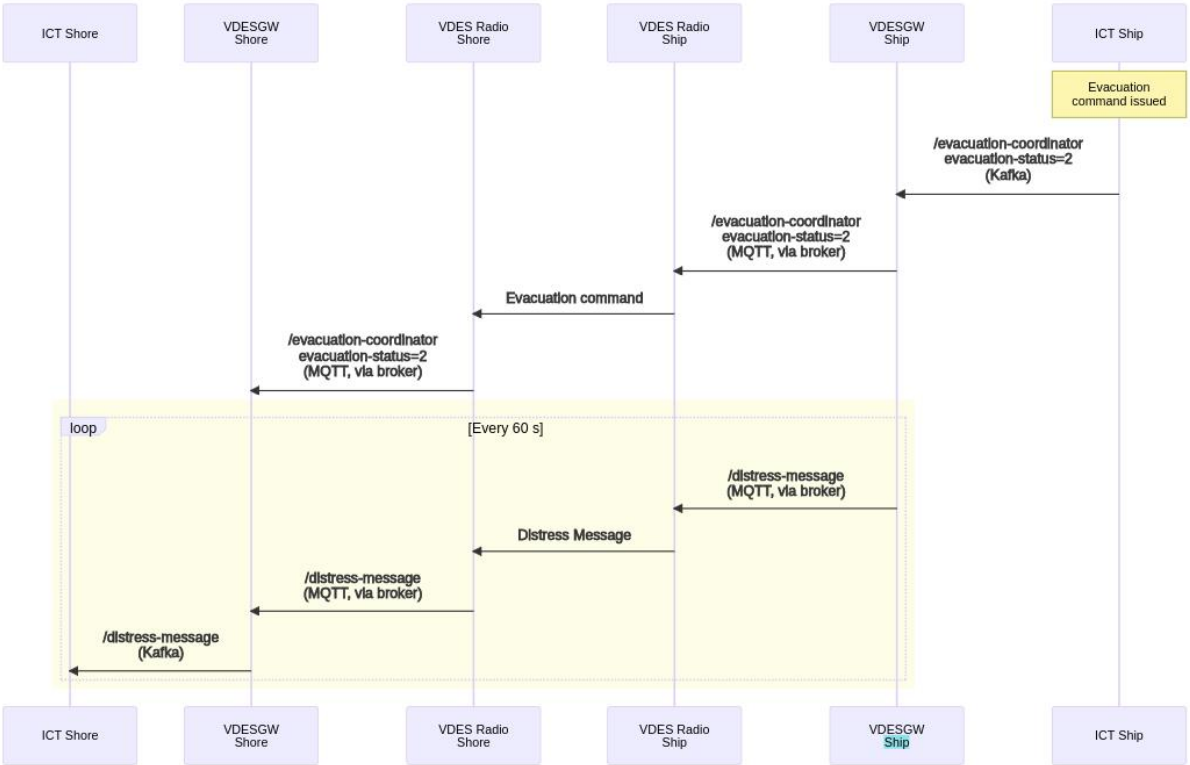
To perform the test the following steps must be performed:

1. Complete [T04] Weather Data Polling and Transmission test procedure
2. Switch on VDES GW Ship on PC-3
3. Set Evacuation status to 2 on KAFKA Cluster
4. Check results on PC-2

After this, the scenario proceeds as following:

- The **`ship`** VDES-GW application receives the **`/evacuation-coordinator`** topic on the Kafka interface
- The **`ship`** VDES-GW application forwards the **`/evacuation-coordinator`** topic to the VDES Simulator application using the homonym topic on the MQTT Ship channel
- The VDES Ship radio receives the topic on the ship MQTT channel
- The VDES Ship radio sends to the VDES Shore radio a resource allocation request (msg ID 90)
- The VDES Shore radio replies sending to VDES ship radio a resource allocation information (msg ID 4)
- The VDES Ship radio sends to the VDES shore radio the relevant message on the allocated TDMA time slot
- The VDES Shore radio forwards the topic to the **`shore`** VDES-GW application via the shore MQTT channel
- The **`shore`** VDES-GW application receives the **`/evacuation-coordinator`** topic. This reception is logged but the message is not forwarded it to the Shore ICT infrastructure.

- If the evacuation status contained in the **/evacuation-coordinator** message indicates an evacuation condition, the VDESGW Ship application starts to generate periodically a **/mayday-message**.
- The **/mayday-message** is forwarded to shore by the VDES radio on the previously allocated TDMA slot, and the VDES-GW Shore application forwards it to the ICT Shore.
- When the evacuation condition terminates, the VDES-GW Ship stops to send the **/mayday-message**.



NOTE: To monitor the generation of the **/evacuation-coordinator** and **/mayday-message** topics on shore side, the following command can be entered:

mosquitto_sub -h localhost -t /evacuation-coordinator -t /mayday-message

Additionally, the VDES Gateway Shore application logs shall be checked to verify that the **/evacuation-coordinator** and **/mayday-message** topics are received, and that the **/mayday-message** topic is sent to the DFB Kafka cluster. On the DFB cluster side collect the evidence of the reception of the **/mayday-message** topic.

2.5.5 Test Outputs

Evacuation coordinator and Mayday messages has to be received from shore gateway and from the DFB shore cluster.

2.5.6 Test Results

The test will be considered OK if the message was received correctly from the shore, FAIL otherwise.