

HELLENIC MEDITERRANEAN UNIVERSITY
DEPARTMENT OF ELECTRONIC ENGINEERING



MASTER THESIS

**Edge Computing Technologies
for Internet of Maritime Things (IoMaT) Systems,
in Vessel Performance Monitoring**

Author:
Athanasios G. RITAS

Supervisor:
Stylianos KOURIDAKIS, As. Professor HMU
Examination Committee Member's:
Emmanuel ANTONIDAKIS, Professor HMU
Evangelos KOKKINOS, As. Professor HMU

*A thesis submitted in fulfillment of the requirements
for the degree of **Master of Sciences***

in the

"Telecommunication and Automation Systems"

Copyright ©Athanasios G. RITAS, 30 September 2022

Declaration of Authorship

I, Athanasios G. RITAS, declare that:

This thesis titled, "Edge Computing Technologies for Internet of Maritime Things (IoMaT) Systems, in Vessel Performance Monitoring." and the work presented in it are my own.

I confirm that:

- The Subject of this Research is Suggestion of Mr Serafeim KATSIKAS, Co-Founder and CPO of "METIS Cyberspace Technology S.A".
- This work was done wholly or mainly while in candidature for a Master Degree at the Electronic Engineering, Dept of, Hellenic Mediterranean University.
- Where any part of this Thesis has previously been submitted for a Degree or any other Qualification at this University or any other Institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this Thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the master thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.
- The opinions and conclusions contained in this document are those of the author and should not be interpreted as representing the official positions of the Hellenic Mediterranean University.

Signed: Athanasios G. RITAS

Date: 30 June 2022

“Thanks to my Wide and Comprehensive Training in matters of the Science of Electronic Engineering, on the Master Degree Program [“Telecommunication and Automation Systems”](#) that I attended.

Also Thanks to the training in real applications, from the Staff of METIS Cyberspace Technology S.A.. I meet advance technologies of applied electronic engineering, information science and artificial intelligence on applications for merchant ships . Specially for the vessels performance monitoring and non-destructive evaluation-inspection..

But also, under the guidance of my Supervisor, Professor Stylianos KOURIDAKIS.

Today I can easy analyze problems from wider area of this research. I can design applicable, effective & efficient solutions in vessels ecosystems.”

Athanasios G. RITAS

Dedication.

"I would like to dedicate this work to my Father

George Ath. RITAS.

Who never saw this academic adventure.

Because he lives in The Angels Neighborhood since 2001. "

Athanasios G. RITAS

Acknowledgements

I gratefully acknowledge advice and support from all person's, at the Master Degree Program "[Telecommunication and Automation Systems](#)" of the [Department of Electronic Engineering](#) of [Hellenic Mediterranean University](#).

In particular:

My Supervisor,
Dr. Stylianos KOURIDAKIS,

The Members of examination committee,
Dr. Emmanuel ANDONIDAKIS.
Dr. Evangelos KOKKINOS.

Also the following Professors,
Dr. Ioannis CHATZAKIS, Chairman of Electronic Engineers Dept.
Dr. Ioannis VARDIAMPASIS, Director of Master Degree Program.
Dr. Antonios KONSTANTARAS, Former Chairman of Electronic Engineers Dept.
Dr. Konstantinos PETRIDIS, Academic Coordinator Officer on IRO of HMU.
Dr. George LIODAKIS.
Dr. Nikolaos PETRAKIS.
Dr. George FOUSKITAKIS.
Dr. Ioannis BARBOUNAKIS
Dr. Ioannis MAKRIS.
Dr. Emmanuel MARAVELAKIS
Dr. Anastasia KATSAMAKI.

My personal friends, and fellow-students in this Master Degree Program:
MSc George MESARCHAKIS, Electronic Eng. (Technician of Maritime Electronic)
MSc Christos ALEXOPOULOS Radio-Electronic Eng. (Militarian -Technical Staff of Hellenic Navy)
MSc Lambros BOGDANIS, Electronic Eng. (Technical Staff of Hellenic & NATO Navies)

The research presented here, was carried out as part of the requirements for the degree of Master of Sciences in "[Telecommunication and Automation Systems](#)".

Suggestion for research in this area became from Mr Serafeim KATSIKAS, Co-Founder and CPO of "[METIS Cyberspace Technology S.A.](#)". An ICT company, specialized in Maritime 4.0 Technologies. I am grateful for his continuous mentoring and knowledge supporting.

Thank You all of You,

- for your time,
- your knowledge,
- and the excellent & invaluable advice.

Athanasios G. RITAS

HELLENIC MEDITERRANEAN UNIVERSITY

Abstract

School of Engineering
Department of Electronic Engineering

Master of Sciences

Edge Computing Technologies for Internet of Maritime Things (IoMaT) Systems, in Vessel Performance Monitoring.

by Athanasios G. RITAS

This master's thesis was written to study the requirements and standards of Industry 4.0 in relation to the Science of Electronic Engineering, in the so-called "Shipping Industry".

It is known, the application of that requirements to ships has some particularities due to the nature of the ship. For example, changes in: geographic location (during the voyage), environmental conditions (weather, sea state, etc.) and maritime commercial activity (density of passing ships). Unforeseen situations may arise, eg acceleration of the aging of electromechanical equipment, increase in the possibility of dangerous situations during navigation, increase in sailing time (means increased fuel consumption).

The areas and official indicators (SPI/KPI) that need increased surveillance, also the analysis & decision-making methods were studied.

An extensive search was made for all applied and able apply Maritime Internet of Things Technologies (IoMaT) and the Development tools. In this search, i have found and mention all relative about with: hardware, software, efficient computing technologies, internet of things protocols, the NMEA family of maritime standards, the Signal-K standard, navigation bridge electronics & information systems, applicable Non-Destructive Testing methods, data representation & visualization methods and the communication systems used.

The collecting ways of the vast amount of data, that continue be generated by New IoT Sensor Networks (IoMaT) and by existing on-board monitoring-data-logging systems were studied. Which data, must be cleaned, stored and processed with various computational techniques (statistical – empirical methods, artificial intelligence or machine learning). At this point, Computational Science Methods were Evaluated in relation to the Difficulties presented in Ship Internet Connectivity. And in conclusion everything converges, in that it is necessary to use the philosophy of "Edge Computing Technologies for Maritime Internet of Things Systems."

Also for the understanding, consolidation of the technologies and the conclusions of the theoretical study. I developed an application based on the Signal-K model. The application includes 3 study (programming) cases: generation of NMEA messages with data generation simulation, creation of NMEA Gateway Server and a Web-Dashboard Server for IoMaT Data Visualization.

Περίληψη Διπλωματικής Εργασίας

Ελληνικό Μεσογειακό Πανεπιστήμιο
Σχολή Μηχανικών, Τμήμα Ηλεκτρονικών Μηχανικών

Τίτλος Διπλωματικής Εργασίας:

“Τεχνολογίες Υπολογιστικής των Άκρων για Συστήματα του Διαδικτύου των Ναυτιλιακών Πραγμάτων (IoMaT), στην Επιτήρηση των Αποδόσεων των Πλοίων.”

Αθανάσιος Γ. Ρήτας

Αυτή η μεταπτυχιακή εργασία γράφτηκε για να μελετηθούν οι απαιτήσεις και τα πρότυπα της Βιομηχανίας 4.0 σε σχέση με την Επιστήμη της Ηλεκτρονικής Μηχανικής, στη λεγόμενη "Ναυτιλιακή Βιομηχανία".

Είναι γνωστό ότι η εφαρμογή αυτών των απαιτήσεων και των προτύπων στα πλοία, έχει κάποιες ιδιαιτερότητες λόγω της φύσης του πλοίου.

Για παράδειγμα οι αλλαγές: της γεωγραφικής θέσης (κατά τη διάρκεια του ταξιδιού), περιβαλλοντολογικές συνθήκες (καιρός, κατάσταση θάλασσας κτλ) και η θαλάσσια εμπορική δραστηριότητα (πυκνότητα παραπλεόντων πλοίων). Μπορεί να δημιουργηθούν απρόβλεπτες καταστάσεις πχ. επιτάχυνση της γήρανσης του ηλεκτρομηχανολογικού εξοπλισμού, αύξηση της πιθανότητας επικίνδυνων καταστάσεων κατά τη διάρκεια της πλοήγησης, αύξηση του χρόνου πλεύσης (σημαίνει αυξημένη κατανάλωση καυσίμου).

Μελετήθηκαν οι τομείς και οι επίσημοι δείκτες (SPI/KPI) που χρήζουν αυξημένης επιτήρησης καθώς και οι μέθοδοι ανάλυσης & λήψης αποφάσεων.

Έγινε εκτεταμένη αναζήτηση για όλες τις εφαρμοσμένες και εφαρμόσιμες Τεχνολογίες του Διαδικτύου των Ναυτιλιακών Πραγμάτων (IoMaT), όπως επίσης και των Αναπτυξιακών εργαλείων. Σε αυτή την αναζήτηση εντοπίστηκαν και αναφέρονται: τεχνολογίες υλικού, λογισμικού, αποτελεσματικής υπολογιστικής, πρωτόκολλα του διαδικτύου των πραγμάτων, η οικογένεια των ναυτιλιακών προτύπων NMEA, το πρότυπο Signal-K, τα ηλεκτρονικά & πληροφοριακά συστήματα της γέφυρας ναυσιπλοΐας, εφαρμόσιμοι μέθοδοι Μη Καταστροφικών Ελέγχων, μέθοδοι αναπαράστασης & οπτικοποίησης των δεδομένων και τα επικοινωνιακά συστήματα που χρησιμοποιούνται.

Μελετήθηκαν οι τρόποι συλλογής του τεράστιου όγκου δεδομένων που συνεχίζουν να δημιουργούνται από Νέα Δίκτυα αισθητήρων IoT (IoMaT) και από τα υπάρχοντα συστήματα παρακολούθησης-καταγραφής δεδομένων επί του πλοίου. Τα οποία δεδομένα, πρέπει να καθαριστούν, να αποθηκευτούν και να υποβληθούν σε επεξεργασία με διάφορες υπολογιστικές τεχνικές (στατιστικές – εμπειρικές μέθοδοι, τεχνητή νοημοσύνη ή μηχανική μάθηση). Σε αυτό το σημείο αξιολογήθηκαν οι Μέθοδοι της Υπολογιστικής Επιστήμης σε σχέση με την δυσκολίες που παρουσιάζονται στη Συνδεσιμότητα των Πλοίων με το Διαδίκτυο.

Και συμπερασματικά όλα συγκλίνουν, στο ότι είναι απαραίτητη η χρήση της φιλοσοφίας των "Τεχνολογιών Υπολογιστικής των Άκρων για Συστήματα του Διαδικτύου των Ναυτιλιακών Πραγμάτων."

Επίσης για την κατανόηση, την εμπέδωση των τεχνολογιών και των συμπερασμάτων της θεωρητικής μελέτης. Ανέπτυξα μία εφαρμογή που στηρίζεται στο μοντέλο Signal-K. Η εφαρμογή περιλαμβάνει 3 περιπτώσεις μελέτης (προγραμματισμού): δημιουργία μηνυμάτων NMEA με εξομίωση παραγωγής δεδομένων, δημιουργία NMEA Gateway Server και έναν Web – Dashboard Server για την Οπτικοποίηση των Δεδομένων.

Keywords :

Edge - Fog - Cloud Computing, GMDSS, AIS, ECDIS, Vessel Performance Monitoring, IoT, Internet of Things, Engine Room, Navigation, Captain , 1st Engineer, Big Data Analytics, Machine Learning, Ship, Vessel, Inmarsat, Satellite, VHF DSC, Radar Arpa, C/C++, Python, Micropython, Jinja, Javascript, HTML, XML,NMEA, JSON, Signal-K,Data Driving Decisions, Performance Monitoring, KPIs, SPIs, PIs, Meteo.

Contents

Edge Computing Technologies for Internet of Maritime Things (IoMaT) Systems, in Vessel Performance Monitoring.	i
Declaration of Authorship	iii
Quotation.	vii
Dedication.	ix
Acknowledgements	xi
Abstract	xiii
Περίληψη.	xv
Keywords.	xvii
Table of Contents	xix
List of figures	xxvii
List of Listing Source Codes.	xxxi
List of Abbreviations.	xxxiii
1 Introduction	1
In introductory chapter,	1
1.1 A bit of History.	1
1.2 Motivations.	1
1.2.1 What is the “Vessel performance monitoring V.P.M.”	2
1.2.2 Digital Shipping	2
1.2.3 Review of Industry 4.0	2
1.3 Main Technological Requirements.	4
1.3.1 Collecting and Managing the ship data.	4
1.3.2 IoT And Computing.	4
Cloud Computing	5
Edge Computing	5
1.3.3 Vessel ecosystem peculiarities for Design IoT Systems.	5
1.4 Challenges of Computing	6
1.5 Structure of Master Thesis.	6
Chapter 1:	6
Chapter 2:	6
Chapter 3:	6
Chapter 4:	6
Chapter 5:	7

Chapter 6:	7
Appendix A:	7
Appendix B:	7
Appendix C:	7
Appendix D:	7
2 Vessel performance monitoring	9
In this chapter,	9
2.1 Defining of Vessel Performance Monitoring (VPM).	9
2.2 Key Areas of Interest in Vessel Performance Monitoring.	10
2.2.1 Fuel Reduction	10
2.2.2 Engine Room Department and Non-destructive Evaluation 4.0	11
Types of Maintenance Procedures	11
1. Preventive or Scheduled Maintenance System	11
2. Corrective or Breakdown Maintenance	11
3. Condition Maintenance system	11
Maintenance Schedule.	12
2.2.3 Cargo Operations	12
2.2.4 Navigation, Safety - Security Management and e-Navigation.	12
2.3 Methods of Analysis in VPM	13
2.3.1 Probabilities and Statistical Analysis.	13
Probability.	13
Quantitative Analysis.	13
Qualitative Analysis.	14
Characteristics of qualitative research methods	14
2.4 Decisions in VPM	14
2.4.1 Data Driven Decisions	15
2.4.2 Empirical and Statistical Decisions	15
Empirical Decisions.	15
Become Decisions Experts.	15
Statistical Decisions.	16
2.4.3 Computational Decision.	16
2.5 Shipping KPIs.	17
Concept	17
Performance Indicators (PIs)	18
Key Performance Indicators (KPIs)	18
Shipping Performance Indicators	18
2.6 Meta data - Ship Attributes	18
2.6.1 Types of Metadata	19
Technical Metadata	19
Business Metadata	19
Operational Metadata	19
Descriptive Metadata	19
Metadada in AIS.	19
3 Internet of Maritime Things Technologies and Development Tools.	21
This chapter,	21
3.1 Definitions of Internet of Things (IoT).	21
3.1.1 Internet of Things (IoT).	21
3.1.2 Industrial Internet of Things (IIoT).	21
3.1.3 Internet of Maritime Things (IoMaT).	21

3.2	Electronic Hardware.	22
3.2.1	Processing Device's.	22
	Central processing unit (CPU)	22
	Multi-core processor	22
	Graphics processing unit (GPU)	22
	Digital signal processor (DSP)	23
	Microcontrollers.	24
	Data processing unit (DPU).	24
3.2.2	Data Storage System at the edge.	26
	Storage Types for Edge Servers.	26
	DAS (Directly Attached Storage)	26
	NAS (Network Attached Storage)	27
	SAN (Storage Area Network)	27
	Computational Storage Architecture.	28
	The "Voyage data recorder" System.	28
	Voyage data	29
3.2.3	Embedded Systems.	31
	Real-Time Embedded Systems	32
3.2.4	Sensors Systems	32
	A base sensor	32
	A smart sensor	32
	Sensors for Instrumentation	33
	NMEA Sensors	34
3.2.5	Interconnections with shipbuilding DAQ Systems	34
3.3	Software.	35
3.3.1	Operating Systems and Firmware.	35
	What is a General Purpose Operating System?	35
	Embedded OS's	35
	What is Embedded Operating System?	35
	Types of Embedded Operating Systems	36
	Single System Control Loop	36
	Multi-Tasking Operating System	36
	Rate Monotonic Operating System	36
	Preemptive Operating System	36
	Real Time Operating System	36
	Real-Time Operating System (RTOS)	37
	What is an RTOS?	37
	RTOS in the IoT era,	38
	Firmware	38
	BIOS	38
3.3.2	Programming Languages	39
	Choice of Influences	39
	Need of Firmware	39
	The Maritime Edge Computing	39
	Programming Languages for IoT Projects	40
	Assembly	40
	C/C++	40
	Python 3.x	40
	Java.	41
	JavaScript and Node JS.	41
	Node-RED	41

	C# (C Sharp).	42
	Rust.	42
3.4	Summary of High Performance Computing Technologies.	43
3.4.1	Classes of parallel computing.	43
	Multi-core computing	43
	Cluster computing	43
	Basics components of a cluster:	44
	Grid computing	44
	General-purpose computing on graphics processing units (GPGPU)	45
3.4.2	Parallel Programming Languages,APIs.	45
	CUDA	45
	CUDA C/C++	46
	CUDA® Python	46
	C++ Accelerated Massive Parallelism	46
	Open Computing Language	46
	OpenCL 3.0	46
	C++ for OpenCL language	47
3.5	IoT Protocols & Standards.	47
3.5.1	IoT Data Protocols	48
	HTTP(Hypertext Transfer Protocol)	48
	MQTT (Message Queuing Telemetry Transport)	48
	CoAP (Constrained Application Protocol)	49
	AMQP (Advanced Message Queuing Protocol)	49
	DDS (Data Distribution Service)	49
3.5.2	Network Protocols for IoT	49
	Industrial Ethernet	49
	WiFi	50
	Bluetooth	50
	ZigBee	51
	LoRaWan	51
3.5.3	The JSON Data Interchange Standard Format.(ECMA-404)	51
3.5.4	The WebSocket Protocol.	52
3.5.5	Extensible Markup Language (XML) Format.	53
3.6	Protobuf by Google.	55
3.7	NMEA Standards.	56
3.7.1	NMEA 0183 Interface Standard	56
	Latest Version of NMEA 0183 - 4.11	56
	NMEA 0183 HS (High Speed)	57
3.7.2	NMEA 2000® Interface Standard	57
3.7.3	OneNet Standard for IP Networking of Marine Electronic Devices	57
	OneNet Goals	58
	OneNet Key Benefits	59
3.7.4	NMEA Data Protocol	59
	Address Fields	59
	Data fields	59
	Checksum	60
	Example	60
3.8	Signal K	62
3.9	Navigation Bridge Systems	63
3.9.1	Radar.	63
3.9.2	Satellite Based Positioning and Navigation Systems.	63

3.9.3	Maritime Human-Machine Interfaces (M-HMI)	64
	Automatic Radar Plotting Aid (ARPA).	64
	Electronic Chart Display and Information System (ECDIS)	64
3.9.4	Integrated bridge system (IBS)	65
3.9.5	AIS	66
3.10	Non -Destructive Evaluation 4.0	67
3.11	Visualizing Data and Info's	68
3.11.1	What Maritime info's (KPIs) we Visualize?	68
3.11.2	Visual Shipping KPI's Dashboard.	69
3.11.3	AI Agent's	70
	Building an intelligent agent	71
3.12	Communication Systems	71
3.12.1	Vessels communications planning.	71
3.12.2	Maritime Communication Systems.	71
	GMDSS Overview.	71
	The main types of equipment used in GMDSS are:	72
	GMDSS sea areas.	72
	Maritime Mobile Service Identity (MMSI).	73
3.12.3	International Mobile Satellite Organization (IMSO).	74
	Inmarsat.	75
	Iridium.	75
	Iridium - GMDSS :	75
3.12.4	LTE/4G and 5G in Maritime Connectivity.	75
3.12.5	Internet	75
3.12.6	Starlink	77
	Starlink Maritime.	77
3.12.7	Latency	79
	Communication Latency	79
	Satellite's Transmission Latency	79
	Audio/video latency,	79
	Network latency,	79
4	Collecting, Manipulating and Compute Data for VPM.	81
	In this chapter,	81
4.1	About Data.	81
4.1.1	Data Definitions.	81
4.1.2	Big Data.	81
4.1.3	Big Data Characteristics.	82
	Volume,	82
	Velocity,	82
	Variety,	82
4.1.4	The Collected Data Types.	82
4.2	Data Manipulating (Managing)	83
4.2.1	Data Managing general overview.	83
4.2.2	IoMaT Collecting Big Data from Multi-sources.	83
4.2.3	Raw Data ("unprocessed").	83
4.2.4	Cleaning Data.	83
4.2.5	Data Analytics.	84
	Descriptive analytics,	84
	Diagnostic analytics,	84
	Predictive analytics,	84

	Prescriptive analytics,	84
4.2.6	Presenting Data in VPM.	84
4.3	Computing Definitions.	85
4.3.1	Computing and Super Computing.	85
	Computing,	85
	Supercomputing,	85
4.3.2	Cloud Computing	85
	Cloud Computing Advantages	85
	Disadvantages of Cloud Computing	86
4.3.3	Vessel Inter-networking and Disadvantages of Cloud Computing.	87
4.3.4	Computing and Edge Computing in IoMaT.	87
5	Case studies for Edge Computing	89
	In this chapter,	89
5.1	Case Study 1: Generating IoMaT and NMEA Data.	90
5.1.1	Meteorological Data from Sensor Hat Emulator and Python3 scripting.	90
5.1.2	Satellite Navigation NMEA Data by Raspberry Pi Pico and Micro Python.	92
5.1.3	Engine Room NMEA Data by Arduino Uno and Arduino C/Cpp.	94
5.2	Case Study 2: NMEA Gateway Server Application.	97
5.2.1	Identify and Reading USB ports.	97
5.2.2	Logging incoming data.	97
5.2.3	Parse NMEA messages.	98
5.2.4	Transform NMEA messages to JSON format.	99
5.3	Case Study 3: Web-based IoMaT Dashboard.	101
5.3.1	Python with Flask Web-server.	101
5.3.2	User Authentication Screen.	102
5.3.3	Navigation Bridge Panel.	103
5.3.4	Vessel Meteo Station Panel.	105
5.3.5	Engines Room Control Panel.	106
5.3.6	Warnings and Alerts, Panel.	108
5.3.7	Archiving Data Files, Access Panel.	111
5.3.8	Optional, by web-access Poseidon Meteo forecast System.	114
6	Conclusions	117
	This is the last chapter	117
6.1	Sort review, of this Master Thesis.	117
6.2	Is real necessary the Edge Computing Technologies??	117
6.3	Usage Advantages of Edge Computing Philosophy within IoMaT, for Vessel Performance Monitoring Case.	118
6.4	Technological Challenges in the IoT and IoMaT Field's.	118
6.5	Contribution in Applied Electronic Engineering Science.	119
A	Software and Hardware Technologies, who I used in Case Studies in Chapter 5.	121
A.1	About Jinja2.	121
A.2	About Flask.	122
A.3	About Raspberry Pi Pico.	123
	RP2040	123
A.4	About Micro Python.	125
A.5	About Arduino UNO.	126

B	Relative Maritime Organizations, Authorities and Systems.	127
B.1	BIMCO	127
B.2	EMSA	127
B.3	GMDSS	127
B.4	IMSO	128
B.5	IMO	128
B.6	INMARSAT	128
B.7	NMEA	128
B.8	OCIMF	128
B.9	STO	128
B.10	SOLAS	129
B.11	AIS.	129
B.12	LRIT	129
C	Relative Technological Challenges.	131
C.1	Cybersecurity on Maritime 4.0.	131
C.2	Maritime Autonomous Surface Ships (MASS).	132
C.3	How can use Blockchain Technology.	133
	C.3.1 Blockchain definition.	133
	C.3.2 Blockchain in Maritime 4.0	134
D	Bibliography	135
D.1	Article's.	135
D.2	BSc-MSc Thesis and PhD Dissertations	139
D.3	Books List.	140
D.4	Reports, White Papers etc.	141
D.5	On Line Sources.	142

List of Figures

1.1	A Bit of History	1
1.2	Industry History by presentationpoint.com	3
1.3	The IoT layers.(by https://www.frontiersin.org)	4
2.1	Overview of VPM , for Fuel Consumption Monitoring (by " Ascenz Solutions Pte Ltd")	9
2.2	Main 5 factors for lower fuel costs for Shipping companies (by "marine-digital[dot]com")	10
2.3	Cargo control room of a oil tanker. https://www.123rf.com/	12
2.4	Data Driven Image.	15
2.5	Shipping Performance Indexes -SPI's (by BIMCO)	17
2.6	Math Formula to calculate value of an SPI (by BIMCO)	18
2.7	Vessel SPI Metadata(by BIMCO)	19
3.1	Diagram of a generic dual-core processor with CPU-local level-1 caches and a shared, on-die level-2 cache (By wikipedia [dot]org)	22
3.2	Abstract architectural block diagrams of PC-class CPU-GPU systems and HMP-SoCs. (A) PC-class CPU-GPU systems. (B) HMPSoCs. (by sciencedirect[dot]com)	23
3.3	Block Diagram of Digital Signal Processing System (by wikipedia)	23
3.4	Basic layout of a microcontroller.	24
3.5	Comparison Compute VS Data Centric Architectures. (by www.fungible.com)	25
3.6	Storage Types Comparison (by networkwalks [dot com]).	25
3.7	Storage Types Comparison Table by networkwalks.com.	27
3.8	Classic and Computational Storage (by www.snia.org)	28
3.9	A Voyage Data Recorder (VDR), the "black box" of a ship, on the container ship SMA CGM Nabucco by wikipedia.org.	29
3.10	A Voyage Data Recorder (VDR), general diagram overview. (by maritimecyprus.com)	30
3.11	Embedded Systems Base Diagram (by 'omnisci[dot]com)	31
3.12	Types-of-Sensors	33
3.13	Smart-Sensors	33
3.14	Marine sensors (by VDO-Marine[dot]de)	34
3.15	A comparison between non-RTOS and RTOS executions. (by "Texas Instruments")	37
3.16	Legacy BIOS boot process.(From Wikimedia Commons)	38
3.17	Top Programming Languages for IoT.(by "iotcentral[dot]io")	40
3.18	Node-Red action flows and Java Script.(by "Node Red Organization.")	41
3.19	Official Logo of the Rust Programming Language.	42
3.20	High Performance Computing Architecture. (by "conocophillips[dot]com")	44
3.21	Cuda Processing flow. (By " https://en.wikipedia.org/wiki/CUDA ")	46
3.22	C++ for OpenCL. (By " www.khronos.org ")	47
3.23	Types of IoT Protocols.(by "Kellton Tech [dot] com")	48
3.24	Dictionary - object, Array - list and Values definitions (by "JSON Org.")	52
3.25	Websockets Overview.	52
3.26	Protocol Buffers Logo by Google.	54

3.27	Protocol Buffers Main Idea Block Diagram.	55
3.28	Protocol Buffers Procedure Diagram.	55
3.29	NMEA 0183 Interface. (by "NMEA Org.")	56
3.30	NMEA 2000 Logo. (by "NMEA Org.")	57
3.31	OneNet Logo. (by "NMEA Org.")	57
3.32	OneNet NMEA Standard Overview. (by "NMEA Org.")	58
3.33	The structure of a NMEA protocol message	60
3.34	Example NMEA: "xxGGA" Message.	61
3.35	Example NMEA GGA Sentence Decoding. (by brandidowns.com)	61
3.36	The Signal K Data Model. (by "Signal-K Org.")	62
3.37	Rose-Point-ECS-Radar-Overlay-Feature-img. (by veinland.net)	65
3.38	AIS overview. Image by Adam Weintrit on researchgate dot net	66
3.39	The essential aspects of NDE 4.0	67
3.40	Hierarchy of Shipping KPI's. (by "shipping-kpi[dot]org")	68
3.41	Live Vessel Performance Monitoring. (by navigationlaptops.com)	69
3.42	A Intelligent Agent. by oreilly.com	70
3.43	Functional Requirement of GMDSS(IMO, 2013)	72
3.44	GMDSS Areas. (by danphone.com)	73
3.45	AIS ECDIS MMSI (by www.collins-marine.co.uk)	74
3.46	Inmarsat Global Xpress. Ka-Band.	76
3.47	Iridium constellation coverage footprint's, by Mobile Internet Resource Center.	76
3.48	Starlink Constellation by https://satellitemap.space	77
3.49	Starlink Maritime Coverage Map.	78
3.50	Latency Time from Maritime Communications Systems.	79
3.51	Comparison of Latency for basic Satellite Types. By dglinfra [dot] com	80
4.1	Big Data 3V's.(by springer.com)	81
4.2	General View of Dashboard Graphs. (by fiverr.com)	84
5.1	IoMaT Application Overview Structure	89
5.2	Hat and Sensor Hat Emulator.(by www.raspberrypi.org)	90
5.3	NMEA Sentences from Satellite Positioning System	93
5.4	NMEA Sentences form Engine Room (ERMED)	93
5.5	NMEA Gateway Server	96
5.6	Web-based Dashboard overview diagram.	101
5.7	User authentication page.	103
5.8	Navigation Bridge Information Panel.	104
5.9	Meteo Station Panel.	106
5.10	Engines Room Control Panel.	107
5.11	Warnings and Alerts Panel.	109
5.12	Edit Warnings and Alerts Panel.	110
5.13	Current Data Files,Access Panel.	112
5.14	Fetchd Archived Data Files,Access Panel.	113
5.15	Direct View of Current and Archived Data Files.	113
5.16	Poseidon Meteo Page,Atmospheric Pressure Forecast.	115
5.17	Poseidon Meteo Page,Rainfall Forecast.	115
5.18	Poseidon Meteo Page, Cloudiness Forecast.	116
5.19	Poseidon Meteo Page, Wave Height and Direction Forecast.	116
A.1	Jinja2 engine logo.	122
A.2	Fask web api logo.	122

A.3	Raspberry Pi Pico Pinout and Dimensions.(by www.raspberrypi.org)	123
A.4	Micro Python logo.	125
A.5	Arduino Uno R3 Pinout Diagram.	126
A.6	Atmega328P Pinout Diagram.	126
C.1	Cyber Security weak point in a vessel, by gard.no	131
C.2	MASS Control Scheme by EMSA.	132
C.3	Classification of autonomous maritime system and autonomous ship types (O.J. Rødseth, H. Nordahl, 2017).	132
C.4	Blockchain information flow in the import carrier process (Oude Weernink et al., 2017)	133
C.5	Building a Blockchain. (by faoglobal.com)	134

Listings

2.1	Sample Vessel Metadata (json file)	20
3.1	Sample of JSON Format File.	53
3.2	Sample of XML Format File.	54
5.1	Python 3.7 Import sense_emu module	91
5.2	Using sense-emu methods with Jinja2 in Html code.	91
5.3	Python 3.7 pure code for meteo emulating	91
5.4	Micro Python Code for emulating GPS NMEA sentences.	92
5.5	Arduino C/C++ code for generating NMEA sentences.	94
5.6	Identify and Reading USB/Serial NMEA Ports.	97
5.7	Logging incoming NMEA data.	98
5.8	Parsing NMEA sentences.	98
5.9	Combining NMEA in JSON Format.Sample code.	99
5.10	NMEA-GPS Data in JSON Format.	99
5.11	NMEA-ERMED Data in JSON Format.	100
5.12	Basic Python3 with Flask module sample web application.	101
5.13	User authentication page.Python -Flask code.	102
5.14	User authentication page.Html code.	102
5.15	Navigation Bridge Information Panel.Python -Flask code.	103
5.16	Navigation Bridge Information Panel.Html sample code.	104
5.17	Meteo Station Panel.Python -Flask code.	105
5.18	Meteo Station Panel.Html sample code.	105
5.19	Engines Room Control Panel.Python -Flask code.	106
5.20	Engines Room Control Panel.Html sample code.	107
5.21	Warnings and Alerts Panel.Python -Flask code.	108
5.22	Warnings and Alerts Panel.Html sample code.	109
5.23	Warnings and Alerts Json file.	110
5.24	Archiving Past Data Files.Python code.	111
5.25	Logging Data Files Panel.Python -Flask code.	111
5.26	Logging Data Files Panel.Html sample code.	112
5.27	Calculating Position Coordinates for Focusing Poseidon App. Python -Flask code.	114
5.28	Redirecting in "blank tab" Poseidon Meteo Page.Html sample code.	114
A.1	RPi Pico dual-core programming example in Arduino-C.	123

List of Abbreviations

IoT	Internet of Things
IoMaT	Internet of Maritime Things
VPM	Vessel Performance Monitoring
Cd.C	Cloud Computing
Ed.C	Edge Computing
BDA	Big Data Analytics
CPU	Central Processing Unit
GPU	Graphics Processing Unit
DSP	Digital Signal Processor
DAS	Directly Attached Storage
NAS	Network Attached Storage
SAN	Storage Area Network
RTOS	Real Time Operating System
BIOS	Basic Input Output System
HPC	High Performance Computing
GPGPU	General Purpose Computing on Graphics Processing Units
SDK	Software Development Kit
API	Application Programming Interface
CUDA	Compute Unified Device Architecture
OpenCL	Open Computing Language
C++ AMP	C++ Accelerated Massive Parallelism
NMEA	National Marine Electronics Association
DAQ	Data Acquisition
KPIs	Key Performance Indicators
SPIs	Shipping Performance Indicators
IA	Intelligent Agent
DDD	Data Driven Decisions
EdStAr	Edge Storage Architecture

Chapter 1

Introduction

In introductory chapter, I refer to the core idea of my research and I associate motivations, technological requirements, and challenges to future development and applications.

1.1 A bit of History.

The Marine Industry (fisheries, cargo or human transportation), has already offered too much to humanity. And promise's to continue offer more and more . Is really the largest carrier of goods throughout the human history in oceans, closed seas, rivers and lakes.

Due to the advantages of nature of the sea environment and technological capabilities of ships.

Of course need improvement in addition to the basics for propulsion and navigating. The evolving tools are used in ship management and communications. With nearby vessels, ship and cargo owners, local and international competent authorities.

The optimal management of the fuel consumption for propulsion ,in global level. The Non Destructive Evaluation & Inspection (NDE&I) Techniques for Clever Maintenance Managing.

These driving to reduce the environmental footprint from the vessels and increase economic benefits for shipping community.

1.2 Motivations.

Motivations for adoption of the "Edge Computing Technologies" over existing Networked Computing philosophies such as Cloud / Fog / Grid etc.

Including challenges such as inter-networking or operational constraints of data collecting - handling - utilizing and better cyber security in new digital era for merchant vessels .

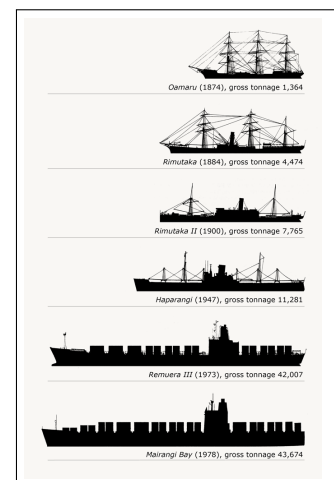


FIGURE 1.1:
A Bit of History

1.2.1 What is the “Vessel performance monitoring V.P.M.”

The case of Vessel Performance Monitoring is the visualization in real time all maritime data. Including (but not limited) the Key Performance Indicators ¹ (KPI's) of a merchant vessel provides financial and environmental benefits.[39, 70]

To achieve optimal operational decision have been developed and evaluated many tools. Mainly for reducing fuel consumption and maximizing profit we have to focus in following axis :

1. real time operational optimization
2. maintenance triggers and
3. evaluating technological interventions

Vessel Performance monitoring is also relevant to charter party analysis, vessel bench-marking and performing policy decisions.

The on-board systems in vessels are complex. It's common for data modeling and analysis techniques to be employed for extracting useful trends. All datasets and modeling procedures have an inherent uncertainty . For aiding the decision maker, the uncertainty must be quantified. In order to fully understand the economic risk ² of a decision. Non acceptable risk requires further investments in data quality and data-analysis techniques. [41, 42].

1.2.2 Digital Shipping

Merchant Marine Vessels are complex ecosystems. That characteristic require huge amounts of data for maximizing the efficiency.

The successful utilization of sensors and IoT in the industry, drive to a forward-thinking approach to leverage the benefits of Industry 4.0 in a more comprehensive manner in Shipping 4.0. While processes can be improved and advanced through such efforts in order the merchant marine to gain more benefit's from smart generation and intelligent handling of data [153].

Innovation in the Shipping 4.0 environment refers to configurations of large-scale interconnected physical and digital components, systems and infrastructures. Which enable the creation of novel business models and the provision of new maritime services.

Internet of Things (IoT), Big Data Analytics (BDA) and Cloud Computing are parts of the contemporary states of art technologies in "Shipping 4.0". [21, 28, 29, 62, 79, 94].

1.2.3 Review of Industry 4.0

Nowadays we live the fourth industrial revolution known as Industry 4.0. All companies to remain competitive in a globalized environment and changing demands of markets must constantly evolve their production and service systems.

The main purpose of Industry 4.0, is to achieve improvements in terms of automation and operational efficiency as well as effectiveness. These in turn have a large impact on industry and

¹Key performance indicators (KPIs) about a vessel, refer to a set of quantifiable measurements used to gauge a company's overall long-term performance. KPIs specifically help determine a company's strategies, financial, and operational achievements, especially compared to those of other businesses within the same sector.

²Definition of economic risk: "As the prospect of financial loss due to unforeseen changes in underlying “risk factors”. These risk factors are the key drivers affecting Maritime portfolio value and financial results."

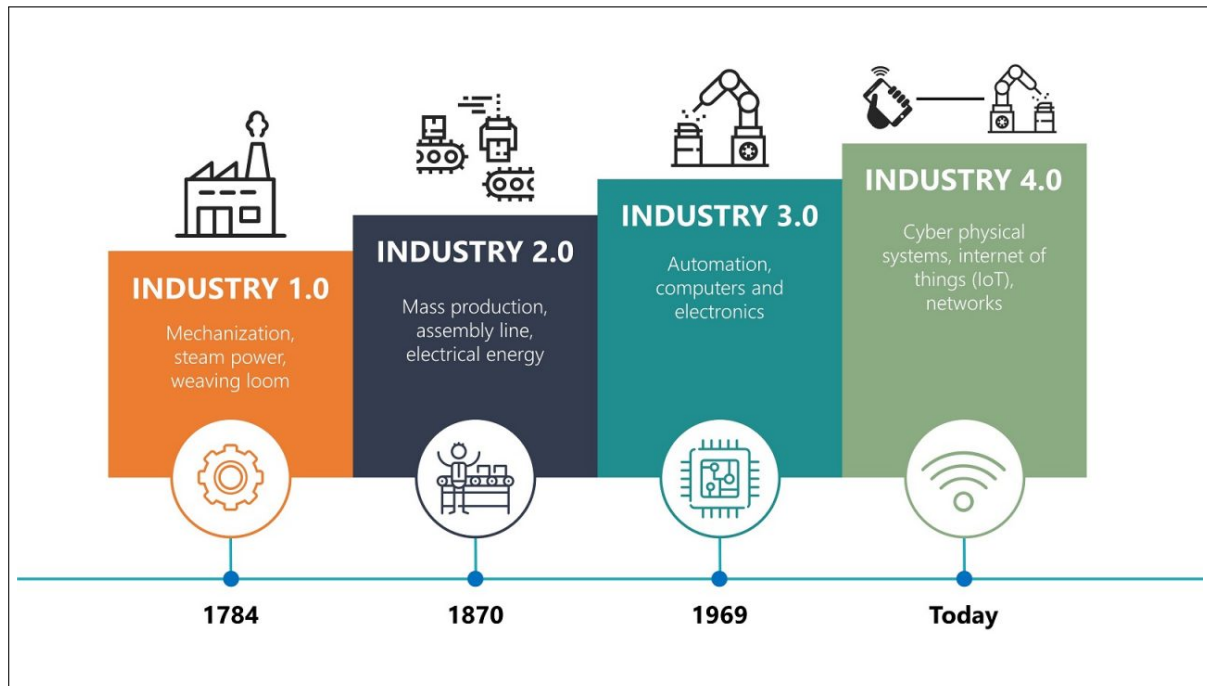


FIGURE 1.2: Industry History by presentationpoint.com

general in markets. Providing new means for production and conduct's of business. Allowing for improvement in processes and enhances in the competitiveness of enterprises. Computers, automation and robots existed in previous decades. The opportunities provided by the Internet, revolutionize their evolution. Increasingly cheaper solutions allowing us to monitoring the activities, operation and processes of machines, materials, the safety of workers and even products themselves. We collecting , analyzing and utilizing data in real-time for decision making.

The Industry 4.0 is based on data. The way it can be gathering, analyzing and using data to make the right decisions , become a mainly factor. Therefore the competitive advantage will not only be productivity on coordinated or aigen completely new basis. But how the companies support decision-making in any level of operation, by intelligent handling the data in order to reduce the cost and increase the economic benefits .

The Applied Sciences of Electronic Engineering and Informatics at the companies has transform both working conditions and efficiency of their workers, in all cases of utilization .For this reason their importance is unquestionable.

The emerging Industry 4.0 concept an umbrella term for new industrial standard which embraces a set of industrial field's of contemporary development. Including Cuber-Physical Systems (CPS), Internet of Things (IoT), Internet of Services (IoS), Robotics, Big Data Analytics (BDA), Cloud Manufacturing (CIM) and Augmented Reality (AuR).

The adoption of these technologies philosophies is essential to the development of more intelligent processes. Which including devices, machines, production modules, software applications and products that are able to independently exchange information, trigger actions and control each other thus enabling an augmenting reality environment.[63, 94, 133].

1.3 Main Technological Requirements.

The main technological requirements for the design and implementation of an IoT system on ships are the collection , management (handling) and processing (computing) of the necessary data. For utilizing the performance monitoring and optimization in vessel management.

1.3.1 Collecting and Managing the ship data.

The data generated by an IoT device as going through three different phases. The first, is the data creation itself. This phase takes place on the level of the IoT device from where the data are generated and transmitted over a network. The second phase of IoT data, is the collection and organization. The third phase involves the actual use of that data.

Opting of collecting all available IoT data and analyzing it in real-time should be backed by a clear rationale. The uninterrupted flow of IoT data, require network and computing resources. Otherwise the sheer volume of the data make an unnecessarily heavy impact on the cloud systems.

Some applications may allow for delays over a second. Others such as security and safety applications are characterized as time-critical. And needing time-frame cycle for delays less than few milliseconds.

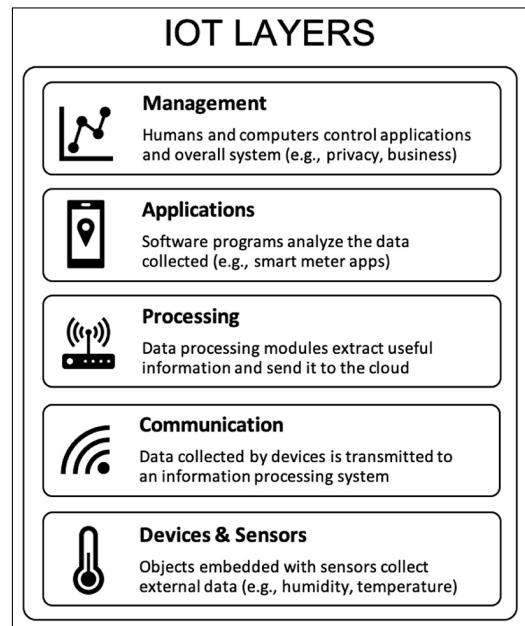


FIGURE 1.3: The IoT layers.(by <https://www.frontiersin.org>)

Many cases may not require high accuracy and send the data over in batches. In some scenarios we will need accurate real-time data for our analyses. In other scenarios historic data will do the job just as well. If the data is sent in batches or micro-batches we still get a record of all data. This takes place not in real time but only at given pre-established intervals. The choice of a particular use case depends on the application requirement. [55, 135]

1.3.2 IoT And Computing.

The Internet of things (IoT) is a system of interrelated computing devices, mechanical and digital machines provided with unique identifiers (UIDs) and the ability to transfer data over a network without requiring human-to-human or human-to-computer interaction.

The definition of the Internet of Things has evolved due to the convergence of multiple technologies real-time analytics, machine learning, commodity sensors and embedded systems.

Traditional fields of embedded systems, wireless sensor networks, automation (including home and industrial) and others. All contribute to enabling the Internet of Things. In the consumer market IoT technology is most synonymous with products pertaining to the concept of "smart home". Including devices and appliances (such as lighting fixtures, thermostats, home security systems, cameras etc.) that support one or more common ecosystems, and can be controlled via devices

associated with that ecosystem such as smartphones and smart speakers.

There are a number of serious concerns about the growth of IoT, especially in the areas of privacy and security. Consequently of the above worryment, industries and governments moves to address these scenarios have begun. [17, 97, 110, 113, 116]

Cloud Computing

Cloud computing is on-demand availability of computer system resources, especially data storage and computing power without direct active management by the user. This term is generally used to describe data centers available to many users over the Internet. Large clouds predominant today often have functions distributed over multiple physical locations for their central servers. If connection of the user is relatively close it may be designated an edge/fog server.

Edge Computing

Edge computing is a networking philosophy. Focused on bringing computing as close to the source of data as possible. In general order to reduce latency and bandwidth use.

In simpler terms edge computing means running fewer processes in the cloud and moving those to local (edge) places. Such as on a user's computer, an IoT device or an edge server. Bringing computation to the network's edge, minimizes the amount of long-distance communication that has to happen between a client and server. [35, 49, 64, 69, 71, 88, 89, 95, 97, 130, 141]

1.3.3 Vessel ecosystem peculiarities for Design IoT Systems.

Design, implementation and utilization of an IoT at a Vessel has some peculiarities. Due to ecosystem structure of the vessel and nature of sea transportation's.

1. **The Distinction in Areas** of Interest, for the installation and use of IoT systems on ships, carries out the following separate case studies:
 - (a) Engine room, generator, propulsion and rudder area.
 - (b) Cargo and fuel, loading-unloading area.
 - (c) Navigation ecosystem.
 - (d) Accommodation area, for crew and passengers.
2. **The Discrete Spaces** on a ship, such as engine room, generator room, engine room control center, cargo areas, navigation bridge, ship radio station room, etc. They are separated by metal (iron) walls. This creates prohibitive connectivity issues for the option of using a wide wireless sensor network (IoT). And it leads to the use of cable and mixed communication networks.
3. **The Interconnection of an IoT** that was installed on a ship, with the world of the Internet and the Cloud Computing. It is usually not technically possible due to geographical changes (voyages), because the average distance from coast's is out of 4G/LTE/5G coverage. If the connectivity with satellites networks is possible, it is very expensive for 24^{hours}7^{days}52^{weeks} for online and real-time use.

1.4 Challenges of Computing

Edge and cloud computing are two different solution approaches, as well as mutually beneficial to many applications on demand, for optimal results.

In contrary to Cloud computing, which assist the IoT systems to store the data and perform computation in order to control and manage the vast amount of data generated by these IoT sensor devices. But the major challenge in Cloud computing is to meet requirements in real-time connection of IoT systems to the internet.

The Edge Computing, is a networking and computing philosophy. Helping to communicating, managing, storing, and processing the data that response in real time. This is made possible by moving these functionalities closer to the End Users. For Edge Computing, the main challenge is to solve three main problems. That is:

1. The **integration of different networking technologies**, which are already applied in vessels. On the Edge Server.
2. The need's of **local storage space** for Vessel generated Big Data.
3. And **computing power** for handling and processing the above data with the various computational techniques.

1.5 Structure of Master Thesis.

This Master Thesis consists of six Chapters and three Appendix, which contains the following sort abstracts.:

Chapter 1: Introduction.

In Introductory [chapter 1](#), I refer to the core idea of my research and I associate motivations, technological requirements, and challenges to future developments and applications.

Chapter 2: Vessel performance monitoring.

In [chapter 2](#), I refer briefly to the definition of vessel performance monitoring (VPM), the main areas of interest for VPM, the methods of analysis in VPM, the decisions in VPM, the Shipping KPIs and the Meta data - Ship Attributes..

Chapter 3: Internet of Maritime Things Technologies and Development Tools.

In [chapter 3](#), I present brief references to Definitions of IoT systems and 10 technological concepts. Electronic components, Software, High Performance Computing, IoT protocols-standards, NMEA standards, Signal-K, Navigation Bridge Systems, NDE 4.0, Visualizing Data & Infos and a view in Communications Systems.

Chapter 4: Managing and Intelligent in Data.

In the [chapter 4](#), I refer "abstractly" the definitions About Data (as data), Data Manipulating and the relative Computing Definitions.

Chapter 5: Case studies for Edge Computing.

In the [chapter 5](#), I refer in three case studies of IoT systems for VPM, based to Edge Computing Technologies. That are Generation of IoMaT & NMEA data, NMEA Gateway Server Application and Web-based IoMaT Dashboard.

Chapter 6: Conclusions.

In the [chapter 6](#), entitled "Conclusions", presents a brief evaluation of the contribution of this Master Thesis and lists the points that may be the subject of future research activity.

Appendix A: Software and Hardware Technologies

In the [Appendix A](#), I refer in Software and Hardware Technologies, who I used in Case Studies in Chapter 5.

Appendix B: Relative Organizations and Authorities.

In the [Appendix B](#), I give the main group of Relative Organizations and Authorities with short description and web references for each. Which have a very important role in the Evolution of Maritime Industry and lead to the new era of Shipping 4.0.

Appendix C: Relative Technological Challenges.

In the [Appendix C](#), i refer out of context, into 3 Relative Technological Challenges, in addition.

Appendix D: Bibliography.

In the [Appendix D](#), I include my **Master Thesis Bibliography Resources**, divided by Article's, BSc-MSc Theses & PhD Dissertations, Books, Reports-White Papers and finally the On Line Sources. Classified (sorting) by Document Type, Publication Year, Author Name and Study Title.

Chapter 2

Vessel performance monitoring

In this chapter, I refer briefly to the definition of vessel performance monitoring (VPM), the main areas of interest for VPM, the methods of analysis in VPM, the decisions in VPM, the Shipping KPIs and the Meta data - Ship Attributes.

2.1 Defining of Vessel Performance Monitoring (VPM).

Definition of Vessel Performance Monitoring is: the visualization in real time of all the Maritime /Shipping Key Performance Indicators (KPI's) 2.5 for efficient performance of a merchant vessel, providing financial and environmental benefits for all relevant stakeholders[39, 70].

To archive optimal operational decisions.Had been developed and evaluated many tools. Mainly for reducing fuel consumption and maximizing profit we have to focus in following axis :

1. real time operations
2. maintenance triggers and
3. evaluating technological interventions

Vessel Performance monitoring is also relevant to charter party analysis, vessel bench-marking and performing policy decisions.

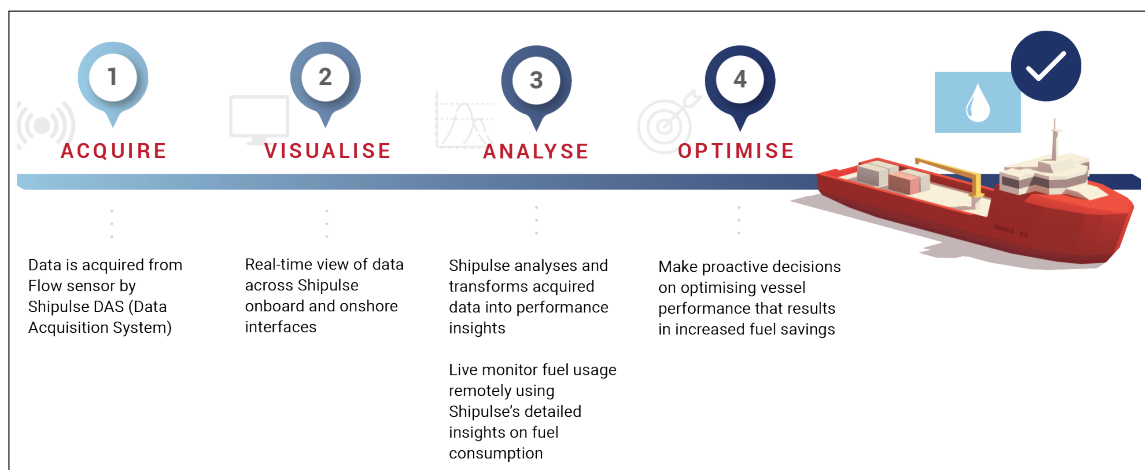


FIGURE 2.1: Overview of VPM , for Fuel Consumption Monitoring (by " Ascenz Solutions Pte Ltd")

The on-board systems in vessels, are complex. And it is common for data modeling and analysis techniques to be employed for extracting useful trends. All datasets and modeling procedures have an inherent uncertainty . And for aiding the decision maker, the uncertainty can be quantified. In order to fully understand the economic risk of a decision an non acceptable risk requires further investments, in data quality and data-analysis techniques. [41, 42].

2.2 Key Areas of Interest in Vessel Performance Monitoring.

2.2.1 Fuel Reduction

Fuel costs represent as much as 50-60% of total ship operating costs, depending on the type of ship and service. Ocean carriers are required to recover these costs to maintain levels of service, meaning the price of shipping goods will continue to face upward pressures.[104]

$$\text{cost} = \text{consumption} * \text{price} * \text{time} \quad (2.1)$$

According to study of the "Marine Digital GmbH"³, there are 5 main factors that affect the fuel consumption of ships:

1. Hull Condition
(Ship Size, Hull Pollution - Damage and Paint)
2. Engine Condition
(Shaft, Engine Fuel Consumption, RPM, Fuel Rate)
3. Trim and Draft
(Draft, Trim, Solidarity/stability)
4. Weather Condition
(Air Temperature, Cloud Cover, Ice Cover, Humidity, Current Speed, Current Direction, Gust Swell Direction, Swell Height, Swell Direction, Visibility, Water Temperature, Sea Level)
5. Vessel Speed
(Min-Max Speed, Average Speed)

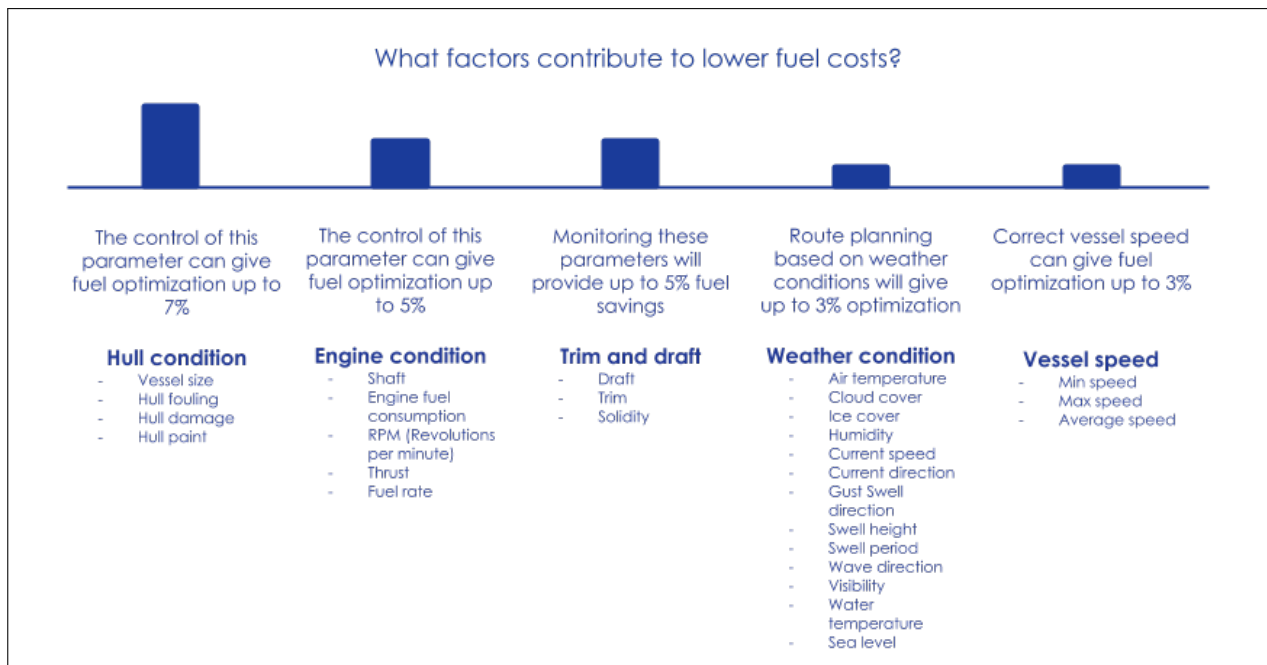


FIGURE 2.2: Main 5 factors for lower fuel costs for Shipping companies
(by "marine-digital[dot]com")

Controlling these parameters, can optimizing fuel consumption from 3 to 7 % , for each parameter. And all these together reducing fuel cumulatively about 23%.[168]

³<https://marine-digital.com/>

Example 1. A vessel, have "consumption=215 tons/day", fuel "price=520 \$/ton"⁴ and we calculate for "time= 280 days" the Yearly Average Cost⁵ is :
 $\text{cost}=215 \times 520 \times 280 = 31,304,000\%$ per year !!!

If this is the average optimal cost for "Fuel Consumption" in Good Conditions of all 5 Main Factors.

We can lose money if one or more of the above factors go into an undesirable state. The loss of money can reach the additional amount of 7,199,920 \$, this is an additional 23 % of the average optimal cost of fuel consumption.

Of course, we can detect in time the degradation of good (desired) state of the factors. If we record and analyze the appropriate performance indicators (PI/KPI /SPI) of the vessel, using techniques based on computational methods (NDE 4.0, BDA, ML, AI, Statistics etc).

2.2.2 Engine Room Department and Non-destructive Evaluation 4.0

The Engine Room Department is responsible for maintaining good condition and performance for the electromechanical equipment of a merchant ship. Whether they are for propulsion, steering, generators, vital systems, or for loading and unloading systems.

Types of Maintenance Procedures⁶

1. Preventive or Scheduled Maintenance System It is famously known as the PMS or Planned Maintenance System. In this type of system the maintenance is carried out as per the running hours like 4000 hrs, 8000 hrs etc., or by the calendar intervals like 6 monthly, yearly etc. of the machinery. The maintenance is carried out irrespective of the condition of the machinery. The parts have to be replaced if it is written in the schedule, even if they can be still used.

2. Corrective or Breakdown Maintenance In this system the maintenance is carried out when the machinery breaks down. This is the reason it is known as the breakdown maintenance. This is not a suitable and good method as situations may occur wherein the machinery is required in emergency. The only advantage of this system is that the working of machinery parts is used to its full life or until it breaks. This system might get costly as during breakdown several other parts may also get damaged.

3. Condition Maintenance system In this system the machinery parts are checked regularly. With the help of sensors etc. the condition of the machinery is accessed regularly and the maintenance is done accordingly. This system requires experience and knowledge as wrong interpretation may damage the machinery and lead to costly repairs which may not be acceptable by the company.

In this case, coming the **Non-Destructive Evaluation Methods**. To assessing the condition of machines with high accuracy. These methods include:

Lock-in Thermography, Pulsed Thermography, DPHM (Diagnostics, Prognostics, Health Management, Prevent, Predict), Structural Health Monitoring and Computational Intelligence Techniques (Machine Learning etc..) [2, 14]

⁴Price updates by <https://shipandbunker.com/prices/#VLSFO>

⁵In the case of this example, a vessel sails assuming 280 days/year at sea (25% in port)

⁶<https://www.marineinsight.com/guidelines/how-maintenance-work-is-done-onboard-a-ship/>



hb

FIGURE 2.3:
Cargo control room of a oil tanker.
<https://www.123rf.com/>

Maintenance Schedule. It is important a good schedule of maintain at the right time, so that repair and maintenance can be done properly and at lower cost in time and money. [60, 61, 82, 83, 100]

2.2.3 Cargo Operations

For all types of merchant ships, with the exception of fishing vessels. For reasons of safety and proper management of the ship's time and cargo space, continuous monitoring, surveillance and supervision of the loading and unloading process is required. This also applies to passenger and car ferries. And for loading processes of consumables (fuel, water, food, spare parts, etc.).

2.2.4 Navigation, Safety - Security Management and e-Navigation.

The Maritime Navigation is defined by the International Maritime Organization (IMO) as:

'process of planning, recording and controlling the movement of the vessel from one port to another'.⁷

The Maritime Safety means:

the safety of life and property at sea, and safety of marine environment from pollution.

Maritime safety is a broad term including everything from ship construction to maintenance to how professional the crews are. It is always the shipping company's overall responsibility to

⁷<https://en.wikipedia.org/wiki/Navigation>

provide optimal conditions and resources for propelling the ship safely at sea.^{8 9 10}

The Maritime Security means the anti-terrorist security of all human activities being realized at sea but especially, the anti terrorist security of the shipping industries, i.e. ships and port facilities from terrorist activities.¹¹

The "Management of the Maritime Safety and Security" means the process of management performed permanently by the Maritime Safety and Security System whose main objective is to ensure the proper level of maritime safety and security of all human maritime activities performed at sea.

The Organization (IMO) defines **e-Navigation** as :

"the harmonized collection, integration, exchange, presentation and analysis of marine information on board and ashore by electronic means to enhance berth to berth navigation and related services for safety and security at sea and protection of the marine environment."

E-navigation is intended to meet present and future user needs of shipping through harmonization of marine navigation systems and supporting shore services. It is expected to provide digital information and infrastructure for the benefit of maritime safety, security and protection of the marine environment, reducing the administrative burden and increasing the efficiency of maritime trade and transport.¹² [11, 13, 16, 19, 20, 25, 27, 46, 65, 84, 114, 118, 122, 127, 138]

2.3 Methods of Analysis in VPM

Shipping Companies as the rest Companies, after passing through an early stage of quantitative management control, concern with qualitative studies of the Key Performance Indicators , i.e. smart measurements and computing-supported evaluations.

A very important task is the accurate evaluation and complete exploitation of the data obtained. [36, 129]

2.3.1 Probabilities and Statistical Analysis.

Probability. Probability is simply how likely something is to happen. Whenever we're unsure about the outcome of an event, we can talk about the probabilities of certain outcomes—how likely they are. The analysis of events governed by probability is called statistics.

Quantitative Analysis. Quantitative analysis is applied to measuring, evaluating performance, valuing a financial instrument and forecasting actual events. Uses mathematical and statistical modelling to draw conclusions based on the numerical values of the measured parameters. Quantitative analysis examines and analyzes past, current and expected future events. Any topic involving numbers can be quantified. Thus, it can be used in both maritime management and navigation.[144]

⁸https://en.wikipedia.org/wiki/Maritime_safety

⁹<https://www.sweship.se/in-english/focal-areas/maritime-safety/>

¹⁰<https://www.imo.org/en/OurWork/Safety/Pages/default.aspx>

¹¹<https://www.imo.org/en/OurWork/Security/Pages/MaritimeSecurity.aspx>

¹²<https://www.imo.org/en/OurWork/Safety/Pages/eNavigation.aspx>

Qualitative Analysis. Qualitative analysis uses subjective judgment to analyze a company's value or prospects based on non-quantifiable information. The qualitative examination and interpretation of observations that aims to discover underlying meanings and patterns of relationships.

Characteristics of qualitative research methods

- Qualitative research methods usually collect data at the sight, where the participants are experiencing issues or problems. These are real-time data and rarely bring the participants out of the geographic locations to collect information.
- Qualitative researchers typically gather multiple forms of data, such as interviews, observations, and documents, rather than rely on a single data source.
- This type of research method works towards solving complex issues by breaking down into meaningful inferences, that is easily readable and understood by all.
- Since it's a more communicative method, people can build their trust on the researcher and the information thus obtained is raw and unadulterated.

Quantitative analysis is not the opposite of qualitative analysis; they are just different philosophies. Used together, they provide useful information for informed decisions that promote a better society, improve financial positions, and enhance business operations [167]

2.4 Decisions in VPM

First of all, we need to determine the definition of "Decision".

Decision: Is the act of determining an action from a number of input info's. Using options and selection logic, defining how the action is determined from the input info's and data.

Seafarers, make several high-stakes decisions every day. Plus, they do this in an environment that is constantly changing and unpredictable, thousands of miles away from land.

With new technology and faster turnaround of ships, seafarers have not only to be fast, but also accurate in making decisions. Of course, we know that, not all the decisions made on the high seas end well. There are on average 100 total losses of ships and over 1000 fatalities each year- most of them attributed to human error. It's easy to get overwhelmed with information which required to make a decision. Or wait for further informations and this can sometimes lead to paralysis by analysis ¹³ [87].

¹³Both paralysis and analysis are Greek words, and the solution- is also a word with Greek root.

2.4.1 Data Driven Decisions

All ways, an Effective Shipping Company puts information's and the relevant decision rights in the same location. In the big data era, information is created and transferred, and expertise is often not where it used to be.

People who understand the problems need to be brought together with the right data. But also with the people who have problem-solving knowledge skills. That can effectively exploiting.



FIGURE 2.4:
Data Driven Image.

The companies who self-characterized themselves as data-driven, have better they performed on objective measures of financial and operational results. Simply, because of big data, managers can measure, and hence know, radically more about their businesses, and directly translate that knowledge into improved decision making and performance.

Prior to the advent of big data, organizations used traditional technologies to analyze large datasets collected from conventional sources such as warehouses, distribution nodes, etc. However, with the advent of big data, it has become easier to perform. Analytics not only consists of large data sets coming from traditional sources, but also takes into account and analyzes unconventional data in real-time nodes, as well as in batch mode. [24]

2.4.2 Empirical and Statistical Decisions

Both for Captains and Engineers, in all layers of maritime universities/academies until today were taught-ed, to make Decisions based in Empirical and Statistical Analysis.

Empirical Decisions.

Expert seafarers have been able to come up with a framework to explain how use their experiences to make their decisions in real-world scenarios. There are three main characteristics of these scenarios:

Dynamics – Every decision made has consequences for the decision made after.

Uncertainty – Information is never perfect in real world environments.

Task sharing – Real world situations are often too complex for one individual to make all the decisions, so decisions must be distributed among-st team members.

Become Decisions Experts. The good news is there are ways to improve how we become experts, which we would encourage trainees and trainers to consider.

These include:

Tactical Decision Games (TDGs) – short paper-and-pencil exercises that describe a situation, a goal and the resources available. TDGs are often presented in small groups, under the supervision of an organizer. At one point, he adds an unexpected and challenging twist that requires a quick

decision. After announcing the unexpected twist, the organizer typically calls on a group member to make a decision with little time to think or analyze, just as they will have to in a real-life situation. It aims to prepare for uncertainty and time pressure, as well as improving communication skills. It also shows individual trainees how other crew members make decisions, which allows for easier exchange of knowledge.

Protocol – this represents good practices, which everyone on the team needs to know. An effective way of learning protocol is through TDGs. In cases of a wrong decision, the organizer “punishes” the group by introducing variations that reflect the consequences of mistakes made in real-life. This allows the decision-makers to understand the reasons for the methods described in the protocol.

On-the-job learning – this is effectively mirroring skilled decision makers as they perform difficult tasks and test out different strategies. A session of on-the-job learning should be followed by a review of the reasons for successes and failures to maximize the learning. In this case, the expert seafarer takes up the role of a mentor for a student.[126]

Statistical Decisions.

Statistical decisions^{14 15} are decisions made on the basis of observations of a phenomenon that obeys probabilistic laws that are not completely known. Statistical decision theory provides a precise mathematical formulation of the concepts pertaining to statistical decisions and to methods of comparing statistical decisions. Divided in to basic branches:[144, 167]

- *Quantitative Statistical Decisions* is a technique that uses mathematical and statistical modeling, measurement, and research to understand behavior. Quantitative analysts represent a given reality in terms of a numerical value. Quantitative analysis is applied to the measurement, performance evaluation, valuation of a financial instrument, and predicting real-world events
- *Qualitative Statistical Decisions* uses subjective judgment to analyze a company’s value or prospects based on non-quantifiable information, such as management expertise, industry cycles, strength of research and development, and labor relations.

2.4.3 Computational Decision.

Computational Decision-making¹⁶ in this context is a process which culminates in the selection of a particular course of action among several alternative possibilities. When decision making is automated using computational techniques such as neural networks and decision trees it is called computational decision making. There are many fields dedicated to trying to find optimal ways of making such decisions including, but not limited to, statistics, operations research, and artificial intelligence. As such, there are a very large number of techniques which could be used each with their own pros and cons [90, 102]

A significant amount of literature has been published covering the applications of computational decision making in fields such as finance, economics, investment management, and trading.

¹⁴<https://encyclopedia2.thefreedictionary.com/Statistical+Decision>

¹⁵<https://learn.g2.com/statistical-analysis>

¹⁶<http://www.turingfinance.com/computational-decision-making-for-finance/>

Computational decision making methods can be broadly split up between five different classes. These include mathematical methods, statistical, logical, search, and hybrid methods which belong to two or more of the previous classes. Mathematical methods such as linear programming can be used to explicitly solve decision making problems when they are posed as systems of equations. Statistical methods rely on statistical inference and probability theory to make probabilistic decision regarding random variables. Logical methods use predicate or fuzzy logic to construct symbolic rules which can be used to make decisions. Search methods such as the Min Max algorithm and A*¹⁷ search through the possible decisions to arrive at the most optimal one. Last, but not least, hybrid methods make use of one or more of the above mentioned techniques.

2.5 Shipping KPIs.

The Shipping KPI System [86] is a global tool for defining, measuring and reporting information on a ship's operational performance in order to:

- **Boost performance** improvements with companies engaged in ship operation activities.
- Provide an efficient **communication platform** on ship operation performance to internal and external stakeholders.

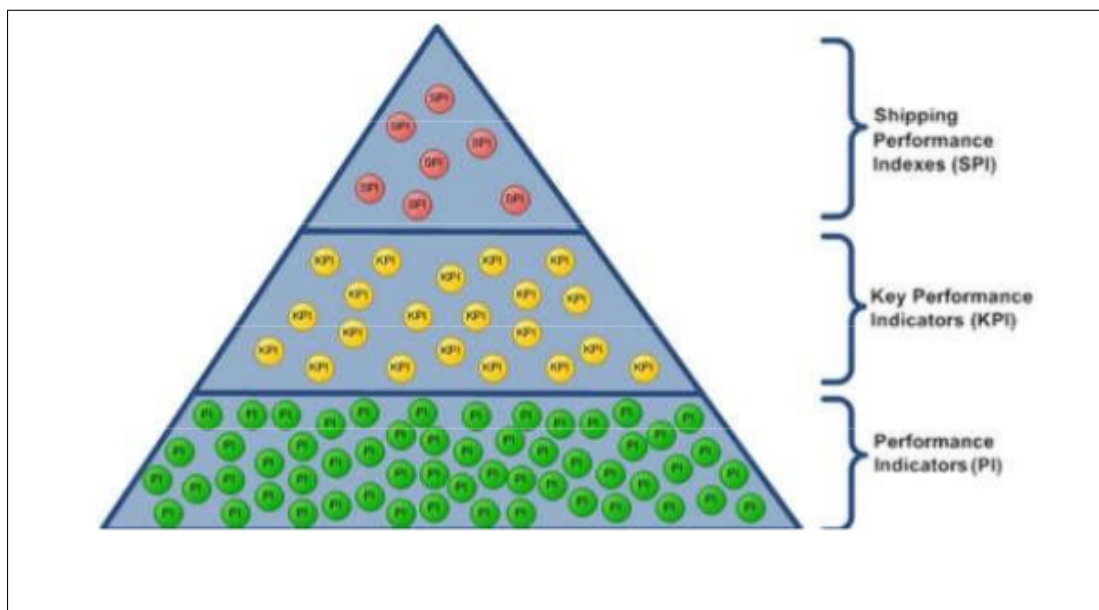


FIGURE 2.5: Shipping Performance Indexes -SPI's (by BIMCO)

Concept The Shipping KPI Standard is built up hierarchical with 7 Shipping Performance Indexes (SPIs) 34 Key Performance Indicators (KPIs) and 66 Performance Indicators (PIs). There is mathematical relation between SPIs which are calculated from KPIs, and KPIs which are calculated from PIs (lowest level).

- Simplifies the way users perceive the KPI Standard.
- Limits the KPI Standard as per the company needs.
- Reduces the data entry volume.

¹⁷A* (pronounced "A-star") is a graph traversal and path search algorithm.

- Customized visual representation of the ship's performance.

A primary KPI is selected by the account manager to be in the KPI Profile. Primary PIs are the PIs, which should be entered in order to calculate the primary KPIs of your KPI profile where primary SPIs are the SPIs to which a primary KPI contributes to. Similarly, a secondary KPI excluded from the KPI profile and secondary PIs should be entered to calculate the secondary KPIs.

Performance Indicators (PIs) PIs are the foundation of the KPI project. They are used to calculate all the KPIs and SPIs. By having a solid understanding of the different PIs, how they are measured and their units measurement, you can ensure that you enter the correct data for your ships, produce accurate reports and correctly benchmark your fleets against the industry.

Key Performance Indicators (KPIs) The Key Performance Indicators (KPIs) are calculated by combining 1 or more PIs. The K ratings will form basis for the SPI score.

Shipping Performance Indicators The Shipping Performance Indicators (SPIs) are calculated by combining 1 or more KPIs. The KPI Standard provides a list with all SPIs along with their formulas, in order for you to have clear understanding of how each of these indicators is calculated and what its rating mean.

Formula to calculate value

$$SPI007 = \frac{KPI006 + KPI012}{2}$$

FIGURE 2.6:
Math Formula to calculate
value of an SPI (by BIMCO)

2.6 Meta data - Ship Attributes

Meta data is used for grouping and filtering during statistical analysis. It usually represents an attribute of the ship, like its length or the nationality of seafarers used during the reporting period. A typical use of meta data allows a ship manager to benchmark his ships not against all other ships in the system, but for example only against ships of the same ship type.^[86]

Descriptive metadata¹⁸ and documentation are critical to maintaining data quality. Metadata is “data about the data” that describes and contextualizes the dataset to ensure it is understandable to future users. Beyond standardized metadata, useful documentation might include standard operations procedures, field notes, etc., from which metadata may be derived or referenced.

Throughout the data lifecycle, both the metadata and documentation must be recorded and updated to reflect the actions taken to the data. This includes collection, acquisition, processing, quality review, and analysis, as well as any other stage of the data lifecycle.^[81]

¹⁸Alternatively, metadata is defined as the data of the identity of an object to be checked-evaluated.

2.6.1 Types of Metadata

There are several types of metadata according to their uses and domain.¹⁹

Technical Metadata This type of metadata defines database system names, tables names, table size, data types, values, and attributes. Further technical metadata also includes some constraints such as foreign key, primary key, and indices.

Business Metadata It consists of the ownership of data, changing policies, business rules and regulations, and other business details. This type of metadata is related to a particular business.

Operational Metadata This type includes the data which is currently under any operation. Besides, it represents the data that is used by executive-level managers to perform any task. Also, this type of metadata can be purged, archived, or activated and can also be migrated.

Descriptive Metadata Descriptive metadata describes any file, folder, book, image, or video. It may include details of data such as title, author, date, size, author name, published on, and similarly others.

Metadata in AIS. In AIS 3.9.5 and LRIT B.12 systems all vessel transmitting within other data, the following metadata (static data)^{20 21} ??:

- Vessel name
- Vessel call sign (if available)
- Maritime Mobile Service Identity (MMSI)
- Draft
- Vessel dimensions

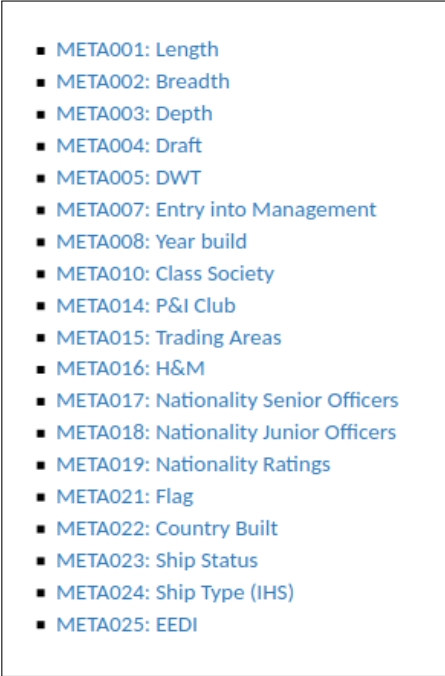
- 
- META001: Length
 - META002: Breadth
 - META003: Depth
 - META004: Draft
 - META005: DWT
 - META007: Entry into Management
 - META008: Year build
 - META010: Class Society
 - META014: P&I Club
 - META015: Trading Areas
 - META016: H&M
 - META017: Nationality Senior Officers
 - META018: Nationality Junior Officers
 - META019: Nationality Ratings
 - META021: Flag
 - META022: Country Built
 - META023: Ship Status
 - META024: Ship Type (IHS)
 - META025: EEDI

FIGURE 2.7:
Vessel SPI Metadata(by BIMCO)

¹⁹<https://www.thecrazyprogrammer.com/2019/12/metadata-in-dbms.html>

²⁰Static data is information about the vessel which must be programmed into the AIS transceiver

²¹The vessel's dynamic data, which includes location, speed over ground (SOG) and course over ground (COG), is calculated automatically using the internal GPS receiver.

```
1 {
2   "MMSI":244750034,
3   "TIME":"2021-07-09 08:06:53 GMT",
4   "LONGITUDE":5.03806,
5   "LATITUDE":52.46015,
6   "COG":360,
7   "SOG":0,
8   "HEADING":211,
9   "ROT":128,
10  "NAVSTAT":8,
11  "IMO":0,
12  "NAME":"PIGASOS II",
13  "CALLSIGN":"PH7002",
14  "TYPE":69,
15  "A":24,
16  "B":6,
17  "C":0,
18  "D":6,
19  "DRAUGHT":4.2,
20  "DEST":"VALENCIA/SPAIN",
21  "ETA":"11-19 21:32"
22 }
```

LISTING 2.1: Sample Vessel Metadata (json file)

Chapter 3

Internet of Maritime Things Technologies and Development Tools.

This chapter, contains brief references to Definitions of IoT systems and 10 technological concepts. Electronic components, Software, High Performance Computing, IoT protocols-standards, NMEA standards, Signal-K, Navigation Bridge Systems, NDE 4.0, Visualizing Data & Info's and a view in Communications Systems.

3.1 Definitions of Internet of Things (IoT).

3.1.1 Internet of Things (IoT).

The Internet of Things (IoT) is the network of technological objects—"things"—that are embedded with sensors, software, communication systems and other technologies for the purpose of exchanging data with other devices and systems, over a local network or via the internet. These devices range from ordinary household objects to sophisticated industrial tools.^{[125] 22 23 24}

3.1.2 Industrial Internet of Things (IIoT).

Industrial IoT (IIoT) refers to the application of IoT technology in industrial ecosystems, especially with respect to instrumentation and control of sensors and devices that engage high performance networking and computing technologies. IIoT is called the fourth wave of the industrial revolution, or Industry 4.0.^{25 26}

3.1.3 Internet of Maritime Things (IoMaT).

The maritime radio-communications connectivity evolution has catapulted the philosophy of Internet of Maritime Things (IoMaT) to the center of advanced digital shipping. Initiating an era of possibilities, similar to Industrial Internet of Things (IIoT). The potential for narrowband connectivity has developed from traditional Inmarsat-type voice applications, to complex networks of M2M devices delivering big data and other value-added services directly to end-users via satellite connectivity (and not only). Technology service providers must take the appropriate combination of communications services. ^{[132] 27 28}

²²<https://www.oracle.com/internet-of-things/what-is-iot/>

²³<https://www.ibm.com/blogs/internet-of-things/what-is-the-iot/>

²⁴<https://docs.microsoft.com/en-us/azure/iot-fundamentals/iot-introduction>

²⁵https://en.wikipedia.org/wiki/Industrial_internet_of_things

²⁶<https://www.ge.com/digital/blog/what-industrial-internet-things-iiot>

²⁷<https://insights.globalspec.com/article/14624/how-the-maritime-industry-is-navigating-the-iot-and-machine-1>

²⁸<https://www.smart-industry.net/iot-and-maritime-industry-a-sea-of-data/>

3.2 Electronic Hardware.

Electronic hardware can range from individual chips/circuits to distributed information processing systems. Well designed electronic hardware is composed of hierarchies of functional modules which inter-communicate via precisely defined interface.

3.2.1 Processing Device's.

A processing device, or "microprocessor," is a small chip that resides in computers and other electronic devices. Its basic job is to receive input and provide the appropriate output. While this may seem like a simple task, modern processors can handle trillions of processing jobs per second.²⁹

Central processing unit (CPU)

A central processing unit (CPU)³⁰, also called a central processor, main processor or just processor, is the electronic circuitry that executes instructions comprising a computer program. The CPU performs basic arithmetic, logic, controlling, and input/output (I/O) operations specified by the instructions in the program. This contrasts with external components such as main memory and I/O circuitry, and specialized processors such as graphics processing units (GPUs).

Multi-core processor A multi-core processor is a computer processor on a single integrated circuit with two or more separate processing units, called cores, each of which reads and executes program instructions. The instructions are ordinary CPU instructions (such as add, move data, and branch) but the single processor can run instructions on separate cores at the same time, increasing overall speed for programs that support multithreading or other parallel computing techniques. Manufacturers typically integrate the cores onto a single integrated circuit die (known as a chip multiprocessor or CMP) or onto multiple dies in a single chip package. The microprocessors currently used in almost all personal computers are multi-core.³¹

Graphics processing unit (GPU)

A graphics processing unit (GPU) is a specialized electronic circuit designed to rapidly manipulate and alter memory to accelerate the creation of images in a frame buffer intended for output to a display device. GPUs are used in embedded systems, mobile phones, personal computers, workstations, and game consoles. Modern GPUs are very efficient at manipulating computer graphics and image processing. Their highly parallel structure makes them more efficient than general-purpose central processing units (CPUs) for algorithms that process large blocks of data in parallel. In a personal computer, a GPU can be present on a video card or embedded on the motherboard.³²

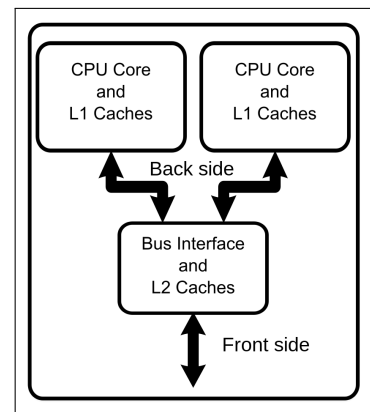


FIGURE 3.1: Diagram of a generic dual-core processor with CPU-local level-1 caches and a shared, on-die level-2 cache (By wikipedia [dot]org)

²⁹<https://techterms.com/definition/processor>

³⁰https://en.wikipedia.org/wiki/Central_processing_unit

³¹https://en.wikipedia.org/wiki/Multi-core_processor

³²In certain CPUs, they are embedded on the CPU die.

This Architecture called "Heterogeneous Multiprocessor System On a Chip"(HMPSoCs).

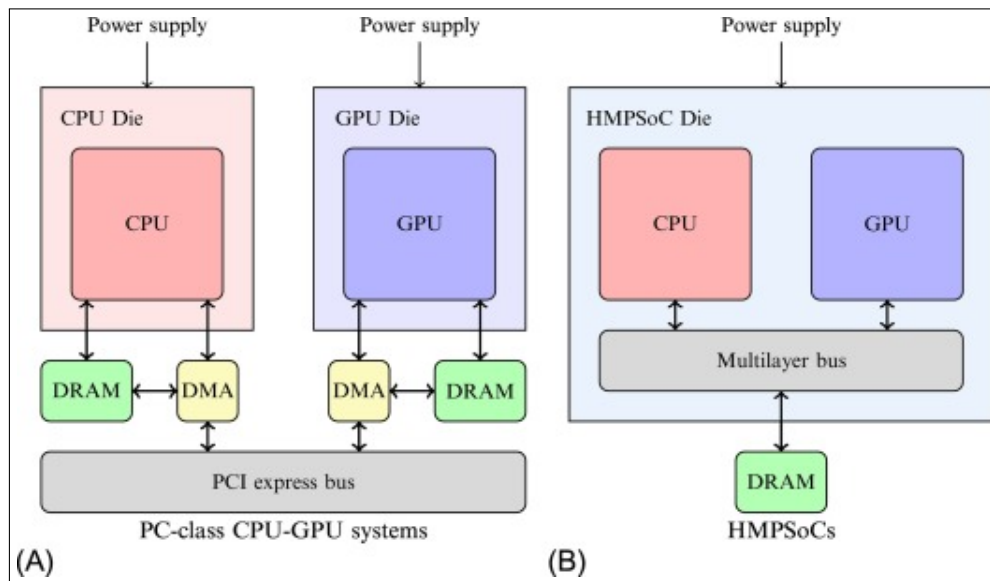


FIGURE 3.2: Abstract architectural block diagrams of PC-class CPU-GPU systems and HMPSoCs. (A) PC-class CPU-GPU systems. (B) HMPSoCs. (by sciencedirect[dot]com)

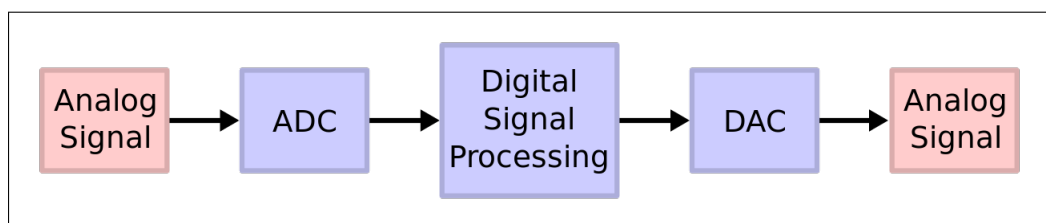


FIGURE 3.3: Block Diagram of Digital Signal Processing System (by wikipedia)

Digital signal processor (DSP)

A digital signal processor (DSP) is a specialized microprocessor chip, with its architecture optimized for the operational needs of digital signal processing. DSPs are fabricated on MOS integrated circuit chips. They are widely used in audio signal processing, telecommunications, digital image processing, radar, sonar and speech recognition systems, and in common consumer electronic devices such as mobile phones, disk drives and high-definition television (HDTV) products.

The goal of a DSP is usually to measure, filter or compress continuous real-world analog signals.³³ Also, dedicated DSPs usually have better power efficiency, thus they are more suitable in portable devices such as mobile phones because of power consumption constraints. DSPs often use special memory architectures that are able to fetch multiple data or instructions at the same time. DSPs often also implement data compression technology, with the discrete cosine transform (DCT) in particular being a widely used compression technology in DSPs.³⁴

³³Most general-purpose microprocessors can also execute digital signal processing algorithms successfully, but may not be able to keep up with such processing continuously in real-time.

³⁴https://en.wikipedia.org/wiki/Digital_signal_processor

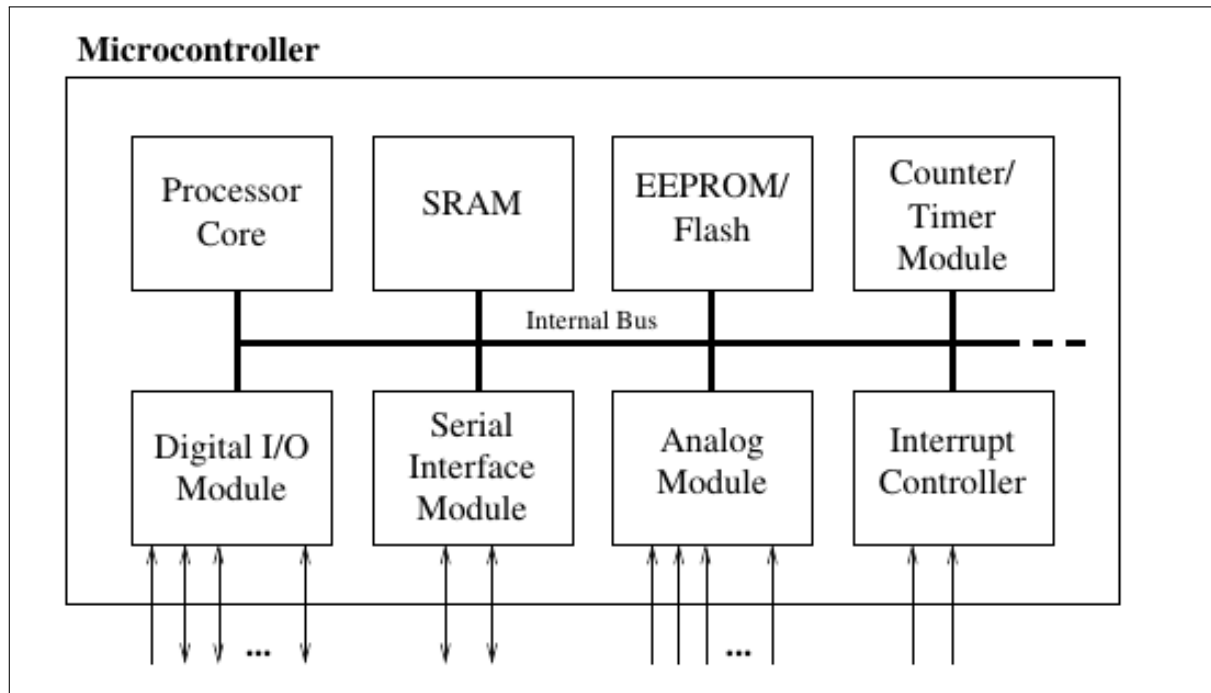


FIGURE 3.4: Basic layout of a microcontroller.

Microcontrollers.

A microcontroller³⁵ is a small computer on a single Integrated Circuit (IC) containing a processor, memory and programmable input/output ports. The program memory, in the form of flash or ROM technologies, is also incorporated on a chip and a small amount of RAM is also included on a single chip. Microcontrollers are specially designed for embedded applications.[32]

Data processing unit (DPU).

A data processing unit (DPU) is a programmable specialized electronic circuit with hardware acceleration of data processing for data-centric computing. The data is transmitted to and from the component as multiplexed packets of information. A DPU generally contains a CPU, NIC³⁶ and programmable data acceleration engines. This allows DPUs to have the generality and programmability of central processing units while being specialized to operate efficiently on networking packets, storage or analytics requests.^{37 38}

The DPU is a new class of programmable processor, a system on a chip (SOC) that combines three elements³⁹:

- An industry standard, high-performance, software programmable, multi-core CPU.
- A high-performance network interface capable of parsing, processing, and efficiently transferring data at network speed.

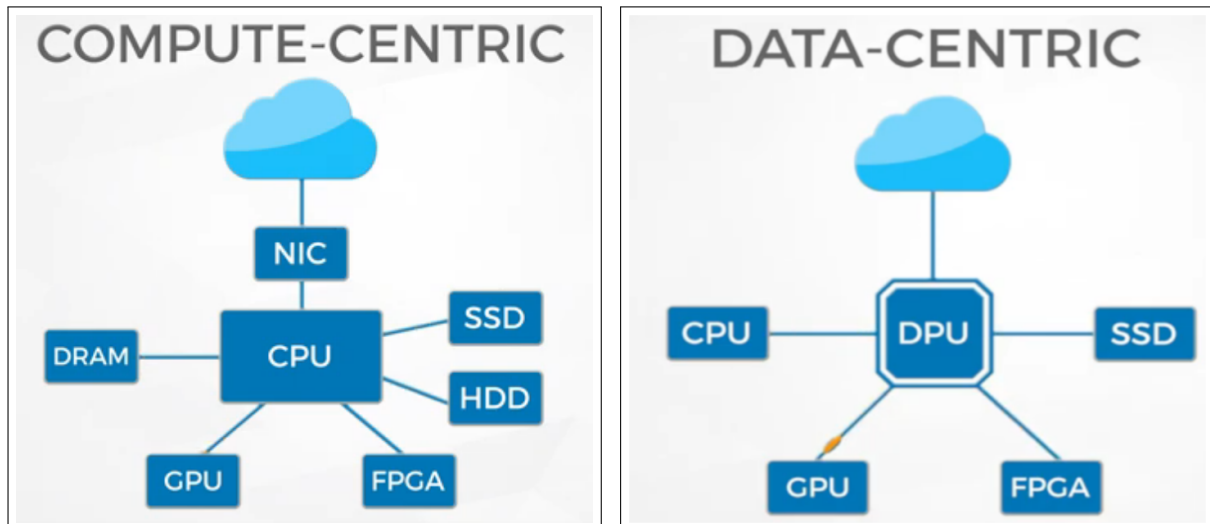
³⁵<https://en.wikipedia.org/wiki/Microcontroller>

³⁶Network Interface Controller

³⁷https://en.wikipedia.org/wiki/Data_processing_unit

³⁸<https://blogs.nvidia.com/blog/2020/05/20/whats-a-dpu-data-processing-unit/>

³⁹<https://www.datacenterdynamics.com/en/opinions/fourth-revolution-and-role-dpu/>



(A) Compute Centric.

(B) Data Centric.

FIGURE 3.5: Comparison Compute VS Data Centric Architectures.
(by www.fungible.com)

- A rich set of flexible and programmable acceleration engines designed to offload networking tasks and optimize application performance for AI and Machine Learning, security, telecommunications and storage etc.

Even with the benefit of GPUs to crunch the data, progress has been held up by the extreme burden it places on the CPUs to access and share information between computers and keep the GPUs fed with data. A key development has been to make the network even faster – but this has overwhelmed the CPU with data management tasks. But now with DPUs we make the network smarter, so the network becomes an active agent in the overall data processing.

DPUs bring the processing closer to the data itself, and they make the network act like a co-processor offloading labour from the central compute engine.

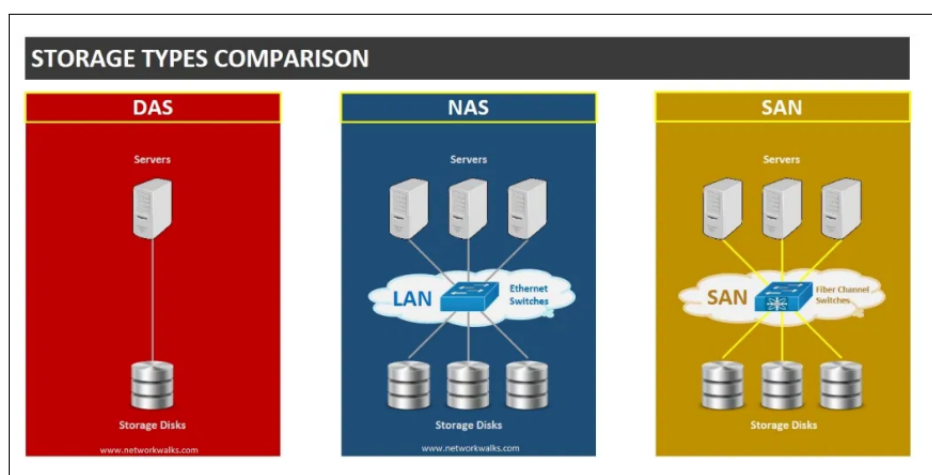


FIGURE 3.6: Storage Types Comparison (by [networkwalks \[dot com\]](http://networkwalks.com)).

3.2.2 Data Storage System at the edge.

While processing at the edge is critical, storage is also essential. However, edge storage presents some interesting problems. **Edge data centers often are ad hoc and are limited on power, cooling and physical space.** With these constraints, the hardware selection may be limited simply because the facility lacks physical space.

These storage systems must be easy and flexible to implement, and preferably work on standard hardware and standard Ethernet networks.

Solutions at the base of the network, improve block storage by making it faster as well as lowering latencies, improving bandwidth and increasing availability for local or distributed file systems. These solutions may also allow for storage and compute to scale independently.[156]

Edge computing architectures typically consists of an array of IoT sensors that are collecting data in realtime which then send data to an edge data center for additional processing. Storage, throughout the edge architecture, needs to provide fast and consistent performance. Each tier of the Edge Storage Architecture (EdStAr) needs SSDs^{40 41 42 43}, but each layer requires different capabilities from the SSD.

At the **top tier** are the IoT sensors. Each sensor is typically capturing data in real-time. The sensors need to store the data they are obtaining at the lowest latency possible so as not to interrupt subsequent captures. A busy sensor can send an almost constant stream of data to its SSD. In most cases, this steam does not require a lot of bandwidth, but it does need consistent latency. SSD vendors need to design the SSD so that latency inducing background functions, like garbage collection, don't negatively impact the write stream.

The **second tier** is the edge data center. These smaller data centers typically act as an offload point for the sensor data as well as performing some processing of data. These devices also need excellent write performance. The IO of a single sensor may not overwhelm them but the aggregate IO of thousands of sensors will. These edge data centers also need to be equipped with SSDs that can handle the high write IO rates of thousands of sensors while again being very consistent. Since the edge data center also processes the data that the IoT devices send them, they also need excellent random IO performance.

The **third tier** is either a private or public cloud data center. This data center will also process data as well as store it long-term. This data center needs SSD than can meet the IO demands of further analysis of data initially collected by the IoT devices. It can also benefit from less expensive SSDs designed for long-term data retention.[112]

Storage Types for Edge Servers.

Storage is the collection of methods and technologies that can capture and hold digital information on media. Storage is normally described as the data storage devices that are connected to the computer through input or output operations that includes flash devices, HDD, SSD, SAN, NAS and other types of medium.[105, 142]

DAS (Directly Attached Storage) This is as simple as it sounds DAS (Directly Attached Storage)⁴⁴ device. A simple example of DAS is an external hard drive connected through a Universal Serial Bus (USB) cable. Basically, USB is much slow for large DAS units. DAS is well suited for a small-to-medium sized business where enough amounts of storage can be configured at a low startup cost. The DAS enclosure can be a separate cabinet that contains the additional disk drives.

⁴⁰https://en.wikipedia.org/wiki/Solid-state_drive

⁴¹<https://www.elinfor.com/knowledge/different-types-of-ssd-interface-p-11190>

⁴²<https://www.10gtek.com/new-1415>

⁴³<https://www.kingston.com/en/community/articledetail/articleid/48543>

⁴⁴https://en.wikipedia.org/wiki/Direct-attached_storage

	DAS	NAS	SAN
Protocols	SATA, SAS	SMB (CIFS), NFS	iSCSI, Fiber Channe
Type	File Storage	Block Storage	Block Storage
Speed	5-10ms	5-10ms	20-50ms
Data Transmission	IDE/SCSI	Ethernet, TCP/IP	FiberChannel
Complexity	Easy	Moderate	Complex
Management Costs	High	Moderate	Low
Storage Type	Sectors	Shared Files	Blocks
Supports Capacity Sharing?	No (possible manually)	Yes	Yes
Connected to Network or not?	No	Yes	Yes
Scalability	Low	Medium	High

FIGURE 3.7: Storage Types Comparison Table by networkwalks.com.

An internal PCI-based RAID controller is typically configured in the server to connect to the storage.

NAS (Network Attached Storage) Every computer that is irrespective of number of disks and the size of storage space available, can be considered a NAS⁴⁵ if it acts as a file server on the network. Another way, a network attached storage (NAS) device is just a computer that shares files over the network. Multiple users and computers can use that resource. The disk arrays in both NAS and DAS are similar in function and operation, meaning you can create similar RAID configurations and partition styles on both. NAS is perfect for Small to Midsize Business's (SMBs) and organizations that need a minimal-maintenance, reliable and flexible storage system that can quickly scale up as needed to accommodate new users or growing data.

SAN (Storage Area Network) SAN stands for Storage Area Network⁴⁶. With SAN we typically see the solutions that are used with medium-to-large size businesses, primarily due to the larger initial investment. SANs require a setup consisting of disk controllers, SAN switches, host bus adapters and fiber cables. The main benefit to a SAN-based storage solution is the ability to share the storage arrays to multiple servers. This allows you to configure the storage capacity as needed, usually by a dedicated SAN administrator. Higher levels of performance throughput are typical in a SAN environment and data is highly available through redundant disk controllers and drives. SAN is typically used in data centers, enterprises or virtual computing environments. It offers the speed of DAS with the sharing, flexibility and reliability of NAS. SAN storage is a very sophisticated option that's meant to support complex, mission-critical applications.

⁴⁵https://en.wikipedia.org/wiki/Network-attached_storage

⁴⁶https://en.wikipedia.org/wiki/Storage_area_network

Computational Storage Architecture.

With the widespread adoption of high-speed storage and the increased performance requirement of data-intensive applications, traditional architectures have created CPU, memory, and storage bottlenecks.⁴⁷

Computational storage is a storage subsystem that includes a number of processors, or CPUs, located on the storage media, or their controllers. These are known as computational storage drives (CSDs), which collectively provide computational storage services.⁴⁸

The idea is to move processing to the data, not data to the processor.

These architectures enable improvements in application performance and/or infrastructure efficiency through the integration of compute resources (outside of the traditional compute & memory architecture) either directly with storage or between the host and the storage. The goal of these architectures is to enable parallel computation and/or to alleviate constraints on existing compute, memory, storage, and I/O.⁴⁹

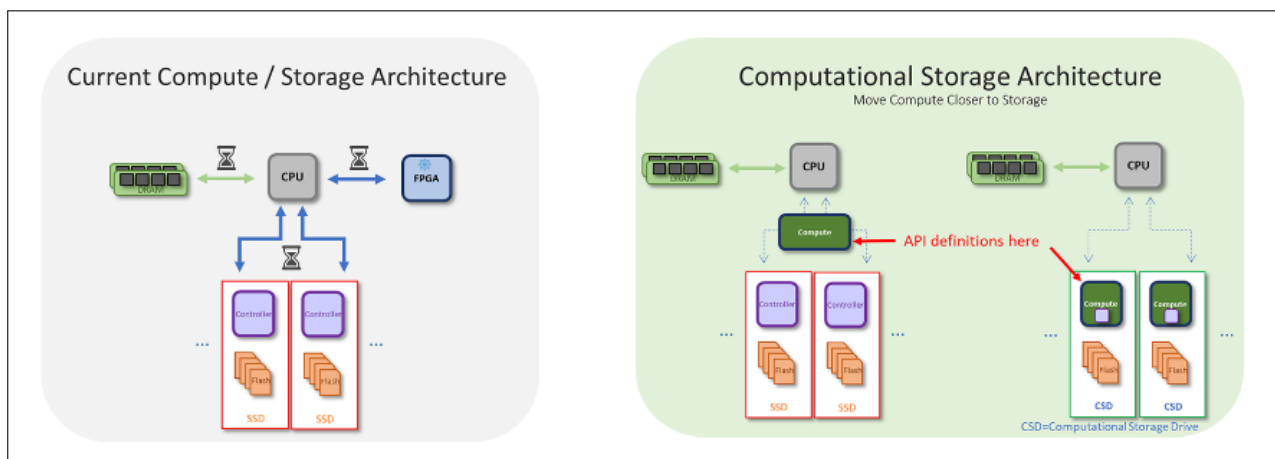


FIGURE 3.8: Classic and Computational Storage (by www.snia.org)

The "Voyage data recorder" System.

The "Voyage data recorder"^{50 51} system or VDR, is a data recording system designed for all vessels required to comply with the IMO's International Convention SOLAS Requirements (IMO Res.A.861(20)) in order to collect data from various sensors on board the vessel. It then digitizes, compresses and stores this information in an externally mounted protective storage unit. The protective storage unit is a tamper-proof unit designed to withstand the extreme shock, impact, pressure and heat, which could be associated with a marine incident (fire, explosion, collision, sinking, etc.).

The protective storage unit may be in a retrievable fixed unit or free float unit (or combined with EPIRB) when the ship sinks in a marine accident.

⁴⁷<https://www.xilinx.com/applications/data-center/computational-storage.html>

⁴⁸<https://www.computerweekly.com/feature/Computational-storage-What-is-it-and-what-are-its-key-use-cases>

⁴⁹<https://www.snia.org/education/what-is-computational-storage>

⁵⁰<https://www.imo.org/en/OurWork/Safety/Pages/VDR.aspx>

⁵¹https://en.wikipedia.org/wiki/Voyage_data_recorder



FIGURE 3.9: A Voyage Data Recorder (VDR), the "black box" of a ship, on the container ship SMA CGM Nabucco by wikipedia.org.

Besides the protective storage unit, the VDR system may consist of a recording control unit and a data acquisition unit, which are connected to various equipment and sensors on board a ship.

Although the primary purpose of the VDR is for accident investigation after the fact, there can be other uses of recorded data for preventive maintenance, performance efficiency monitoring, heavy weather damage analysis, accident avoidance and training purposes to improve safety and reduce running costs.

Simplified voyage data recorder (S-VDR), is a lower cost simplified version VDR for small ships with only basic ship's data recorded.

It depend from regulations for ship type and size, the time for held of recorded data in VDR-SVDR storage media.

Voyage data The information recorded in the unit(s) (sometimes also called the ship's black box) may include the following information:

- Position, date, time using GPS
- Speed log – Speed through water or speed over ground
- Gyro compass – Heading
- Radar* – As displayed or AIS data if no off-the-shelf converter available for the Radar video

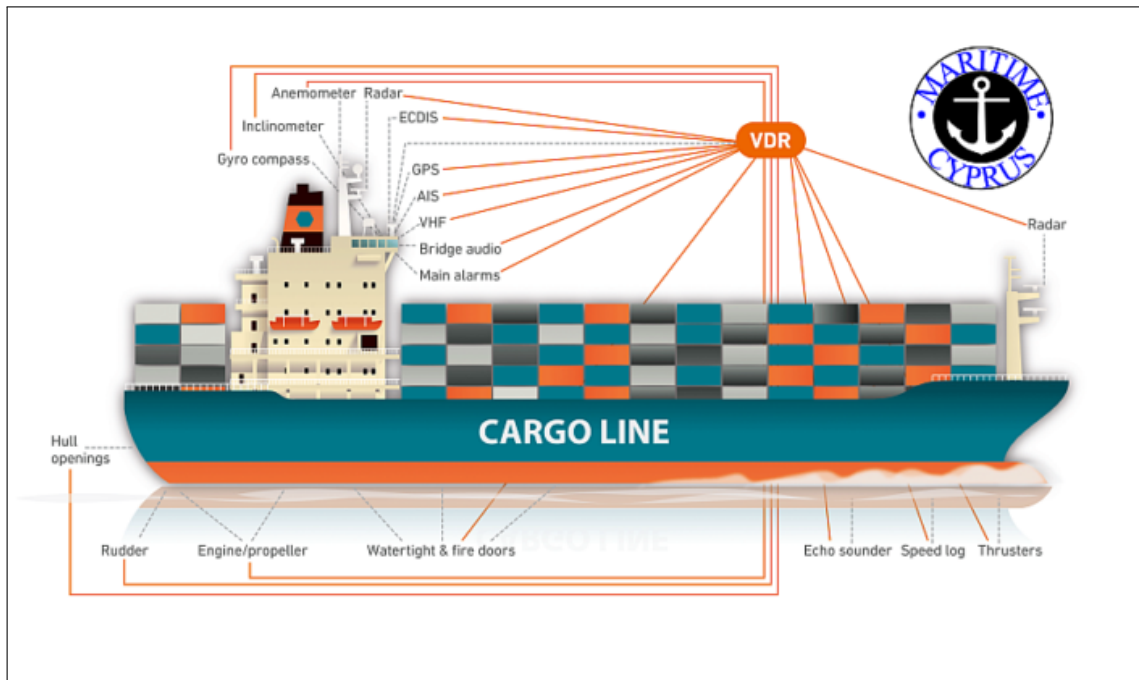


FIGURE 3.10: A Voyage Data Recorder (VDR), general diagram overview.
(by maritimecyprus.com)

- ECDIS* – A screen capture every 15 seconds and a list of navigational charts in use every 10 minutes or when a chart change occurs
- Audio from the bridge, including bridge wings
- VHF radio communications
- Echo sounder* – Depth under keel
- Main alarms* – All IMO mandatory alarms
- Hull openings* – Status of hull doors as indicated on the bridge
- Watertight & fire doors* status as indicated on the bridge
- Hull stress* – Accelerations and hull stresses
- Rudder* – Order and feedback response
- Engine/Propeller* – Order and feedback response
- Thrusters* – Status, direction, amount of thrust % or RPM
- Anemometer and weather vane* – Wind speed and direction

Data marked with * may not be recorded in S-VDR, except Radar and Echo Sounder if data and standard interfaces available.

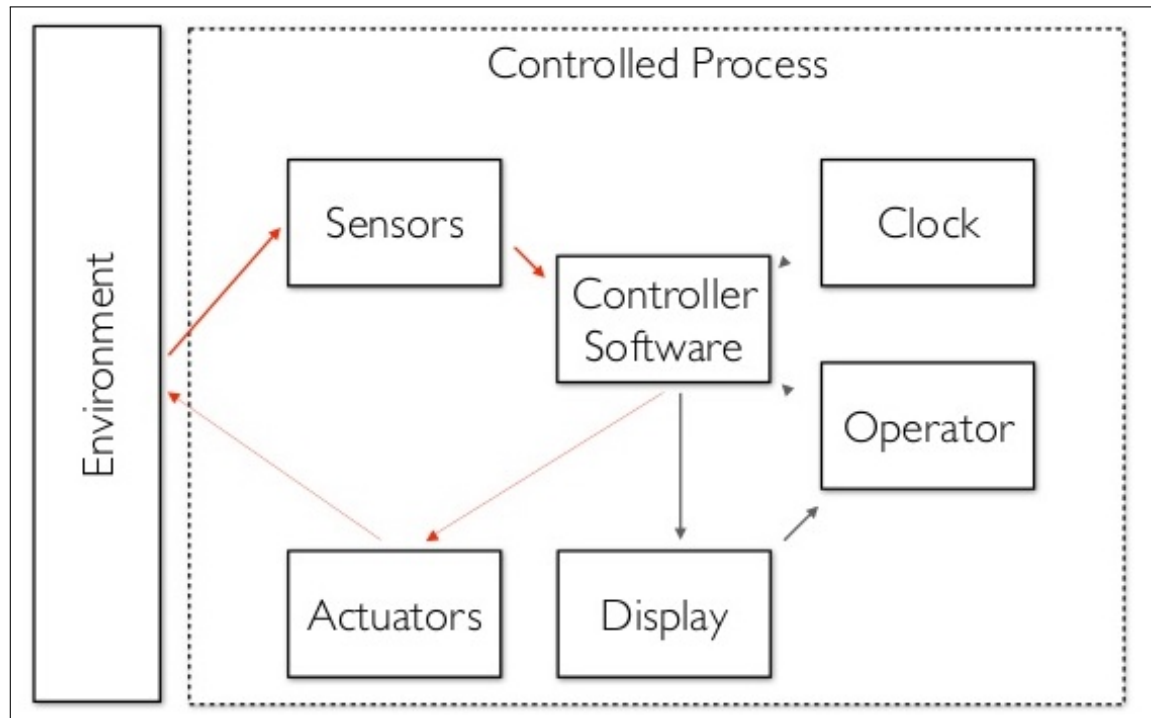


FIGURE 3.11: Embedded Systems Base Diagram (by 'omnisci[dot]com)

3.2.3 Embedded Systems.

An Embedded System^{52 53 54} is a microprocessor-based computer hardware system with software that is designed to perform a dedicated function, either as an independent system or as a part of a large system. At the core is an integrated circuit designed to carry out computation for real-time operations.[32, 45]

Complexities range from a single microcontroller to a suite of processors with connected peripherals and networks; from no user interface to complex graphical user interfaces. The complexity of an embedded system varies significantly depending on the task for which it is designed.

Embedded systems are managed by microcontrollers or digital signal processors (DSP), application - specific integrated circuits (ASIC), field-programmable gate arrays (FPGA), GPU technology, and gate arrays. These processing systems are integrated with components dedicated to handling electric and/or mechanical interfacing.

Embedded systems programming instructions, referred to as firmware, are stored in read-only memory or flash memory chips, running with limited computer hardware resources. Embedded systems connect with the outside world through peripherals, linking input and output devices.

that can capture and hold digital information on media. Storage is normally described as the data storage devices that are connected to the computer through input or output operations that includes flash devices, hard disks, SAN, NAS, tape systems and other different types of medium.

⁵²https://en.wikipedia.org/wiki/Embedded_system

⁵³<https://www.omnisci.com/technical-glossary/embedded-systems>

⁵⁴In computing disciplines, the term "embedded system" is used to refer to an electronic system that is designed to perform a dedicated function and is often embedded within a larger system.

Real-Time Embedded Systems

There are systems that need to respond to a service request within a certain amount of time: they are called real-time systems [45]. To a Real-Time System⁵⁵, each incoming service request imposes a task (job) that is typically associated with a real-time computing constraint, or simply called its timing constraint. The timing constraint of a task is normally specified in terms of its deadline, which is the time instant by which its execution (or service) is required to be completed. Depending on how serious missing a task deadline is, a timing constraint can be either a hard or a soft constraint:

- A timing constraint is hard if the consequence of a missed deadline is fatal. A late response (completion of the requested task) is useless, and sometimes totally unacceptable.
- A timing constraint is soft if the consequence of a missed deadline is undesirable but tolerable. A late response is still useful as long as it is within some acceptable range (say, it occurs occasionally with some acceptably low probability). Actual systems may have both hard and soft timing constraints. A system in which all tasks have soft timing constraints is a soft real-time system. A system is a hard real-time system if its key tasks have hard timing constraint

Actual systems may have both hard and soft timing constraints. A system in which all tasks have software timing constraints is a **soft real-time system**⁵⁶. A system is a **hard real-time system** if its key tasks have hardware timing constraints.

3.2.4 Sensors Systems

In the broadest definition, a sensor is a device, module, machine, or subsystem whose purpose is to detect events or changes in its environment and send the information to other electronics, frequently a computer processor. A sensor is always used with other electronics.

A base sensor , is a sensor that does not include an embedded Microcontroller Processing Unit (MPU). This typically contains just the sensor itself, while all processing and calculation work is done by the manufacturer. By using a base sensor, a manufacturer takes advantage of a lower price point with the component and enables full control of the design process.

A smart sensor , are basic sensing elements with embedded intelligence. The advantage of this is that it can take input from the physical environment and uses built-in compute resources to perform predefined functions upon detection of specific input and then process data before passing it on. As you may already guess, processing of the data captured by the sensor is done on the sensor's IC with the embedded processor. The processed data is then shared to the rest of the device through a serial interface.

At a minimum, a smart sensor is made of a sensor, a microprocessor and communication technology of some kind. Computation resources are provided by the low-power mobile microprocessor. The compute resources must be an integral part of the physical design – a sensor that just sends its data along for remote processing is not considered a smart sensor.

A smart sensor may also include a number of other components besides the primary sensor. These components can include transducers, amplifiers, excitation control, analog filters and compensation. A smart sensor also incorporates software-defined elements that provide functions such as data conversion, digital processing and communication to external devices.[8, 15, 17, 28, 40, 101, 106]

⁵⁵A real-time system is called a real-time embedded system if it is designed to be embedded within some larger system.

⁵⁶More info's at page 37

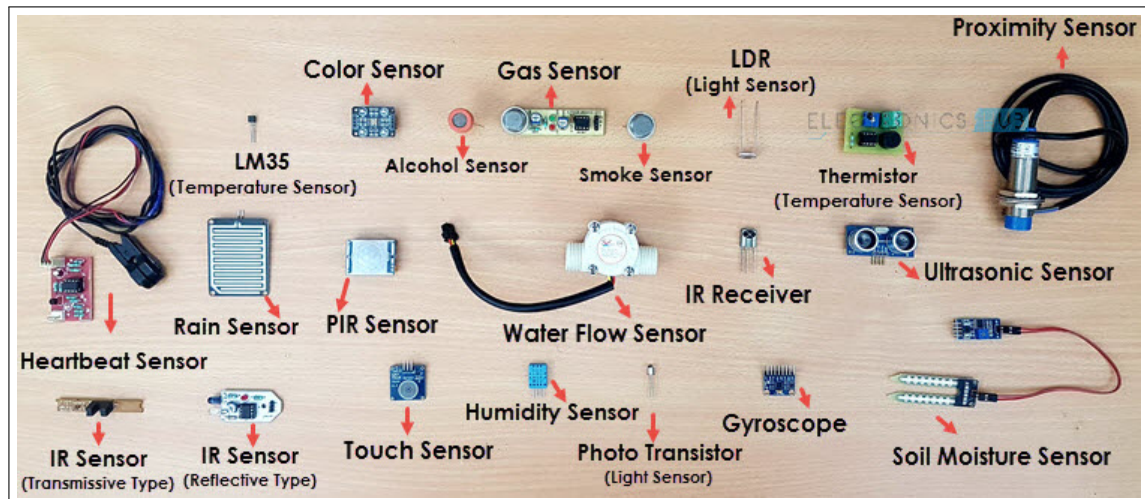


FIGURE 3.12: Types-of-Sensors

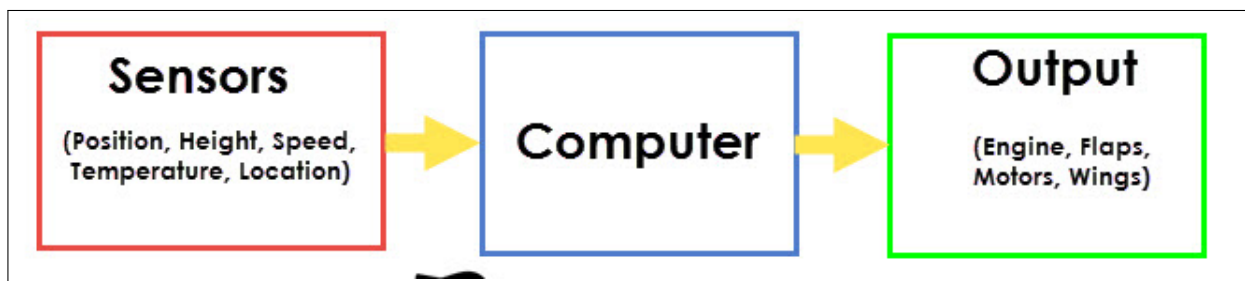


FIGURE 3.13: Smart-Sensors

Sensors for Instrumentation Instrument sensors (and virtual instruments) are electronic devices that convert the magnitude of a physical / chemical phenomenon into an electrical signal. Which signals we use to visualize the previous natural phenomenon. The sensors are listed as follows:⁵⁷

- Inertial
- Pressure
- RPM & Speedometers
- Temperature
- Fluids level.
- etc.

⁵⁷<https://www.vdo-marine.de/216.html?&L=1>

Sensors for Instrumentation	NMEA Sensors
<ul style="list-style-type: none"> ▶ Inertial Sensor ▶ Fluid Level Control ▶ Hydrostatic Head Level Switches ▶ Lever-Type Senders ▶ Ultrasonic Tank Sensors ▶ Reed contact immersion tube level sensor for gasoline and diesel ▶ Pressure Switches ▶ EMPS ▶ EPS ▶ MAP / T-MAP ▶ Murphy PXT-K Series Pressure Transmitters 4-20 mA ▶ Temperature Switches ▶ Temperature Senders ▶ RPM and Speed Senders 	<ul style="list-style-type: none"> ▶ Smart GPS Sensor ▶ Veratron Go ▶ Ultrasonic ▶ VDO GPS Speed Sender ▶ NMEA Sensors for Speed/Depth/Temperature ▶ NMEA Sensor Depth ▶ NMEA Sumlog® - Transducer ▶ NMEA NAV sensor ▶ NMEA Wind sensor

FIGURE 3.14: Marine sensors (by VDO-Marine[dot]de)

NMEA Sensors The NMEA sensors are specialized sensors-devices for measurements that concern exclusively the ship ecosystem. These sensors generate an electronic message based on the NMEA protocol. Which is sent through local area networks to other devices for use in navigation and ship security. Typically sensors referred as:

- GPS
- Speedometer
- Meteorological
- Echo-sounder
- Radar/ARPA.
- AIS
- etc.

[17, 98]

3.2.5 Interconnections with shipbuilding DAQ Systems

The interconnection between the existing DAQ-systems on ships and the news IoT /IoMaT systems is done using the existing Industrial Prototypes and NMEA protocol. Which used by the respective manufacturer of maritime-electronic equipment.

3.3 Software.

Software, is instructions that tell a computer what to do. Software comprises the entire set of programs, procedures, and routines associated with the operation of a computer system. The term was coined to differentiate these instructions from hardware—i.e., the physical components of a computer system. A set of instructions that directs a computer’s hardware to perform a task is called a program, or software program.

The two main types of software are system software and application software. System software controls a computer’s internal functioning, chiefly through an operating system, and also controls such peripherals as monitors, printers, and storage devices. Application software, by contrast, directs the computer to execute commands given by the user and may be said to include any program that processes data for a user. Application software thus includes word processors, spreadsheets, database management, inventory and payroll programs, and many other “applications.” A third software category is that of network software, which coordinates communication between the computers linked in a network.

Software is typically stored on an external long-term memory device, such as a Hard Disk Drive, Solid State Disk or Micro Solid Disk. When the program is in use, the computer reads it from the storage device and temporarily places the instructions in random access memory (RAM). The process of storing and then performing the instructions is called “running,” or “executing,” a program. By contrast, software programs and procedures that are permanently stored in a computer’s memory using a read-only (ROM) technology are called firmware, or “hard software.” [160]

3.3.1 Operating Systems and Firmware.

What is a General Purpose Operating System?

An operating system is a computer program that supports a computer’s basic functions, and provides services to other programs (or applications) that run on the computer. The applications provide the functionality that the user of the computer wants or needs. The services provided by the operating system make writing the applications faster, simpler, and more maintainable. If you are reading this web page, then you are using a web browser (the application program that provides the functionality you are interested in), which will itself be running in an environment provided by an operating system

Embedded OS’s

What is Embedded Operating System? An Embedded System’s Operating System, has limited features VS General Purpose OS’s. It is usually designed for some particular operations to control an electronic device. And run in embedded processors.[158, 159]

The main characteristics of Embedded Operating Systems are as follows

- Direct use of interrupts
- Reactive operation
- Real-time operation
- Streamlined protection mechanisms
- I/O device flexibility
- Configurability

There are two different kinds of operating system, either general purpose operating system that is modified in such a way that it runs on top of a device or the operating system can be custom written. The approaches for the design of operating system include that either we take embedded Operating System that is existing and adapt it to our embedded application or we can design and use a new operating system that is particularly for our Embedded System. We can adapt the existing Operating System to our embedded application by streamline operation, real-time capability and be adding other necessary functions. The advantage of this approach that it has a familiar interface and its disadvantage is that it is not optimized for real-time. The most common examples of embedded operating system around us include Windows Mobile/CE (handheld Personal Data Assistants), Symbian (cell phones) and Linux.

Types of Embedded Operating Systems

Single System Control Loop Single system control loop is the simplest type of embedded operating system. It is so like operating system but it is designed to run the only single task. It still under debate that this system should be classified as a type of operating system or not.

Multi-Tasking Operating System As the name suggests that this operating system can perform multiple tasks. In multi-tasking operating system there are several tasks and processes that execute simultaneously. More than one function can be performed if the system has more than one core or processor. The operating system is switched between tasks. Some tasks wait for events while other receive events and become ready to run. If one is using a multitasking operating system, then software development is simplified because different components of software can be made independent to each other.

Rate Monotonic Operating System It is a type of operating system that ensures that task runs in a system can run for a specific interval of time and for a specific period of time. When it is not ensured, there comes a notification of failure to system software to take suitable action. This time limit cannot be ensured if the system is oversubscribed, at this point another event may occur during run time and the failure notification comes.

Preemptive Operating System A preemptive operating system is a type of multitasking operating system that interprets the preemptive predominance for tasks. A higher priority task is always defined and run before a lower priority task. Such multi-tasking operating systems are efficient in increasing system response to events and also simplify the development of software making the system more reliable. The designer of the system may be able to calculate the time required for the service interprets in a system and also the time is taken by the scheduler for switching tasks. Such systems may fail to meet the deadline of a system and the software is unaware of the missed deadline. CPU loading in a preemptive operating system can be measured naturally by defining a lower priority task that only increments counter and do nothing else.

Real Time Operating System A real-time operating system is the one which serves real time applications. It processes data as it comes in. The time requirements for processing of operating system are usually measured in shorter increments or in 10th of seconds. They may be time sharing or driven by events. Real time Operating systems are used in small embedded systems. The main features of real-time operating system include

- Process threads that can be defined
- Multitasking

- Interrupt levels

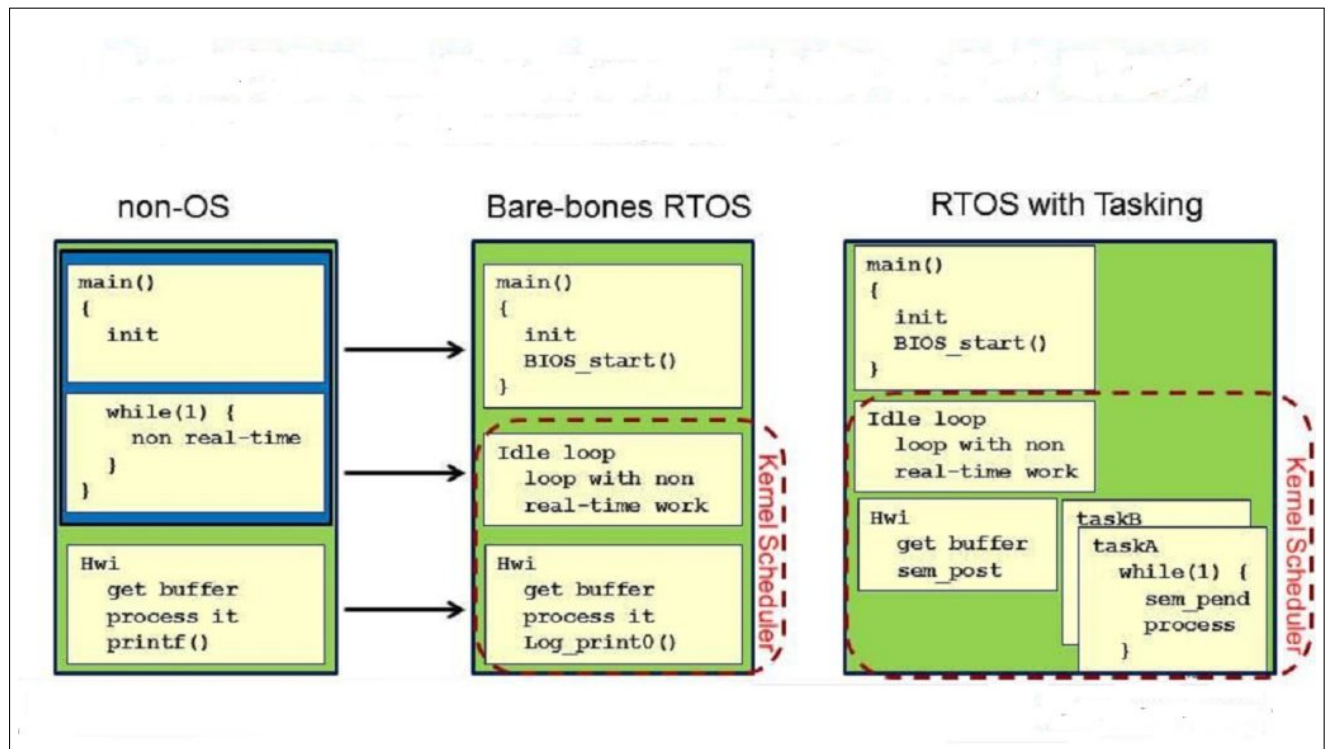


FIGURE 3.15: A comparison between non-RTOS and RTOS executions.
(by "Texas Instruments")

Real-Time Operating System (RTOS)

What is an RTOS? Most operating systems appear to allow multiple programs to execute at the same time. This is called multi-tasking. In reality, each processor core can only be running a single thread of execution at any given point in time. A part of the operating system called the scheduler is responsible for deciding which program to run when, and provides the illusion of simultaneous execution by rapidly switching between each program.

The type of an operating system is defined by how the scheduler decides which program to run when. For example, the scheduler used in a multi user operating system (such as Unix) will ensure each user gets a fair amount of the processing time. As another example, the scheduler in a desk top operating system (such as Windows) will try and ensure the computer remains responsive to its user.⁵⁸

The scheduler in a Real Time Operating System (RTOS) is designed to provide a predictable (normally described as deterministic) execution pattern. This is particularly of interest to embedded systems as embedded systems often have real time requirements. A real time requirements is one that specifies that the embedded system must respond to a certain event within a strictly defined time (the deadline). A guarantee to meet real time requirements can only be made if the behaviour of the operating system's scheduler can be predicted (and is therefore deterministic).

Traditional real time schedulers, such as the scheduler used in FreeRTOS, achieve determinism by allowing the user to assign a priority to each thread of execution. The scheduler then uses the priority to know which thread of execution to run next. In FreeRTOS, a thread of execution is called a task.

⁵⁸RTOS is not a big operating system, and isn't designed to run on a desktop computer class processor.

RTOS in the IoT era, The Internet of Things (IoT) bandwagon is accelerating the use of third-party software stacks, especially the TCP/IP and USB stacks that few developers want to write. And these third-party stacks and tools are often compatible with various RTOS solutions.

Moreover, popular RTOS solutions like Azure RTOS, Amazon FreeRTOS, and Segger's em-bOS are being qualified for MCUs from large suppliers. This out-of-box integration is crucial in managing software complexity and lowering the barriers to entry for smaller embedded design outfits.

However, the use of RTOS also increases the overall system complexity and may introduce new types of bugs. So, developers must have adequate knowledge of how to implement RTOS-based programs effectively.

[66, 121, 155, 161]

Firmware

In Computing, **Firmware**⁵⁹ is a specific class of computer software that provides the low-level control for a device's specific hardware. Firmware, such as the BIOS of a personal computer, may contain only elementary basic functions of a device and may only provide services to higher-level software. For less complex devices, firmware act as the device's complete operating system, performing all control, monitoring and data manipulation functions. Typical examples of devices containing firmware are embedded systems, consumer appliances, computers, and computer peripherals.

Firmware is held in non-volatile memory devices such as ROM, EPROM, EEPROM, and Flash memory. Updating firmware requires ROM integrated circuits to be physically replaced, or EPROM or flash memory to be reprogrammed through a special procedure. Some firmware memory devices are permanently installed and cannot be changed after manufacture. Common reasons for updating firmware include fixing bugs or adding features to the device

BIOS

In computing, **BIOS is Firmware**⁶⁰ used to perform hardware initialization during the booting process (power-on startup), and to provide runtime services for operating systems and programs. The BIOS in modern PCs initializes and tests the system hardware components, and loads a boot loader from a mass storage device which then initializes an operating system. In the era of DOS, the BIOS provided BIOS interrupt calls for the keyboard, display, storage, and other input/output (I/O) devices that standardized an interface to application programs and the operating system. More recent operating systems do not use the BIOS interrupt calls after startup.

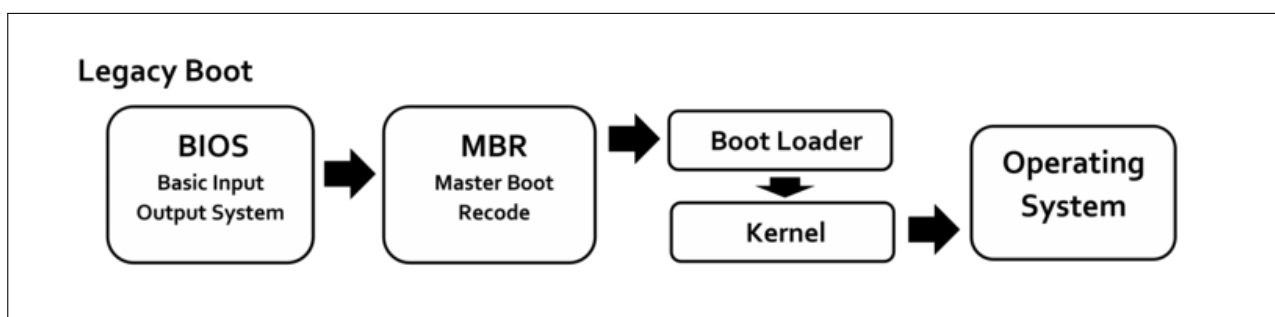


FIGURE 3.16: Legacy BIOS boot process.(From Wikimedia Commons)

⁵⁹<https://en.wikipedia.org/wiki/Firmware>

⁶⁰<https://en.wikipedia.org/wiki/BIOS>

3.3.2 Programming Languages

In Industry 4.0 and Maritime 4.0, the IoT (IoMaT) is the next big thing. Developers and techies, arming themselves with programming skills needed to meet the demands of new era[67, 152]. The IoT design architecture is divided in three vital sections, are:

1. the data created by sensors,
2. this organizes local gateways or hubs
3. and the geographically distance where the data are collected by centralized servers.

Choice of Influences In a given business case there are many factors which determine what language is better to choose. Here are the some important things that are taken into consideration:

- Hardware
- Speed and cost of development
- Developer tools
- Already mainstream and firmly established in the Software Development industry.
- Top-ranked in the renowned programming languages ranking websites.
- Popularity is not decreasing but it is stable or increasing.
- This consists of sensors and actuators are portable connected end devices where the “field-work” requires all the firmware to gather metrics huge library sets, schemes, tools support, and community groups.
- Good salary incomes are demanded in the job market.

Need of Firmware Sensors and actuators are portable connected end devices where the “field-work” requires all the firmware to gather metrics, perform some simple actions to turn on/off and in order, they belong to finite memory capacity with low power computation. The leading programming languages used in IoT devices are low leveled languages such as C or C++. Here the optimized compact scripts are generated while writing on C/C++ and the firmware runs only on RAM. However, the IoT hardware layer makes use of micro-controllers for compatible C/C++ languages. Simultaneously, we need good knowledge skills in programming languages to write best, quality and excellent codes.

The Maritime Edge Computing The Data Servers and Gateways in Maritime Edge Systems for IoMaT on the vessel, have more complex functionality with several programming languages running on this devices. This edge layer tool performs various tasks like performance monitoring, automation, local intelligence, manages multiple data streams, routing data from the cloud and uses machine learning trained models for numerous processes.

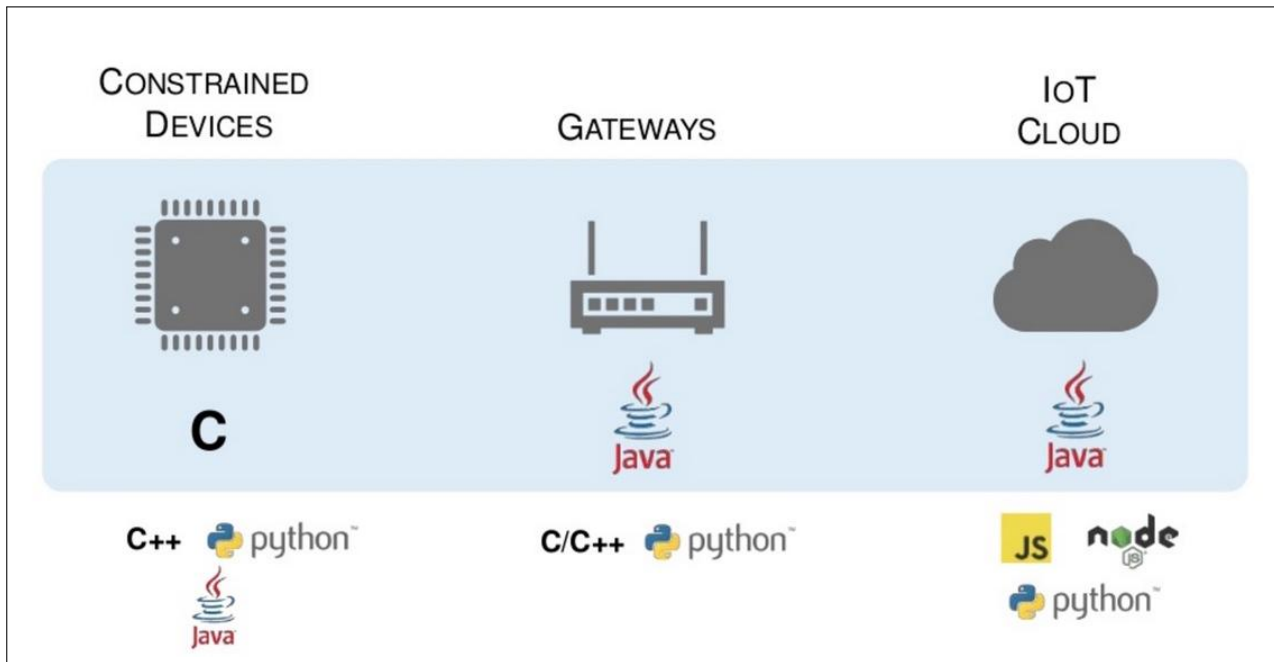


FIGURE 3.17: Top Programming Languages for IoT.(by "iotcentral[dot]io")

Programming Languages for IoT Projects

Assembly In spite of the fact that it's not the most famous programming language on the run-down, Assembly is an incredible choice in the event that you need to keep your IoT applications smaller. It's a low-level programming language, so don't hope to do much with it as its abilities are insignificant.

C/C++ C it's essentially a beginning stage and is the most well-known language for installed gadgets. C has been utilized with IoT sheets, like Arduino, and it is utilized frequently, despite the fact that different languages may rank significantly higher. C doesn't have the preparing intensity of a question arranged pre-processor like C++. Therefore, it's utilized as a pre-processor for C to empower it to run a larger amount of languages. It's easy to commit heaps of errors with this programming language, yet it's most loved by software engineers. Since the most widely-recognized Linux extends as the installed programming language, it empowers layers of items and reflections. It's optimal for engineers hoping to expand their programming code for IoT and implant code.

Further, C++ causes you to utilize different languages including C#, Java, and Python.

Python 3.x A couple of years prior, nobody figured Python would be utilized for IoT for the most central part around web applications. Be that as it may, that has changed recently as it's a simple programming language to comprehend and use in IoT ventures. Despite the fact that Python began as a scripting language to stick code together, it has become one of the essential languages and is utilized by plenty of engineers. As little gadgets have restricted computational power and memory, developers need to get inventive to make life simple. Subsequently, it has developed significantly inside inserted gadgets' space while empowering developers to make applications that can convey understandable information mining results. Nowadays, the vast majority of the well-known microcontrollers are likewise using Python. For instance, there are even little forms like the **MicroPython** board [A.4](#) , which is just a couple of square inches and a programming

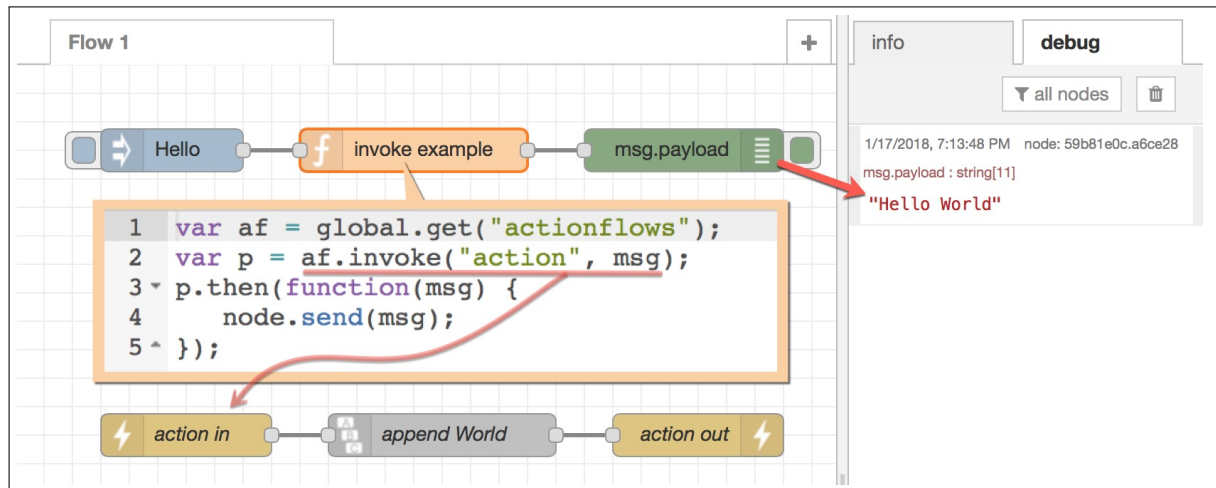


FIGURE 3.18: Node-Red action flows and Java Script.(by "Node Red Organization.")

bundle. In the event that you need to create something cool for Alexa⁶¹, you better catch up on your Python programming aptitudes.

Java. Java is the well-known programming languages used by the experts. They consider it is the best choice for IoT as it is known for write once, run anywhere. Developers can easily produce and debug code on their computer. It can be transferred to any chip using Java Virtual Machine. As a result, it can be run on places where Java Virtual Machines (JVMs) are used and on any other machine as well.

JavaScript and Node JS. All HTML and Internet browsers today utilize **JavaScript** as their programming language. In spite of the fact that it has taken odds and ends from different languages, like Python and C, you can state that it's a scripting language shares other programming language libraries, like Java. This goes far in making gadgets interoperable and its broad use in present programming helps make things less demanding. The famous branch in IoT advancement has been **Node.js**, a significant part of the work is centered around centers and servers to accumulate information and store it. Two microcontrollers that run JavaScript from the earliest starting point are Espruino and Tessel. JavaScript is ubiquitous in web applications and sites, and now, web developers can undoubtedly proceed onward to IoT advancement without taking up another language.

Node-RED Node-RED^{62 63} is a flow-based development tool for visual programming developed originally by IBM for wiring together hardware devices, APIs and online services as part of the Internet of Things.

Node-RED provides a web browser-based flow editor, which can be used to create JavaScript functions. Elements of applications can be saved or shared for re-use. The runtime is built on Node.js. The flows created in Node-RED are stored using JSON. Since version 0.14, MQTT nodes can make properly configured TLS connections.

In 2016, IBM contributed Node-RED as an open source JS Foundation project.[37]

⁶¹Alexa is Amazon's voice AI. Alexa lives in the cloud and is happy to help anywhere there's internet access and a device that can connect to Alexa. <https://www.amazon.com/b?node=21576558011>

⁶²<https://en.wikipedia.org/wiki/Node-RED>

⁶³<https://nodered.org/about/>



FIGURE 3.19: Official Logo of the Rust Programming Language.

C# (C Sharp). Already, the IoT space is dominated by the usual popular programming languages, but there are many more used by developers to create interesting and smart things. The internet of things usually is a polyglot effort, which will not see a heavy reliance on one language.

A lot of things would not truly exist without one of the most important languages around, which is C#. Basically, it's a starting point and the most popular language for embedded devices. It has been used with IoT boards, and most often used, even though other programming languages may rank much higher.

Rust. The Rust Programming Language^{64 65 66}, is blazingly fast and memory-efficient: with no runtime or garbage collector, it can power performance-critical services, run on embedded devices, and easily integrate with other languages [43].

Rust's rich type system and ownership model guarantee memory-safety and thread-safety — enabling to eliminate many classes of bugs at compile-time.

Rust is using Rust which means that all the standard compiler libraries are written in rust; there is a bit of use of the C programming language but most of it is Rust.

Rust has been noted for its growth as a newer language and has been the subject of academic programming languages research.

⁶⁴<https://www.rust-lang.org/>

⁶⁵<https://rustwiki.org/en/book/>

⁶⁶[https://en.wikipedia.org/wiki/Rust_\(programming_language\)](https://en.wikipedia.org/wiki/Rust_(programming_language))

3.4 Summary of High Performance Computing Technologies.

High performance computing (HPC)⁶⁷ is the ability to process data and perform complex calculations at high speeds. One of the best-known types of HPC solutions is the **Super Computing**. A supercomputer contains thousands of compute nodes that work together to complete one or more tasks. This is called parallel processing. It's similar to having thousands of PCs networked together, combining compute power to complete tasks faster.

3.4.1 Classes of parallel computing.

Parallel computing can be roughly classified according to the level at which the hardware supports parallelism. This classification is broadly analogous to the distance between basic computing nodes. These are not mutually exclusive; for example, clusters of symmetric multiprocessors are relatively common.

Multi-core computing A multi-core processor is a processor that includes multiple processing units (called "cores") on the same chip. This processor differs from a super-scalar processor, which includes multiple execution units and can issue multiple instructions per clock cycle from one instruction stream (thread); in contrast, a multi-core processor can issue multiple instructions per clock cycle from multiple instruction streams. Each core in a multi-core processor can potentially be super-scalar as well—that is, on every clock cycle, each core can issue multiple instructions from one thread.

Simultaneous multithreading (of which Intel's Hyper-Threading is the best known) was an early form of pseudo-multi-coreism. A processor capable of concurrent multithreading includes multiple execution units in the same processing unit—that is it has a super-scalar architecture—and can issue multiple instructions per clock cycle from multiple threads. Temporal multithreading on the other hand includes a single execution unit in the same processing unit and can issue one instruction at a time from multiple threads

Cluster computing A cluster is a group of loosely coupled computers that work together closely, so that in some respects they can be regarded as a single computer⁶⁸. Clusters are composed of multiple standalone machines connected by a network. While machines in a cluster do not have to be symmetric, load balancing is more difficult if they are not. The most common type of cluster is the Beowulf cluster, which is a cluster implemented on multiple identical commercial off-the-shelf computers connected with a TCP/IP Ethernet local area network.[44] Beowulf technology was originally developed by Thomas Sterling and Donald Becker. 87% of all Top500 supercomputers are clusters.⁶⁹

Because grid computing systems (described below) can easily handle embarrassingly parallel problems, modern clusters are typically designed to handle more difficult problems—problems that require nodes to share intermediate results with each other more often. This requires a high bandwidth and, more importantly, a low-latency interconnection network. Many historic and current supercomputers use customized high-performance network hardware specifically designed for cluster computing, such as the Cray Gemini network[22].

⁶⁷Commonly known as "Super Computing".

⁶⁸https://en.wikipedia.org/wiki/Computer_cluster

⁶⁹<https://www.top500.org/statistics/list/>

Basics components of a cluster:

- Multiple computers
- Connected over a network
- With a shared file system
- Supports running a parallel application across several computers

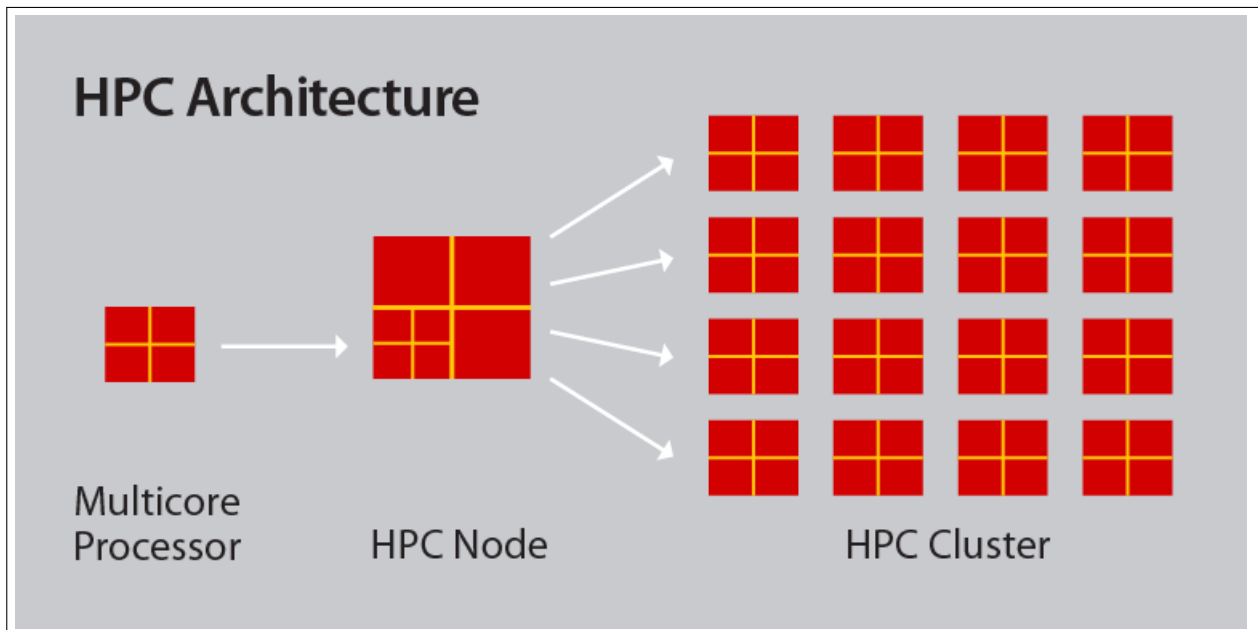


FIGURE 3.20: High Performance Computing Architecture.
(by "conocophillips[dot]com")

Grid computing Grid computing is the most distributed form of parallel computing. It makes use of computers communicating over the Internet to work on a given problem. Because of the low bandwidth and extremely high latency available on the Internet, distributed computing typically deals only with embarrassingly parallel problems. Many distributed computing applications have been created, of which SETI@home⁷⁰ is the best-known example.

Most grid computing applications use middleware⁷¹. The most common distributed computing middleware is the Berkeley Open Infrastructure for Network Computing (BOINC)⁷². Often, distributed computing software makes use of "spare cycles", performing computations at times when a computer is idling.

⁷⁰<https://en.wikipedia.org/wiki/SETI@home>

⁷¹<https://en.wikipedia.org/wiki/Middleware>

⁷²<https://boinc.berkeley.edu/>

General-purpose computing on graphics processing units (GPGPU) General-purpose computing on graphics processing units (GPGPU) is a fairly recent trend in computer engineering research. GPUs are co-processors that have been heavily optimized for computer graphics processing [12]. Computer graphics processing is a field dominated by data parallel operations—particularly linear algebra matrix operations.

In the early days, GPGPU programs used the normal graphics APIs for executing programs. However, several new programming languages and platforms have been built to do general purpose computation on GPUs with both **Nvidia** and **AMD** releasing programming environments with **CUDA** and **Stream SDK** respectively. Nvidia has also released specific products for computation in their Tesla series. The technology consortium Khronos Group has released the OpenCL specification, which is a framework for writing programs that execute across platforms consisting of CPUs and GPUs. AMD, Apple, Intel, Nvidia and others are supporting OpenCL.

3.4.2 Parallel Programming Languages, APIs.

Concurrent programming languages, libraries, APIs, and parallel programming models⁷³ have been created for programming parallel computing systems. These can generally be divided into classes based on the assumptions they make about the underlying memory architecture—shared memory, distributed memory, or shared distributed memory. Shared memory programming languages communicate by manipulating shared memory variables. Distributed memory uses message passing. POSIX Threads and OpenMP are two of the most widely used shared memory APIs, whereas Message Passing Interface (MPI) is the most widely used message-passing system API. One concept used in programming parallel programs is the future concept, where one part of a program promises to deliver a required datum to another part of a program at some future time.

CAPS enterprise and Pathscale are also coordinating their effort to make Hybrid Multi-core Parallel Programming (HMPP) directives an open standard called OpenHMPP. The OpenHMPP directive-based programming model offers a syntax to efficiently offload computations on hardware accelerators and to optimize data movement to/from the hardware memory. OpenHMPP directives describe Remote Procedure Call (RPC) on an accelerator device (e.g. GPU, Cluster) or more generally a set of cores. The directives annotate C or Fortran codes to describe two sets of functionalities: the offloading of procedures (denoted codelets) onto a remote device and the optimization of data transfers between the CPU main memory and the accelerator memory.

The rise of consumer GPUs has led to support for compute kernels, either in graphics APIs (referred to as compute shaders), in dedicated APIs (such as OpenCL), or in other language extensions.

CUDA (an acronym for Compute Unified Device Architecture)⁷⁴ is a parallel computing platform and application programming interface (API) model created by Nvidia. It allows software developers and software engineers to use a CUDA-enabled graphics processing unit (GPU) for general purpose processing – an approach termed GPGPU (general-purpose computing on graphics processing units). The CUDA platform is a software layer that gives direct access to the GPU's virtual instruction set and parallel computational elements, for the execution of compute kernels.

⁷³For example an parallel algorithmic skeleton[3]

⁷⁴<https://en.wikipedia.org/wiki/CUDA>

CUDA C/C++ is just one of the ways you can create massively parallel applications with CUDA. It lets you use the powerful C++ programming language to develop high performance algorithms accelerated by thousands of parallel threads⁷⁵ running on GPUs. Many developers have accelerated their computation- and bandwidth-hungry applications this way, including the libraries and frameworks that underpin the ongoing revolution in artificial intelligence known as Deep Learning.

So, you've heard about CUDA and you are interested in learning how to use it in your own applications. To follow along, you'll need a computer with an CUDA-capable GPU (Windows, Mac, or Linux, and any NVIDIA GPU should do), or a cloud instance with GPUs (AWS, Azure, IBM SoftLayer, and other cloud service providers have them). You'll also need the free CUDA Toolkit installed.[75]

CUDA® Python is a preview software release providing Cython/Python wrappers for CUDA driver and runtime APIs. Python developers will be able to leverage massively parallel GPU computing to achieve faster results and accuracy.

Python is an important programming language that plays a critical role within the science, engineering, data analytics, and deep learning application ecosystem. [171]

C++ Accelerated Massive Parallelism (C++ AMP)⁷⁶ accelerates the execution of your C++ code by taking advantage of the data-parallel hardware that's commonly present as a graphics processing unit (GPU) on a discrete graphics card. The C++ AMP programming model includes support for multidimensional arrays, indexing, memory transfer, and tiling. It also includes a mathematical function library. You can use C++ AMP language extensions to control how data is moved from the CPU to the GPU and back. [53]

Open Computing Language (OpenCL™) is an open, royalty-free standard for cross-platform, parallel programming of diverse accelerators found in supercomputers, cloud servers, personal computers, mobile devices and embedded platforms. OpenCL greatly improves the speed and responsiveness of a wide spectrum of applications in numerous market categories including professional creative tools, scientific and medical software, vision processing, and neural network training and inferencing.

OpenCL 3.0 realigns the OpenCL roadmap to enable developer-requested functionality to be broadly deployed by hardware vendors, and it significantly increases deployment flexibility by empowering conformant OpenCL implementations to focus on functionality relevant to their target markets. OpenCL 3.0 also integrates subgroup functionality into the core specification,

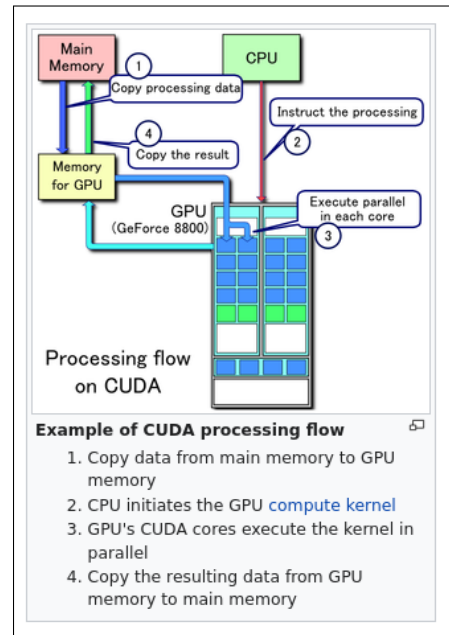


FIGURE 3.21: Cuda Processing flow. (By "<https://en.wikipedia.org/wiki/CUDA>")

⁷⁵This threads, generated and killed by a function calling Cuda - Kernel.

⁷⁶<https://en.wikipedia.org/wiki/C++AMP>

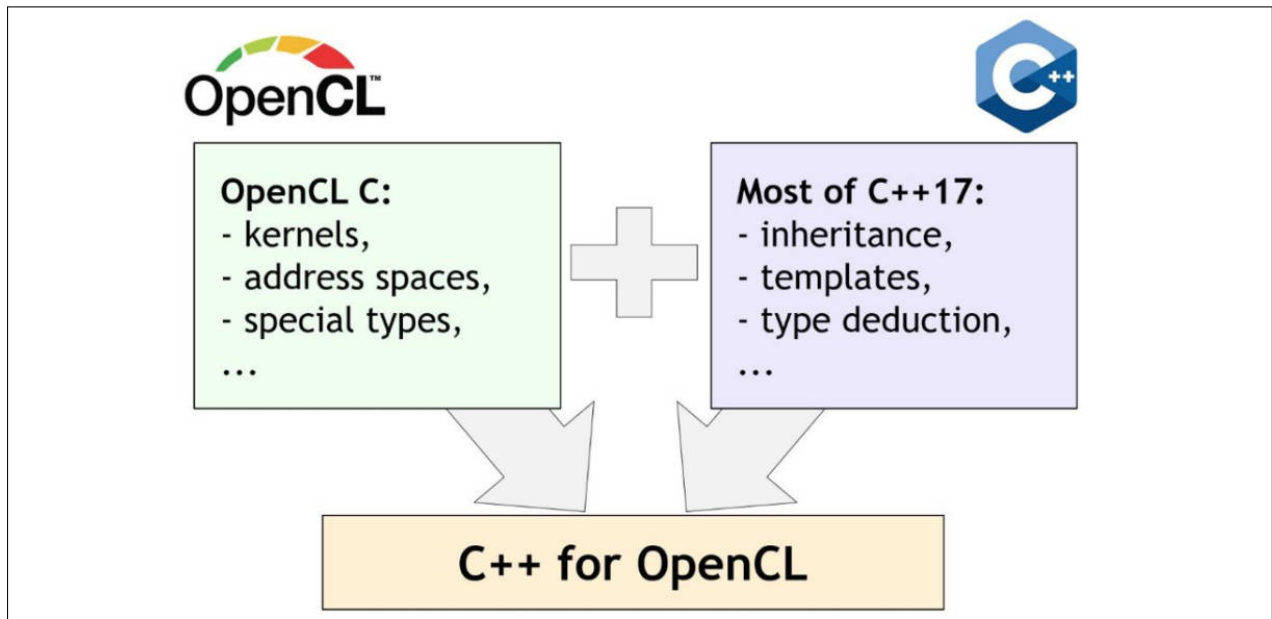


FIGURE 3.22: C++ for OpenCL. (By "www.khronos.org")

ships with a new unified API and OpenCL C 3.0 language specifications and introduces extensions for asynchronous data copies to enable a new class of embedded processors.[172]

C++ for OpenCL language In 2020 Khronos announced the transition to the community driven C++ for OpenCL programming language that provides features from C++17 in combination with the traditional OpenCL C features. This language allows to leverage a rich variety of language features from standard C++ while preserving backward compatibility to OpenCL C. This opens up a smooth transition path to C++ functionality for the OpenCL kernel code developers as they can continue using familiar programming flow and even tools as well as leverage existing extensions and libraries available for OpenCL C.

The language semantics is described in the documentation published in the releases of OpenCL-Docs repository hosted by the Khronos Group but it is currently not ratified by the Khronos Group. The C++ for OpenCL language is not documented in a stand-alone document and it is based on the specification of C++ and OpenCL C. The open source Clang compiler has supported C++ for OpenCL, since release 9.⁷⁷

3.5 IoT Protocols & Standards.

IoT protocols are an integral part of the IoT technologies. Without IoT protocols and standards, hardware are useless . This is because enable hardware, to exchange data. And, out of these transferred pieces of data, useful information can be extracted by the end-user.[140, 165]

IoT protocols and standards are often overlooked, when people think about the Internet of Things (IoT). More often than not, the industry/shipping has its attention firmly fixed upon communication. And while the interaction between devices, iot sensors, gateways, servers, and user applications are essential components of the IoT, communication would fall down without the right IoT protocols.

⁷⁷<https://en.wikipedia.org/wiki/OpenCL>

IoT protocols and standards are broadly classified into two separate categories⁷⁸. These are:

1. IoT data protocols (Presentation / Application layers)⁷⁹
2. Network protocols for IoT (Datalink / Physical layer)⁸⁰

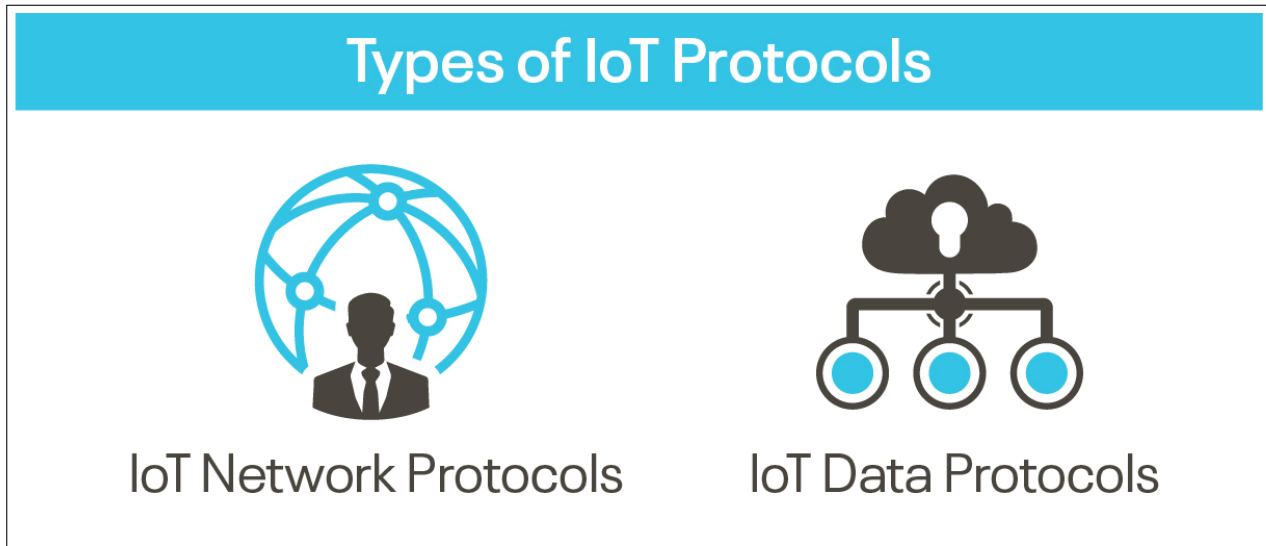


FIGURE 3.23:
Types of IoT Protocols.(by "Kellton Tech [dot] com")

3.5.1 IoT Data Protocols

IoT data protocols are used to connect IoT devices. They provide communication with hardware on the user side – without the need for any internet connection. The connectivity in IoT data protocols and standards is through a wired or cellular network.

Some examples of IoT data protocols are:

HTTP(Hypertext Transfer Protocol) If one has no problem with network connectivity, HTTP⁸⁰ is a feasible communication option. As a web protocol, the communication channel is through a web browser. Typically used together with HTML, it follows a request/response model. The essential message topics for HTTP are URI and QRL. Once a client relays a request to the server, the server will prompt a response. The server will determine the message size limit also. Compared to MQTT, HTTP is more on the heavy side as it is derived to be a text protocol. As a text protocol, it follows a bigger message size and higher overhead.

MQTT (Message Queuing Telemetry Transport) MQTT⁸² is an OASIS⁸³ standard messaging protocol for the Internet of Things (IoT). It is designed as an extremely lightweight publish/subscribe messaging transport that is ideal for connecting remote devices with a small code footprint

⁷⁸<https://www.kelltontech.com/kellton-tech-blog/internet-of-things-protocols-standards>

⁷⁹The parenthesis is a reference to the OSI network layer model

⁸⁰https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

⁸¹Hypertext Transfer Protocol Secure (HTTPS) is an extension of the Hypertext Transfer Protocol (HTTP). It is used for secure communication over a computer network, and is widely used on the Internet. In HTTPS, the communication protocol is encrypted using Transport Layer Security (TLS) or, formerly, Secure Sockets Layer (SSL).

⁸²<https://mqtt.org/>

⁸³<https://www.oasis-open.org/> >About > Organization #History

and minimal network bandwidth. MQTT today is used in a wide variety of industries, such as automotive, manufacturing, telecommunications, oil and gas, etc.

CoAP (Constrained Application Protocol) RFC 7252 Constrained Application Protocol “The Constrained Application Protocol (CoAP) is a specialized web transfer protocol for use with constrained nodes and constrained networks in the Internet of Things. The protocol is designed for machine-to-machine (M2M) applications such as smart energy and building automation.”⁸⁴

AMQP (Advanced Message Queuing Protocol) The Advanced Message Queuing Protocol (AMQP) is an open standard application layer protocol for message-oriented middleware. The defining features of AMQP are message orientation, queuing, routing (including point-to-point and publish-and-subscribe), reliability and security.

AMQP mandates the behavior of the messaging provider and client to the extent that implementations from different vendors are interoperable, in the same way as SMTP, HTTP, FTP, etc. have created interoperable systems. Previous standardizations of middleware have happened at the API level (e.g. JMS) and were focused on standardizing programmer interaction with different middleware implementations, rather than on providing interoperability between multiple implementations. Unlike JMS, which defines an API and a set of behaviors that a messaging implementation must provide, AMQP is a wire-level protocol. A wire-level protocol is a description of the format of the data that is sent across the network as a stream of bytes. Consequently, any tool that can create and interpret messages that conform to this data format can interoperate with any other compliant tool irrespective of implementation language.^{85 86}

DDS (Data Distribution Service) DDS is a networking middleware that simplifies complex network programming. It implements a publish–subscribe pattern for sending and receiving data, events, and commands among the nodes. Nodes that produce information (publishers) create “topics” (e.g., temperature, location, pressure) and publish “samples”. DDS delivers the samples to subscribers that declare an interest in that topic.⁸⁷

3.5.2 Network Protocols for IoT

IoT network protocols are used to connect devices over a network. These sets of protocols are typically used over the internet. Here are some examples of various IoT network protocols.

Industrial Ethernet⁸⁸ (IE) is the use of Ethernet in an industrial environment with protocols that provide determinism and real-time control. Protocols for industrial Ethernet Fieldbuses⁸⁹ include EtherCAT, EtherNet/IP, PROFINET, POWERLINK, SERCOS III, CC-Link IE, Modbus⁹⁰ TCP and NMEA - OneNet⁹¹. Many industrial Ethernet protocols use a modified Media Access Control (MAC) layer to provide low latency and determinism. Some microcontrollers such as Sitara provide industrial Ethernet support.

Industrial Ethernet can also refer to the use of standard Ethernet protocols with rugged connectors and extended temperature switches in an industrial environment, for automation or process

⁸⁴<https://coap.technology/> and https://en.wikipedia.org/wiki/Constrained_Application_Protocol

⁸⁵<https://www.amqp.org/about/what>

⁸⁶https://en.wikipedia.org/wiki/Advanced_Message_Queueing_Protocol

⁸⁷<https://www.dds-foundation.org/what-is-dds-3/>

⁸⁸https://en.wikipedia.org/wiki/Industrial_Ethernet

⁸⁹https://en.wikipedia.org/wiki/Fieldbus#IEC_61784-1:_Fieldbus_profiles

⁹⁰<https://www.modbus.org/>

⁹¹<https://www.nmea.org/content/STANDARDS/OneNet>

control. Components used in plant process areas must be designed to work in harsh environments of temperature extremes, humidity, and vibration that exceed the ranges for information technology equipment intended for installation in controlled environments. The use of fiber-optic Ethernet variants reduces the problems of electrical noise and provides electrical isolation.

Some of the advantages over other types of industrial network include:

- Increased speed, up from 9.6 kbit/s with RS-232 to 1 Gbit/s with Gigabit Ethernet
- Ability to use ubiquitous Cat5e/Cat6 cables
- Option to use optical fiber for increased distance
- Ability to use standard networking hardware for wired and wireless communication
- Ability to have more than two nodes on link, which was possible with RS-485 but not with RS-232
- Potential to use peer-to-peer architectures as opposed to client-server ones
- Better interoperability

Difficulties of using industrial Ethernet include:

- Migrating existing systems to a new protocol
- Real-time performance may suffer for protocols using TCP
- Additional complexity associated with network technology
- The minimum Ethernet frame size is 64 bytes, while typical industrial communication data sizes can be closer to 1–8 bytes. This protocol overhead affects data transmission efficiency.

WiFi Wi-Fi provides an internet connection to nearby devices within a specific range. Another way to use Wi-Fi is to create a Wi-Fi hot-spot. Mobile phones or computers may share a wireless or wired internet connection with other devices by broadcasting a signal.

Wi-Fi uses radio waves that broadcast information on specific frequencies, such as 2.4 GHz or 5GHz channels. Furthermore, both of these frequency ranges have a number of channels through which different wireless devices can work. This prevents the overflowing of wireless networks.

A range of 100 meters is typical of a Wi-Fi connection. That being said, the most common is limited to 10-35 meters. The main impacts on the range and speed of a Wi-Fi connection are the environment and whether it provides internal or external coverage.

Bluetooth Bluetooth is a short-range wireless technology standard that is used for exchanging data between fixed and mobile devices over short distances using UHF radio waves in the ISM bands, from 2.402 GHz to 2.48 GHz, and building personal area networks (PANs).[4] It was originally conceived as a wireless alternative to RS-232 data cables. It is mainly used as an alternative to wire connections, to exchange files between nearby portable devices and connect cell phones and music players with wireless headphones. In the most widely used mode, transmission power is limited to 2.5 milliwatts, giving it a very short range of up to 10 meters (30 feet).

ZigBee Zigbee is an IEEE 802.15.4-based specification for a suite of high-level communication protocols used to create personal area networks with small, low-power digital radios, such as for home automation, medical device data collection, and other low-power low-bandwidth needs, designed for small scale projects which need wireless connection. Hence, Zigbee is a low-power, low data rate, and close proximity (i.e., personal area) wireless ad hoc network.

The technology defined by the Zigbee specification is intended to be simpler and less expensive than other wireless personal area networks (WPANs), such as Bluetooth or more general wireless networking such as Wi-Fi. Applications include wireless light switches, home energy monitors, traffic management systems, and other consumer and industrial equipment that requires short-range low-rate wireless data transfer.

Its low power consumption limits transmission distances to 10–100 meters line-of-sight, depending on power output and environmental characteristics. Zigbee devices can transmit data over long distances by passing data through a mesh network of intermediate devices to reach more distant ones. Zigbee is typically used in low data rate applications that require long battery life and secure networking. (Zigbee networks are secured by 128 bit symmetric encryption keys.) Zigbee has a defined rate of 250 kbit/s, best suited for intermittent data transmissions from a sensor or input device.⁹²

LoRaWAN The LoRaWAN® specification is a Low Power, Wide Area (LPWA) networking protocol designed to wirelessly connect battery operated ‘things’ to the internet in regional, national or global networks, and targets key Internet of Things (IoT) requirements such as bi-directional communication, end-to-end security, mobility and localization services.⁹³

3.5.3 The JSON Data Interchange Standard Format.(ECMA-404)

JSON⁹⁴ (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate. It is based on a subset of the JavaScript Programming Language Standard ECMA-262 3rd Edition - December 1999.[68, 73, 77]

JSON is a text format that is completely language independent but uses conventions that are familiar to programmers of the C-family of languages, including C, C++, C#, Java, JavaScript, Perl, Python, and many others. **These properties make JSON an ideal data-interchange language.**

JSON is built on two structures:

- A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
- An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

These are universal data structures. Virtually all modern programming languages support them in one form or another. It makes sense that a data format that is interchangeable with programming languages also be based on these structures.

In JSON, they take on these forms:

An object is an unordered set of name/value pairs. An object begins with *{ left brace* and ends with *} right brace*. Each name is followed by *:colon* and the name/value pairs are separated by *,comma*.

⁹²<https://csa-iot.org/>

⁹³<https://lora-alliance.org/about-lorawan/>

⁹⁴<https://www.json.org/json-en.html>

An array is an ordered collection of values. An array begins with *left bracket* and ends with *right bracket*. Values are separated by *comma*.

A value can be a string in double quotes, or a number, or true or false or null, or an object or an array. These structures can be nested.

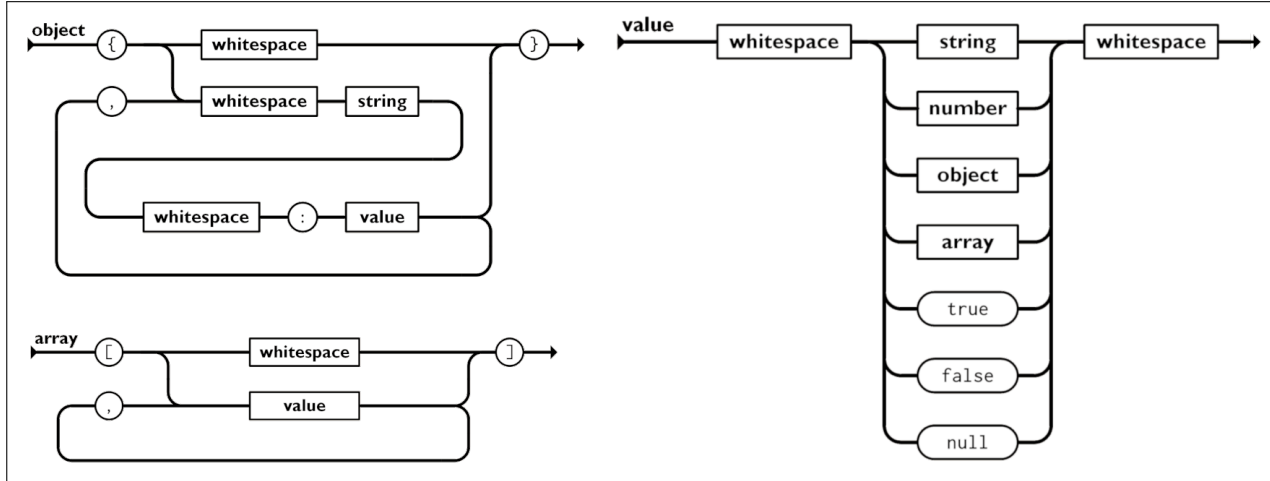


FIGURE 3.24:
Dictionary - object, Array - list and Values definitions (by "JSON Org.")

3.5.4 The WebSocket Protocol.

WebSocket^{95 96} is a computer communications protocol, providing full-duplex communication channels over a single TCP connection. The WebSocket protocol was standardized by the IETF as RFC 6455 in 2011, and the WebSocket API in Web IDL is being standardized by the W3C.

WebSocket is distinct from HTTP. Both protocols are located at layer 7 in the OSI model and depend on TCP at layer 4. Although they are different, RFC 6455 states that WebSocket "is designed to work over HTTP ports 443 and 80 as well as to support HTTP proxies and intermediaries," thus making it compatible with HTTP.

The WebSocket protocol enables interaction between a web browser (or other client application) and a web server with lower overhead than half-duplex alternatives such as HTTP polling, facilitating real-time data transfer from and to the server. This is made possible by providing a standardized way for the server to send content to the client without being first requested by the client, and allowing messages to be passed back and forth while keeping the connection open. In this way, a two-way ongoing conversation can take place between the client and the server. The communications are usually done over TCP port number 443 (or 80 in the case of unsecured connections), which is beneficial for environments that block non-web Internet connections using a firewall.

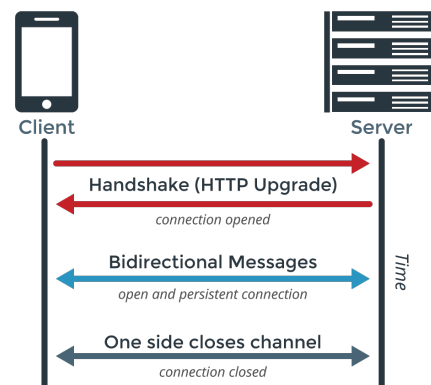


FIGURE 3.25: Websockets Overview.

⁹⁵<https://en.wikipedia.org/wiki/WebSocket>

⁹⁶<https://datatracker.ietf.org/doc/html/rfc6455#section-1.2t>

3.5.5 Extensible Markup Language (XML) Format.

Extensible Markup Language (XML)⁹⁷ is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. XML have an increasingly important role in the exchange of a wide variety of data on the Web and elsewhere.

The design of XML emphasize simplicity, generality, and usability across the Internet. It is a textual data format with strong support via Unicode for different human languages. Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary data structures such as those used in web services.

```
1 {
2 "Engine Room":
3   [
4     {"/dev/ttyACM0": "ERMED"},
5     {"System Time": "2021-11-29 06:42:01.381919"},
6     {"Left Main Engine ":
7       [
8         0,
9         {"Temperatures":
10          { "T#1": 81.97,
11            "T#2": 82.46,
12            "T#3": 81.97,
13            "T#4": 81.97,
14            "T#5": 81.97,
15            "T#6": 82.46
16          }
17        ]
18      ]
19    },
20    {"Rigth Main Engine ":
21      [
22        1,
23        {"Temperatures":
24          { "T#1": 84.97,
25            "T#2": 85.46,
26            "T#3": 84.97,
27            "T#4": 84.97,
28            "T#5": 84.97,
29            "T#6": 85.46
30          }
31        ]
32      ]
33    }
34  ]
35 }
```

LISTING 3.1: Sample of JSON Format File.

⁹⁷<https://www.w3.org/XML/> and <https://en.wikipedia.org/wiki/XML>

```
1 <?xml version="1.0" encoding="UTF-8" ?>
2 <root>
3   <Engine_Room>
4     </dev/ttyACM0>ERMED
5   </dev/ttyACM0>undefined</Engine_Room>undefined<Engine_Room>
6   <System_Time>2021-11-29 06:42:01.381919</System_Time>undefined</Engine_Room>
   undefined<Engine_Room>
7   <Left_Main_Engine>0</Left_Main_Engine>
8   <Left_Main_Engine>
9     <Temperatures>
10      <T#1>81.97</T#1>
11      <T#2>82.46</T#2>
12      <T#3>81.97</T#3>
13      <T#4>81.97</T#4>
14      <T#5>81.97</T#5>
15      <T#6>82.46</T#6>
16    </Temperatures>
17  </Left_Main_Engine>undefined</Engine_Room>undefined<Engine_Room>
18  <Rigth_Main_Engine>1</Rigth_Main_Engine>
19  <Rigth_Main_Engine>
20    <Temperatures>
21      <T#1>84.97</T#1>
22      <T#2>85.46</T#2>
23      <T#3>84.97</T#3>
24      <T#4>84.97</T#4>
25      <T#5>84.97</T#5>
26      <T#6>85.46</T#6>
27    </Temperatures>
28  </Rigth_Main_Engine>undefined</Engine_Room>undefined</root>
```

LISTING 3.2: Sample of XML Format File.



FIGURE 3.26: Protocol Buffers Logo by Google.

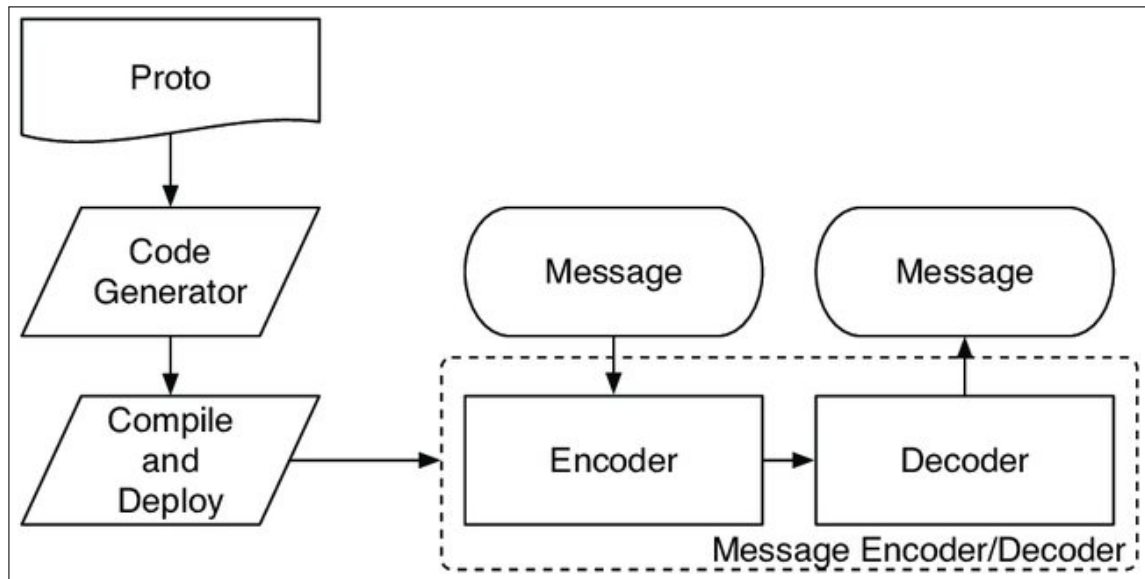


FIGURE 3.27: Protocol Buffers Main Idea Block Diagram.

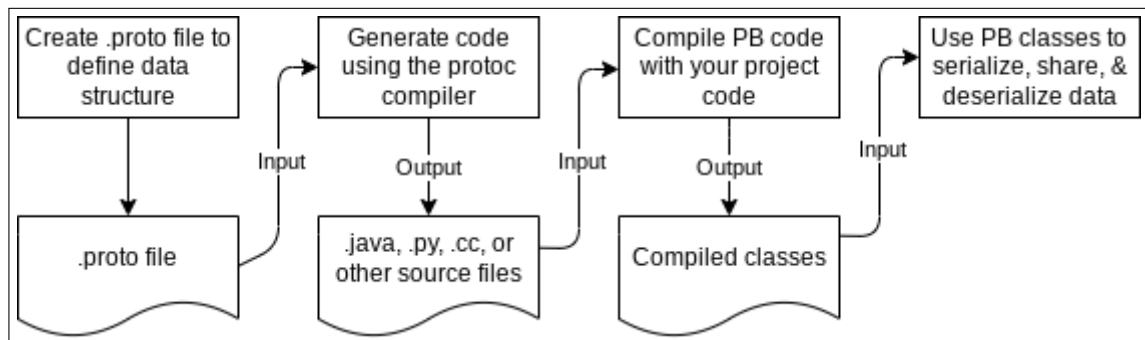


FIGURE 3.28: Protocol Buffers Procedure Diagram.

3.6 Protobuf by Google.

Protocol buffers^{98 99 100} are Google’s language-neutral, platform-neutral, extensible mechanism for serializing structured data – think XML, but smaller, faster, and simpler. Define how we want our data to be structured once, then we can use special generated source code to easily write and read your structured data to and from a variety of data streams and using a variety of languages.

Protocol buffers currently support generated code in **Java, Python, Objective-C, and C++**. With new proto3 language version, can also work with **Kotlin, Dart, Go, Ruby, and C#**, with more languages to come.

⁹⁸<https://developers.google.com/protocol-buffers>

⁹⁹<https://protobuf.dev>

¹⁰⁰https://en.wikipedia.org/wiki/Protocol_Buffers

3.7 NMEA Standards.

The National Marine Electronics Association[169] is a US-based marine electronics trade organization setting standards of communication between marine electronics. There is no other centric organization that provides standards and education for the Marine Electronics Industry.

Marine electronic dealers, manufacturers, boat builders, government and other stakeholders worldwide utilize the NMEA Standards as their reference for marine electronics, whether it is for communication protocols, installation or education in order to try to achieve harmonization in the worldwide market.

NMEA Installation Standards provide the marine industry a common technical methodology for the safe installation and operation of marine electronics. NMEA 0183, NMEA 2000, and NMEA OneNet are the communication protocol interfaces that are used in marine electronic devices worldwide. These provide a common interface so devices on vessels have a universal language to communicate. NMEA utilizes these standards for educational purposes to the marine industry at large.

The NMEA 0400 Installation Standard is a voluntary standard that clarifies and defines competent installation best practices applicable to vessels from 20' to 150' and up to 300 gross tons. These standards are published to aid electronics installers, technicians, electricians, surveyors, vessel owners and/or others who may service or modify the installation of electronics, electrical systems or other associated peripherals.

3.7.1 NMEA 0183 Interface Standard

The NMEA 0183 Interface Standard¹⁰¹ is used worldwide across many industry segments. The standard defines electrical signal requirements, data transmission protocol and time, and specific sentence formats for a 4800-baud serial data bus. Each bus may have only one talker but many listeners. This standard is intended to support one-way serial data transmission from a single talker to one or more listeners. This data is in printable ASCII form and may include information such as time, position, speed, water depth, etc.

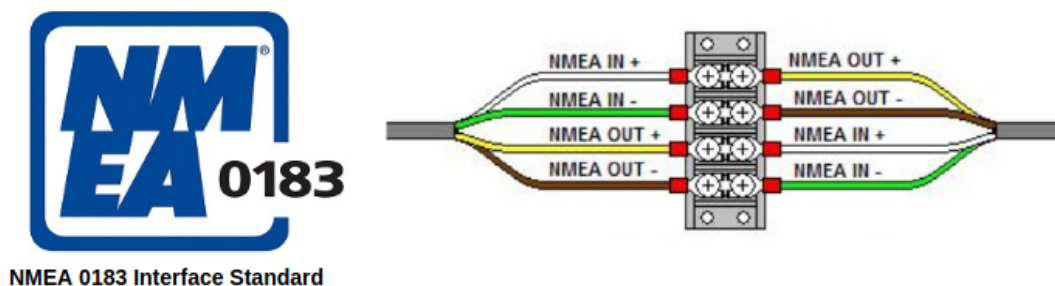


FIGURE 3.29:
NMEA 0183 Interface. (by "NMEA Org.")

Latest Version of NMEA 0183 - 4.11 As of November 2018, NMEA has published a new version of NMEA 0183—Version 4.11, which replaces Version 4.10 and includes updates to the entire suite of GNSS (Global Navigation Satellite System) sentences. This includes interface clarification for the use of GPS (US), GLONASS (Russia), GALILEO (Europe), BDS (China), QZSS (Japan), and NavIC ((IRNSS) (India)). Deployment of these new systems ramped up the need for sentence

¹⁰¹https://www.nmea.org/content/STANDARDS/NMEA_0183_Standard

updates. Also included are new Navigation Satellite System Receiver Talker Identifiers and Sentence Formatters, an expanded GNSS Identification Table and examples of the use of the satellite system.

NMEA 0183 HS (High Speed) The NMEA 0183-HS standard defines electrical signal requirements, data transmission protocol, and timing for a 38.4K-baud serial data bus. Each bus shall have only a **single TALKER** but may have **multiple LISTENERS**. Specific sentence formats are common to both NMEA 0183 and NMEA 0183-HS and are defined in NMEA 0183. Standard rate NMEA 0183 operates at 4800-baud and utilizes a different electrical interface

3.7.2 NMEA 2000® Interface Standard

The NMEA 2000® standard ¹⁰² contains the requirements of a serial data communications network to inter-connect marine electronic equipment on vessels. The standard describes a low-cost moderate capacity bi-directional, multi-transmitter/multi-receiver instrument network to inter-connect marine electronic devices. It is multi-master and self configuring, and there is no central network controller.

Equipment designed to this standard will have the ability to share data, including commands and status with other compatible equipment over a single channel. It is based on CAN (Controller Area Network)¹⁰³ [56].

All NMEA 2000® products must be certified by NMEA. Although this standard is 50 times faster than NMEA 0183, it is not intended to support high-bandwidth applications such as video. NMEA 2000® represents the cumulative efforts of the NMEA 2000® Standards Committee, a committee of multi-national industry and government subject matter experts and professionals who have contributed time and expertise to the development this standard. IEC 61162-3 references NMEA 2000®.

NMEA 2000® is a registered Trademark of the National Marine Electronics Association.[38, 146]



FIGURE 3.30:
NMEA 2000 Logo.
(by "NMEA Org.")

3.7.3 OneNet Standard for IP Networking of Marine Electronic Devices

The OneNet Standard for IP Networking¹⁰⁴ of Marine Electronic Devices is an open industry standard based on Internet Protocol, Version 6 (IPv6) and the IEEE 802.3 Ethernet Local Area Network. OneNet provides a common network infrastructure for marine devices and/or services on IPv6. All OneNet application protocols, such as PGN Messages, are designed to use a standard IPv6 network protocol stack. This allows OneNet to coexist with other protocols and services that operate parallel on the same network (including other marine standards such as IEC 61162-450). The standard also specifies mechanisms for connecting OneNet networks, NMEA 2000 networks, and other networks via gateway devices. Like NMEA 2000, all OneNet products will need to be certified by the manufacturer and verified by NMEA.



FIGURE 3.31: OneNet Logo. (by "NMEA Org.")

¹⁰²https://www.nmea.org/content/STANDARDS/NMEA_2000

¹⁰³<https://www.totalphase.com/blog/2019/08/5-advantages-of-can-bus-protocol/>

¹⁰⁴<https://www.nmea.org/content/STANDARDS/OneNet>

OneNet Goals

- To specify the transport of NMEA Network Messages over IPv6
- To utilize standard IP network and addressing infrastructure
- To facilitate the safe and secure communication and operations among equipment
- To co-exist with other IP protocols / services on the same cable
- To discover devices and services automatically to create an extendible and scalable network architecture
- To define an open interface to interoperate with current and upcoming open services
- To deliver interoperability with the established industry standards including NMEA 2000 (i.e., establishing gateway rules between NMEA 2000 and OneNet)
- To support high-bandwidth applications such as audio/video data transport

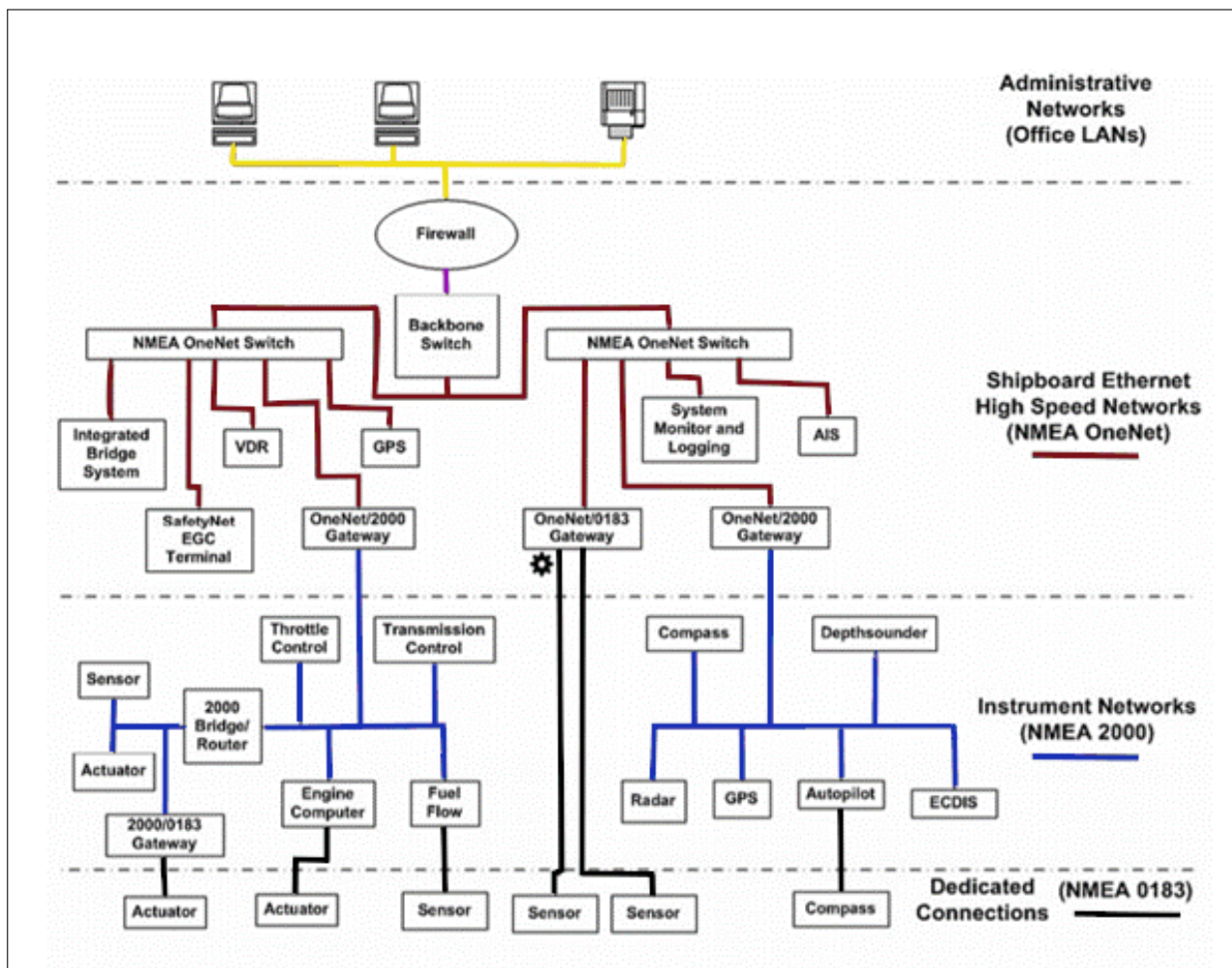


FIGURE 3.32:
OneNet NMEA Standard Overview. (by "NMEA Org.")

OneNet Key Benefits

- Standardized architecture. Transport of NMEA Network Messages as well as other protocols over IPv6-based networks (this aligns with M2M, IoT trends)
- Greater bandwidth. With a range of 100Megabit per second to 10 Gigabit per second transfer speeds directly to OneNet Devices, Ethernet is about 400 to 40,000 times faster than NMEA 2000.
- Much greater number of potential devices (addresses). OneNet allows larger and more complex networks to be created than can be achieved with the 252 address limitation on NMEA 2000.
- Greater power capacity. With Power over Ethernet (PoE), each OneNet Device may be separately powered up to 25.5Watts directly from the Ethernet Switch (refer to IEEE 802.3at).
- Ubiquitous technology. Ethernet is used everywhere in homes, offices and industrial environments and is well understood. Many marine electronic products already implement and support Ethernet.

3.7.4 NMEA Data Protocol

The NMEA Protocol is a simple, yet comprehensive ASCII protocol which defines both the communication interface and the data format. The protocol was originally established to enable marine navigation equipment to share information. Since it is a well-established industry standard, NMEA has also gained popularity for use in applications other than marine electronics.[30]

There are two major types of NMEA messages supported by the NMEA protocol: Approved (Standard) messages and Proprietary (Additional) messages. Standard messages are defined by the NMEA standard while Proprietary messages are set by GNSS receiver manufacturers and are based on NMEA standard regulations. Standard messages are sent by a navigation receiver by default or as a reply to a Query sentence (Q). Proprietary messages may be defined as incoming (Input), outgoing (Output) or both.

NMEA messages vary in length, but each message is limited to a maximum of 79 characters between the starting delimiter "\$" and the terminating sequence <CR><LF>.

Address Fields The Address field is divided into 2 fields.

The first field of standard (output) messages must be 2 characters representing Taker ID: "GP" for GPS, "GL" – GLONASS, "GA" – GALILEO and "GN" for combined navigation solution (if information of more than one satellite navigation system is used for data generation).

The second field of standard (output) messages must be 3 characters representing Sentence ID. Please refer to messages Message Description for further information regarding Talker ID and Sentence ID. The first character of the Address field of Proprietary messages should be the character "P" followed by Sentence ID.

The query message address field consists of five characters and is used for the purpose of requesting transmission of a specific message(s). The first two characters are the Talker Identifier of the device requesting data, the next two characters are the Talker ID of the device being addressed (navigation receiver) and the final character is the query character "Q". Address field of the messages must contain only digits and uppercase letters and cannot be "null".

Data fields A data field consists of a string of valid characters (or no characters - null field) between two delimiters ",". The string length may be fixed or variable. If a particular data field has zero length then only delimiter represents this field.

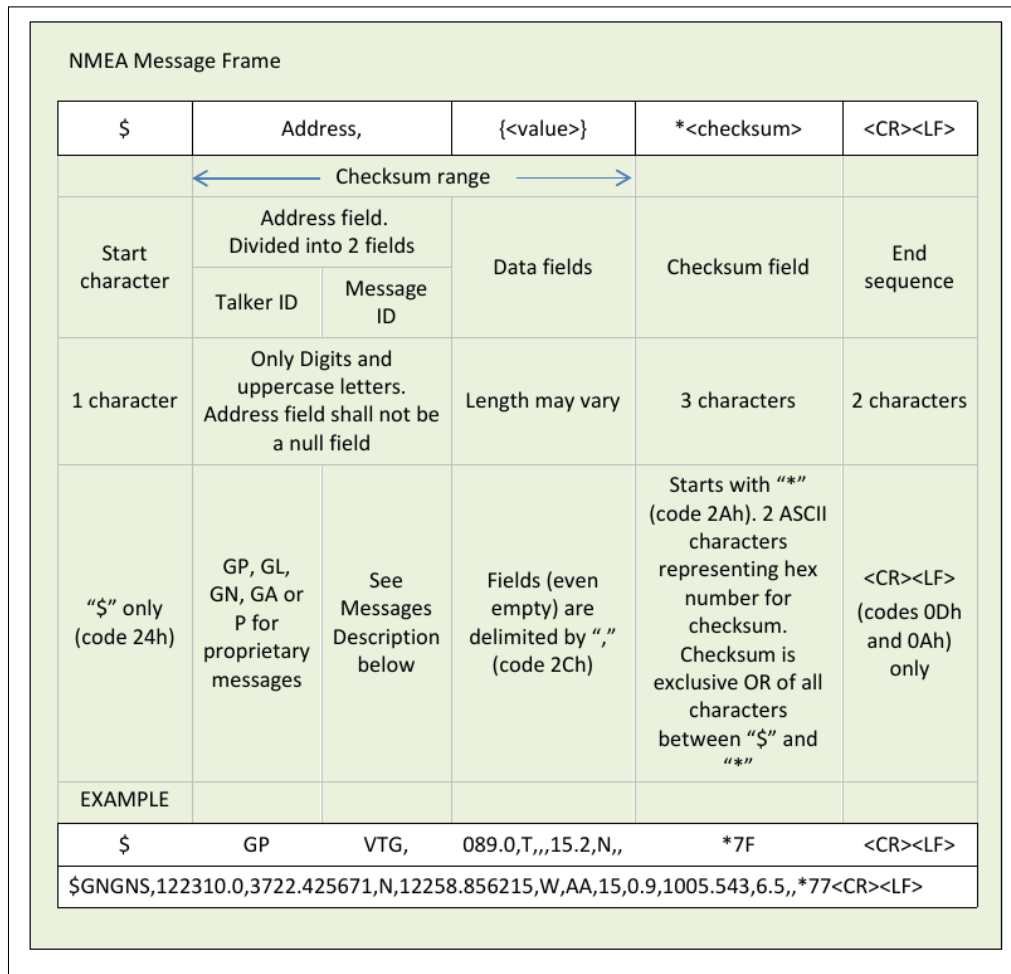


FIGURE 3.33: The structure of a NMEA protocol message

According to NMEA 0183 Latitude and Longitude are transmitted in the format Degrees, Minutes and (Decimal) Fractions of Minutes. Time is transmitted in the format Hours, Minutes, Seconds and (Decimal) Fractions of Seconds. Fractions are delimited by ".". Both fractions and delimiter "." are optional.

Some (defined) fields are specified to contain pre-defined constants, most often alphabetic characters. Such fields are indicated in the NMEA standard by the presence of one or more valid characters. Excluded from the list of allowable characters are the following characters that are used to indicate field types: "A", "a", "c", "hh", "hhmmss.ss", "llll.ll", "x", "yyyyy.y

Checksum A checksum field is required and shall be transmitted in all sentences. The checksum field is the last field in a sentence and follows the checksum delimiter character "*".

The checksum is the 8-bit exclusive OR (no start or stop bits) of all characters in the sentence, including "," delimiters, between but not including the "\$" and the "*" delimiters. The hexadecimal value of the most significant and least significant 4 bits of the result is converted to two ASCII characters (0-9, A-F (upper case)) for transmission. The most significant character is transmitted first.

Example The GGA message outputs time, position and fix related data. The fix is based on all available GNSS. This message is similar to the GNS – GNSS Fix Data GNS message.

```
$GPGGA,181722.00,4000.1256100,N,08301.5461206,W,1,12,99.99,221.231,M,-33.698,M,,*4A
```

Message Format

\$ aa GGA, hhmmss.s-s, llll.l-l, a, yyyyy.y-y, a, x, xx, x.x, x.x, M, x.x, M, x.x, xxxx *hh <CR><LF>

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Message Fields

#	Field	Format
1	\$	Start character (Code 24h)
2	aa	Talker ID (see chapter 2.3)
3	GGA	Message ID
4	hhmmss.s-s	Time of position fix
5	llll.l-l, a	Latitude, a a: N (North) or S (South)
6	yyyyy.y-y, a	Longitude, a a: E (East) or W (West)
7	x	Position Fix Flag 0 = Fix not available or invalid 1 = Position fix valid, autonomous mode 2 = Position fix valid, Differential mode 6 = Estimated data (extrapolation, dead reckoning)
8	xx	Number of satellites used in calculation
9	x.x	Horizontal dilution of precision (HDOP)
10	x.x, M	Altitude re: mean-sea-level (geoid), metres
11	x.x, M	Geoidal separation, metres Geoidal Separation: the difference between the WGS 84 earth ellipsoid surface and mean-sea-level (geoid) surface, "." = mean-sea-level surface below WGS 84 ellipsoid surface.
12	x.x	Age of Differential GNSS data. Time in seconds since last SC104 Type 1 or 9 update, null field when DGPS is not used.
13	xxxx	Differential reference station ID, 0000-1023. Null field when DGNSS is not used
14	*hh	Checksum indicator ("*", code 2Ah) and checksum
15	<CR><LF>	End of message indicator (codes 0Dh and 0Ah)

FIGURE 3.34: Example NMEA: "xxGGA" Message.

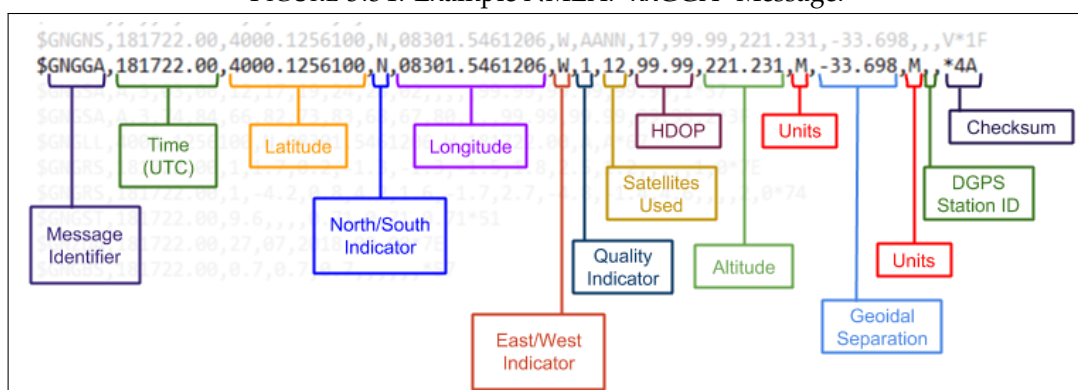


FIGURE 3.35: Example NMEA GGA Sentence Decoding. (by brandidowns.com)

3.8 Signal K

Signal K ¹⁰⁵ is a modern and open data format for marine use. Built on standard web technologies including JSON ¹⁰⁶, Web-sockets^{107 108} and HTTP. Signal K provides a method for sharing information in a way that is friendly to WiFi, cellphones, tablets and the Internet.

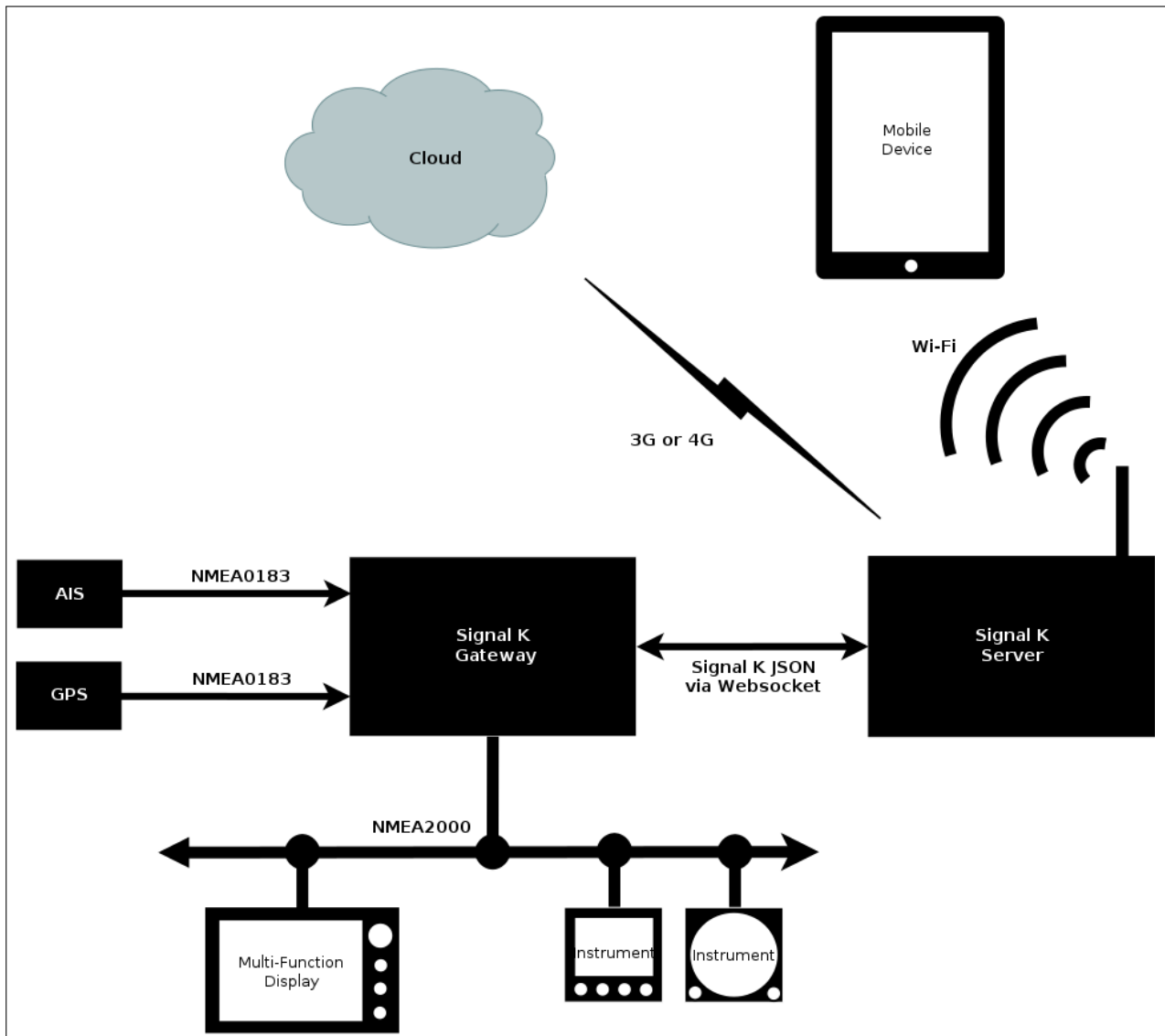


FIGURE 3.36:
The Signal K Data Model. (by "Signal-K Org.")

Signal K is the next generation solution for marine data exchange¹⁰⁹. It not only allows for communication between instruments and sensors on board a single vessel, but also allows sharing of data between multiple boats, aids to navigation, bridges, marinas and other land-based resources. It is designed to be easily used by Web and Mobile applications and to connect modern

¹⁰⁵<https://signalk.org>

¹⁰⁶JSON (JavaScript Object Notation) is a lightweight data-interchange format. It is easy for humans to read and write. It is easy for machines to parse and generate.

¹⁰⁷<https://en.wikipedia.org/wiki/WebSocket>

¹⁰⁸<https://datatracker.ietf.org/doc/html/draft-ietf-hybi-thewebsocketprotocol>

¹⁰⁹Is the basic alternative to NMEA OneNet, without extra specific networking equipment's.

boats to the Internet of Things.

Signal K uses modern techniques to create a marine data standard for the 21st Century. It is:

- **Open.** Signal K is managed by a community of boaters, and developers are able to propose improvements to the standard.
- **Free.** All of the code is published under Open Source licenses for anyone to use free of charge.
- **Modern.** It uses widely available, Open Source technologies.
- **Extensible.** It can mature with new requirements as they emerge.
- **Flexible.** Signal K is not tied to specific hardware.
- **Respectful.** It is designed to interface to existing equipment and protocols.

3.9 Navigation Bridge Systems

In a complete Vessel Performance Monitoring system, where everything is networked. We can not exclude the Electronic and Information Systems of the Navigation Bridge. Which systems, we treat, as smart and special sensors.[119]

3.9.1 Radar.

A mandatory aid to navigation, the radar is used in identifying, tracking and positioning of vessels (including one's own vessel) among other things in order to safely navigate a ship from one point to another.¹¹⁰

Radar can connect via NMEA Networking Protocol to ARPA and ECDIS systems. For more efficient management of identified targets. [1, 10, 31, 136, 151, 162]

Using,

- mathematical algorithms for maritime motion calculations,
- comparison with electronic cartographic data, and
- identification of target vessels, through the data received by the AIS.

3.9.2 Satellite Based Positioning and Navigation Systems.

A satellite navigation¹¹¹ or satnav system is a system that uses satellites to provide autonomous geo-spatial positioning. It allows small electronic receivers to determine their location (longitude, latitude, and altitude/elevation) to high precision (within a few centimeters to metres) using time signals transmitted along a line of sight by radio from satellites. The system can be used for providing position, navigation or for tracking the position of something fitted with a receiver (satellite tracking). The signals also allow the electronic receiver to calculate the current local time to high precision, which allows time synchronization. These uses are collectively known as Positioning,

¹¹⁰<https://www.marineinsight.com/marine-navigation/marine-radars-and-their-use-in-the-shipping-industry/>

¹¹¹https://en.wikipedia.org/wiki/Satellite_navigation

Navigation and Timing (PNT). Satnav systems operate independently of any telephonic or internet reception, though these technologies can enhance the usefulness of the positioning information generated.

A satellite navigation system with global coverage may be termed a global navigation satellite system (GNSS). The status as of September 2020 ,

- the United States' Global Positioning System (GPS),
- Russia's Global Navigation Satellite System (GLONASS),
- China's BeiDou Navigation Satellite System (BDS)
- European Union's Galileo[2] are fully operational GNSSs.
- Japan's Quasi-Zenith Satellite System (QZSS) is a (US) GPS satellite-based augmentation system to enhance the accuracy of GPS,
- and Indian Regional Navigation Satellite System (IRNSS) plans to expand to a global version in the long term.

3.9.3 Maritime Human-Machine Interfaces (M-HMI)

A Human-Machine Interface (HMI) is a user interface or dashboard that connects a person to a machine, system, or device. While the term can technically be applied to any screen that allows a user to interact with a device, HMI is most commonly used in the context of an industrial or shipping processes.

Although HMI is the most common term for this technology, it is sometimes referred to as Man-Machine Interface (MMI), Operator Interface Terminal (OIT), Local Operator Interface (LOI), or Operator Terminal (OT). HMI and Graphical User Interface (GUI) are similar but not synonymous: GUIs are often leveraged within HMIs for visualization capabilities.¹¹² For vessels in Navigation Bridge Systems the Maritime Human-Machine Interfaces (M-HMI) including Automatic Radar Plotting Aid (ARPA) and Electronic Chart Display and Information System (ECDIS).

Automatic Radar Plotting Aid (ARPA).

A marine radar with automatic radar plotting aid (ARPA) capability can create tracks using radar contacts. The system can calculate the tracked object's course, speed and closest point of approach (CPA), thereby knowing if there is a danger of collision with the other ship or landmass.¹¹³

Electronic Chart Display and Information System (ECDIS)

An Electronic Chart Display and Information System (ECDIS) is a geographic information system used for nautical navigation that complies with International Maritime Organization (IMO) regulations as an alternative to paper nautical charts.^{114 115}

An ECDIS system displays the information from Electronic Navigational Charts (ENC) and integrates position information from position, heading and speed through water reference systems and optionally other navigational sensors. Other sensors which could interface with an ECDIS are radar, Navtex, Automatic Identification Systems (AIS), and depth sounders.¹¹⁶ [19]

¹¹²<https://www.inductiveautomation.com/resources/article/what-is-hmi>

¹¹³https://en.wikipedia.org/wiki/Automatic_radar_plotting_aid

¹¹⁴https://en.wikipedia.org/wiki/Electronic_Chart_Display_and_Information_System

¹¹⁵<https://www.imo.org/en/OurWork/Safety/Pages/ElectronicCharts.aspx>

¹¹⁶In recent years concerns from the industry have been raised as to the system's security especially with regards to cyber attacks and GPS spoofing attacks

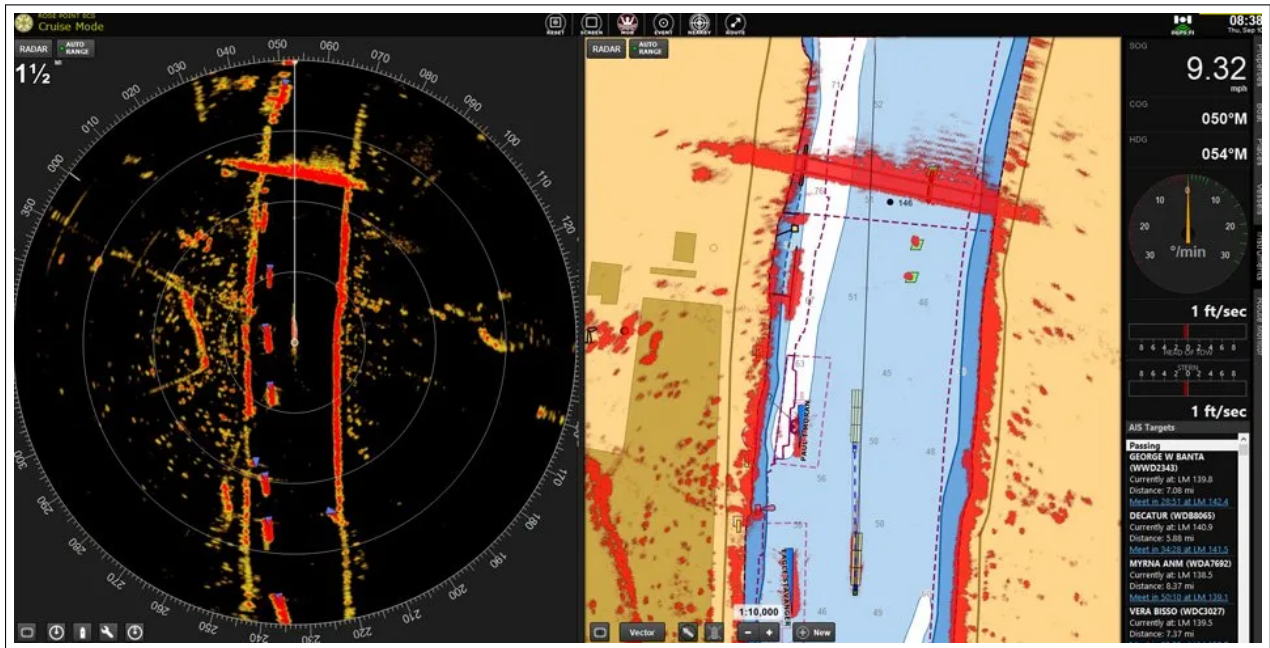


FIGURE 3.37: Rose-Point-ECS-Radar-Overlay-Feature-img. (by veinland.net)

3.9.4 Integrated bridge system (IBS)

An integrated bridge system (IBS) is defined as a combination of systems^[4, 5] which are interconnected in order to allow centralized access to sensor information or command/control from workstations, with the aim of increasing safe and efficient ship's management by suitably qualified personnel.¹¹⁷

An integrated bridge navigation system is generally connected to¹¹⁸ :

- Autopilot
- Radar
- Gyro
- Position fixing systems
- ECDIS
- Power distribution system
- Steering gear
- AIS
- Echo Sounder 1
- Navtex

¹¹⁷<https://www.imo.org/en/OurWork/Safety/Pages/IntegratedBridgeSystems.aspx>

¹¹⁸<http://maritimeknowledge.blogspot.com/2015/12/ibsintegrated-bridge-system.html>

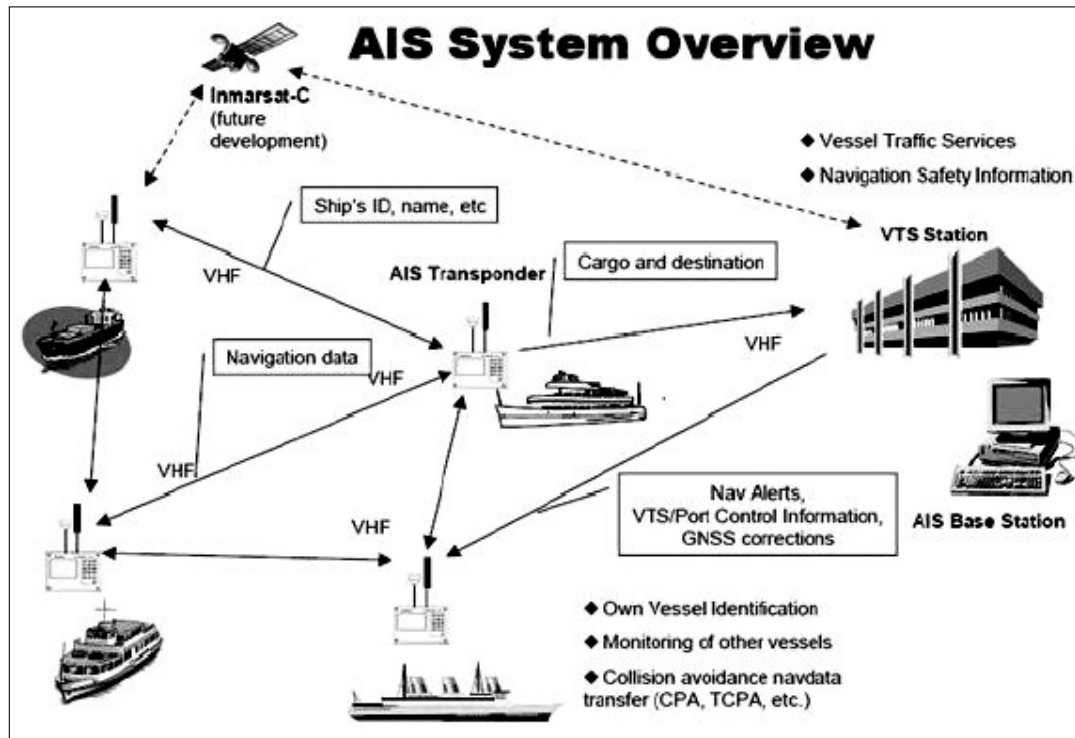


FIGURE 3.38: AIS overview. Image by Adam Weintrit on researchgate dot net

3.9.5 AIS

The automatic identification system (AIS) is an automatic tracking system that uses transceivers on ships and is used by vessel traffic services (VTS). When satellites are used to detect AIS signatures, the term Satellite-AIS (S-AIS) is used. AIS information supplements marine radar, which continues to be the primary method of collision avoidance for water transport. Although technically and operationally distinct, the ADS-B system is analogous to AIS and performs a similar function for aircraft.^{119 120}

Information provided by AIS equipment, such as unique identification, position, course, and speed, can be displayed on a screen or an electronic chart display and information system (ECDIS). AIS is intended to assist a vessel's watchstanding officers and allow maritime authorities to track and monitor vessel movements. AIS integrates a standardized VHF transceiver with a positioning system such as a Global Positioning System receiver, with other electronic navigation sensors, such as a gyrocompass or rate of turn indicator. Vessels fitted with AIS transceivers can be tracked by AIS base stations located along coast lines or, when out of range of terrestrial networks, through a growing number of satellites that are fitted with special AIS receivers which are capable of de-conflicting a large number of signatures. [7, 23, 34, 48, 57, 65, 76, 84, 93, 122]

¹¹⁹https://en.wikipedia.org/wiki/Automatic_identification_system

¹²⁰<https://www.aishub.net/>

3.10 Non -Destructive Evaluation 4.0

There is a new and rapidly developing landscape in non-destructive evaluation (NDE) and non-destructive testing (NDT) called NDE 4.0. This movement toward digital transformation has the potential to drastically shift how NDE is performed in the field. And how the businesses and shipping, the operate on a macro level.

NDE 4.0 is modeled after Industry 4.0, which has its roots in German manufacturing. Industry 4.0 is known as a subset of the fourth industrial revolution. And utilized by NATO in Aircraft maintainability.

NDE 4.0, as a "triplets sister" of Industry 4.0 and Maritime/Shipping 4.0 gets a great part in the vessel maintainability planning.

At its essence, Industry 4.0 is the practice of automating manufacturing processes with technologies like machine learning, cyber physical systems (CPS), digital twins, the internet of things (IoT), cloud computing and artificial intelligence (AI).

For example, Industry 4.0 smart factories (and vessels) rely on automated machines and sensors that are connected to computer systems, allowing them to collect data, learn from the data, and then finally make the necessary decisions.

Based on their levels of NDE 4.0 adoption, vessels can harness the power of digital transformation to become much more efficient in their operations, and more competitive with others in the shipping field. The applied NDE 4.0, as a "triplets sister" of Industry 4.0 and Maritime/Shipping 4.0 gets a great part in the vessel maintainability planning

Future Maritime 4.0 platforms will integrate NDE 4.0 advanced testing instruments into a single source of records, automate inspection operations practices, improve workforce recruitment and training, and provide better customer experiences. [2, 14, 60, 61, 82, 83, 100, 111]

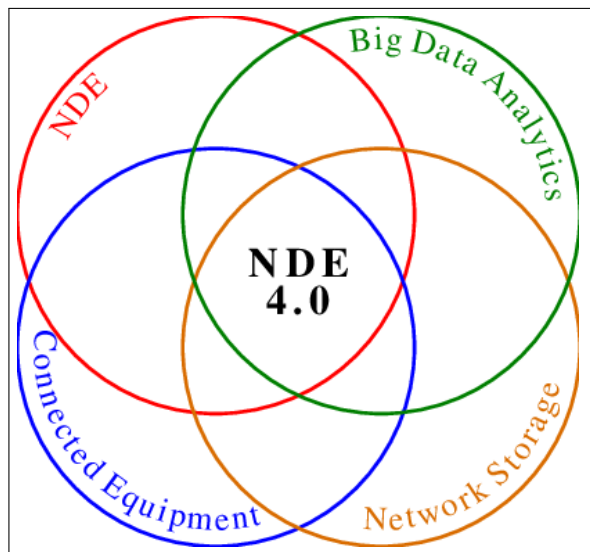


FIGURE 3.39: The essential aspects of NDE 4.0

3.11 Visualizing Data and Info's

Data visualization software is the computing process of turning raw data¹²¹ into visual object, like graphs, maps and other types of that reveal trends, provide analysis and predict outcomes for all kinds of projects.

3.11.1 What Maritime info's (KPIs) we Visualize?

The **Shipping KPI System**¹²² is a global tool for defining, measuring and reporting information on a ship's operational performance in order to:

- Boost performance improvements with companies engaged in ship operation activities
- Provide an efficient communication platform on ship operation performance to internal and external stakeholders

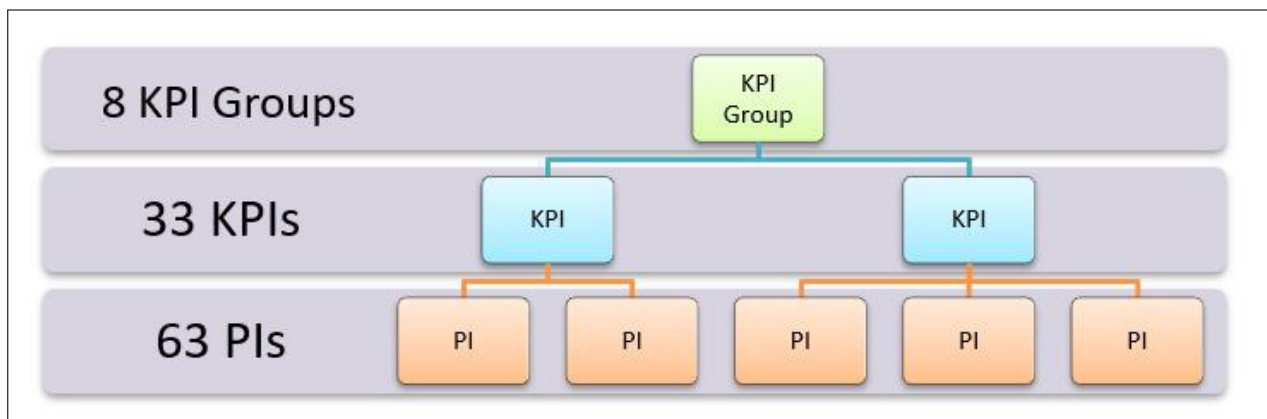


FIGURE 3.40:
Hierarchy of Shipping KPI's. (by "shipping-kpi[dot]org")

The benchmarking, monitoring, reporting tool comprising Key Performance Indicators (KPI) and Performance Indicators (PI) has been developed. was developed and launched back in 2010. The Shipping KPI Standard is built up hierarchical with 3 different levels:

- Key Performance Indicators groups (KPI groups)
- Key Performance Indicators (KPIs)
- Performance Indicators (PIs)

The **Shipping KPI Costs** is an additional feature added to establish the cost of a ship's/fleet's performance. The Costs comprise of Cost Categories (CC) and Cost Item (CI) to capture the cost information. A parameter of Cost Factor (CF) and Factor Item (FI) have been added to support in calculating the cost per day of each Cost Category.

The Last Release, of "**Shipping KPI Standard V4.0**" Launched September 2020, with additional feature's "Shipping KPI Group on Costs" by BIMCO Organization.¹²³

¹²¹Raw data, also known as primary data, are data (e.g., numbers, instrument readings, figures, etc.) collected from a source.

¹²²A simple definition for SPI's, is "a predifining and standarized common language for maritime vessel performance monitoring".

¹²³<https://www.bimco.org/ships-ports-and-voyage-planning/shipping-kpi-system>

3.11.2 Visual Shipping KPI's Dashboard.

A Dashboard is an information management tool¹²⁴ that visually tracks, analyzes and displays Key Performance Indicators (KPI), metrics and key data points to monitor the health of a business, department or specific process.

They are customizable to meet the specific needs of application. Behind the images, a dashboard connects to our files, attachments, services and API's, but on the surface displays all this data in the form of tables, line charts, bar charts and gauges.

A dashboard is the most efficient choice to track multiple data sources because it provides a central location for businesses to monitor and analyze performance. Real-time monitoring reduces the hours of analyzing and long line of communication that previously challenged businesses.

Most businesses and in shipping, use multiple services to track KPIs and metrics, which takes up time and resources to properly monitor and analyze.

Dashboards use raw data from spreadsheets, databases and rest data sources to create tables, line and/or bar charts and gauges in a central dashboard that users can look at and immediately understand the key metrics they are looking for. Data dashboards also simplify end of a period (duty/day/week/month/ year etc.) reporting by allowing users to communicate information at any time without hours of preparation and analyzing.



FIGURE 3.41: Live Vessel Performance Monitoring. (by navigationlaptops.com)

¹²⁴Live Dashboard Examples : <https://www.klipfolio.com/live-dashboards>

3.11.3 AI Agent's

In artificial intelligence, an intelligent agent (IA) refers to an autonomous entity which acts, directing its activity towards achieving goals (i.e. it is an agent), upon an environment using observation through sensors and consequent actuators (i.e. it is intelligent). Intelligent agents may also learn or use knowledge to achieve their goals.[134, 164]

They may be very simple or very complex. A reflex machine, such as a thermostat, is considered an example of an intelligent agent.

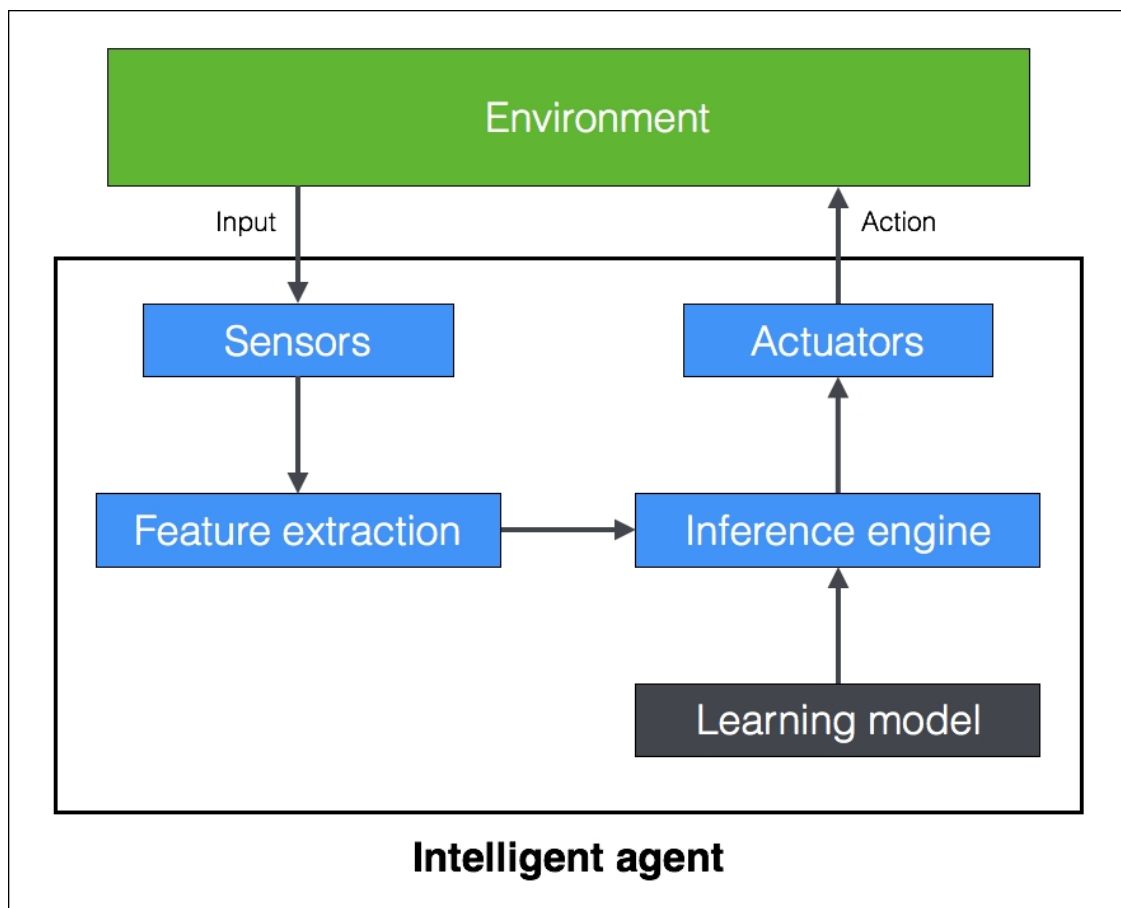


FIGURE 3.42:
A Intelligent Agent. by oreilly.com

Intelligent agents are often described schematically as an abstract functional system similar to a computer program. Researchers such as Russell & Norvig (2003) consider goal-directed behavior to be the essence of intelligence; a normative agent can be labeled with a term borrowed from economics, "rational agent". In this rational-action paradigm, an IA possesses an internal "model" of its environment. This model encapsulates all the agent's beliefs about the world. The agent also has an "objective function" that encapsulates all the IA's goals. Such an agent is designed to create and execute whatever plan will, upon completion, maximize the expected value of the objective function. A reinforcement learning agent can have a "reward function" that allows the programmers to shape the IA's desired behavior, and an evolutionary algorithm's behavior is shaped by a "fitness function". Abstract descriptions of intelligent agents are sometimes called abstract intelligent agents (AIA) to distinguish them from their real world implementations as computer systems, biological systems, or organizations. Some autonomous intelligent agents are designed to function in the absence of human intervention. As intelligent agents become more popular, there are increasing legal risks involved.

Intelligent agents in artificial intelligence are closely related to agents in economics, and versions of the intelligent agent paradigm are studied in cognitive science, ethics, the philosophy of practical reason, as well as in many interdisciplinary socio-cognitive modeling and computer social simulations.

Intelligent agents are also closely related to software agents (an autonomous computer program that carries out tasks on behalf of users). In computer science, an intelligent agent is a software agent that has some intelligence, for example, autonomous programs used for operator assistance or data mining (sometimes referred to as bots) are also called "intelligent agents."

Building an intelligent agent There are many ways to impart intelligence to an agent. The most commonly used techniques include **machine learning, stored knowledge, rules, and so on**. In Vessel Performance Monitoring, using multiple techniques together.

3.12 Communication Systems

3.12.1 Vessels communications planning.

When choice the combination of communication systems for ships, operators will have three key requirements: it should be sufficient to address needs, it must be reliable and it should be cost effective.[166]

3.12.2 Maritime Communication Systems.

Marine communication has long history and continues developing to this day. After the times of semaphores and banners (which is as yet pertinent today at times) radio brought about a drastic change in marine communication at sea. [18]

GMDSS Overview.

In 1979, a group of experts drafted the International Convention on Maritime Search and Rescue, which called for the development of a global search and rescue plan. This group also passed a resolution calling for the development of a Global Maritime Distress and Safety System (GMDSS) to provide the communication support needed to implement the global search and rescue plan.

GMDSS consists of several systems which are intended to perform the following functions:

- alerting (including position determination of the ship in distress) ships in the vicinity and ashore authorities
- search and rescue coordination,
- locating (homing),
- maritime safety information broadcasts,
- general communications, and bridge-to-bridge communications.

Specific radio carriage requirements depend upon the ship's area of operation, rather than its tonnage. The system also provides redundant means of distress alerting, and emergency sources of power.

Recreational vessels do not need to comply with GMDSS radio carriage requirements, but will increasingly use the Digital Selective Calling (DSC) Marine VHF radios¹²⁵. Offshore vessels may elect to equip themselves further. [6, 7, 47]

¹²⁵https://continuouswave.com/whaler/reference/DSC_Datagrams.html

Area A1	Area A2	Area A3	Area A4
VHF DSC : 1, 2, 3, 6, 8 VHF Voice : 1, 2, 3, 4, 5, 6, 7, 8, 9	VHF DSC : 3, 6 VHF Voice : 3, 5, 6, 9		
MF DSC : 1, 2, 3, 6 MF Voice : 1, 2, 3, 4, 6, 7	MF DSC : 1, 2, 3, 6, 8 MF Voice : 1, 2, 3, 4, 6, 7, 8	MF DSC : 3 MF Voice : 3	
HF DSC : 1, 2, 3, 6 HF Voice : 1, 2, 3, 4, 6, 7		HF DSC : 1, 2, 3, 6, 8 HF Voice : 1, 2, 3, 4, 6, 7, 8	
HF NBDP : 1, 2, 3, 4, 7, 8 EPIRB : 1, 6 SART : 1, 6		Survival craft VHF : 3, 5 Passenger ship aeronautical radios : 4, 5, 6 NAVTEX Receiver : 7	

1 : Transmit ship-to-shore distress alerts
2: Receive shore-to-ship distress alert relays
3: Transmit and receiving ship-to-ship distress alerts
4: Transmit and receive search and rescue coordinating communications
5: Transmit and receive on-scene communications
6: Transmit and receive signals for locating
7 : Receive Maritime safety information(MSI)
8 : Transmit and receive general communications
9: Transmit and receive bridge-to-bridge communications

FIGURE 3.43: Functional Requirement of GMDSS(IMO, 2013)

The main types of equipment used in GMDSS are: ¹²⁶

- Emergency position-indicating radio beacon (EPIRB) via COSPAS/SARSAT,
- NAVTEX
- Satellite Communication System (Inmarsat),
- High Frequency(HF/VHF) Radiotelephone with Digital Selective Calling (DSC),
- Search and rescue locating device (SART).

GMDSS sea areas. Different radio communication systems are required (in minimum) by the vessels, depending on the area of navigating.

- Sea Area A1—An area within the radiotelephone coverage of at least one VHF coast station in which continuous digital selective calling (Ch.70/156.525 MHz) alerting and radiotelephony services are available. Such an area could extend typically 30 to 40 nautical miles (56 to 74 km) from the Coast Station.
- Sea area A2—An area within a coverage of at least one coast station continuous listening on MF (2187.5 kHz) other than Area A1,

¹²⁶The Maritime Safety Information (MSI) service is an internationally co-ordinated network of broadcasts of Maritime Safety Information. This information contains: Navigational warnings; Meteorological information (forecasts and warnings); and Distress alerts.

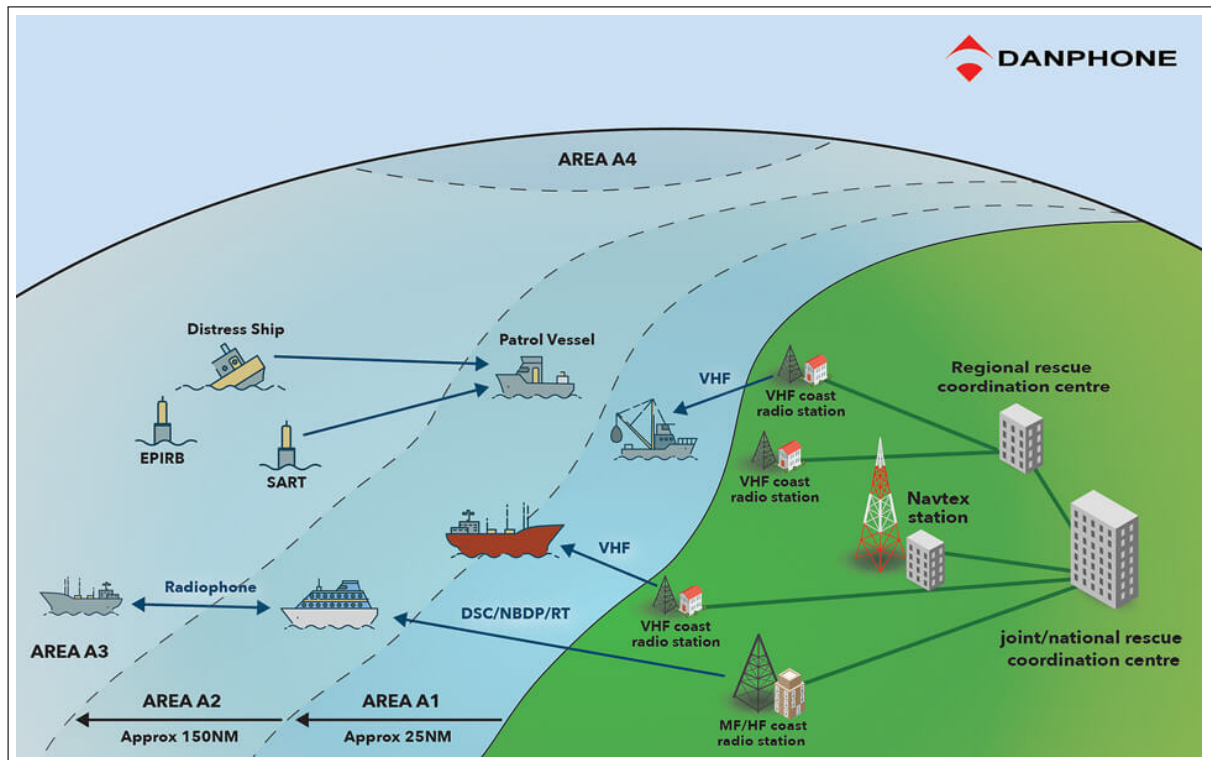


FIGURE 3.44: GMDSS Areas. (by danphone.com)

- Sea Area A3—An area, excluding sea areas A1 and A2, within the coverage of an Inmarsat geostationary satellite. This area lies between about latitude 76 Degrees North and South, but excludes A1 and/or A2 designated areas. Inmarsat guarantees their system will work between 70 South and 70 North though it will often work to 76 degrees South or North.
- Sea Area A4—An area outside Sea Areas A1, A2 and A3 is called Sea Area A4. This is essentially the polar regions, north and south of about 76 degrees of latitude, excluding any A1, A2 and A3 areas.

In addition to equipment listed, all GMDSS-regulated ships must carry a satellite EPIRB, a NAVTEX receiver (if they travel in any areas served by NAVTEX), an Inmarsat-C SafetyNET receiver (if they travel in any areas not served by NAVTEX), a DSC-equipped VHF radiotelephone, two (if between 300 and less than 500 GRT) or three VHF handhelds (if 500 GRT or more), and two 9 GHz search and rescue radar transponders (SART).

Maritime Mobile Service Identity (MMSI).

A **Maritime Mobile Service Identity (MMSI)** is a series of nine digits which are sent in digital form over a radio frequency channel in order to uniquely identify ship stations, ship earth stations, coast stations, coast earth stations, and group calls. These identities are formed in such a way that the identity or part there of can be used by telephone and telex subscribers connected to the general telecommunications network to call ships automatically. ^{127 128 129}

¹²⁷ https://en.wikipedia.org/wiki/Maritime_Mobile_Service_Identity

¹²⁸ https://www.ic.gc.ca/eic/site/smt-gst.nsf/eng/h_sf06198.html

¹²⁹ <https://www.amsa.gov.au/safety-navigation/distress-beacons/about-maritime-mobile-service-identity-informati>

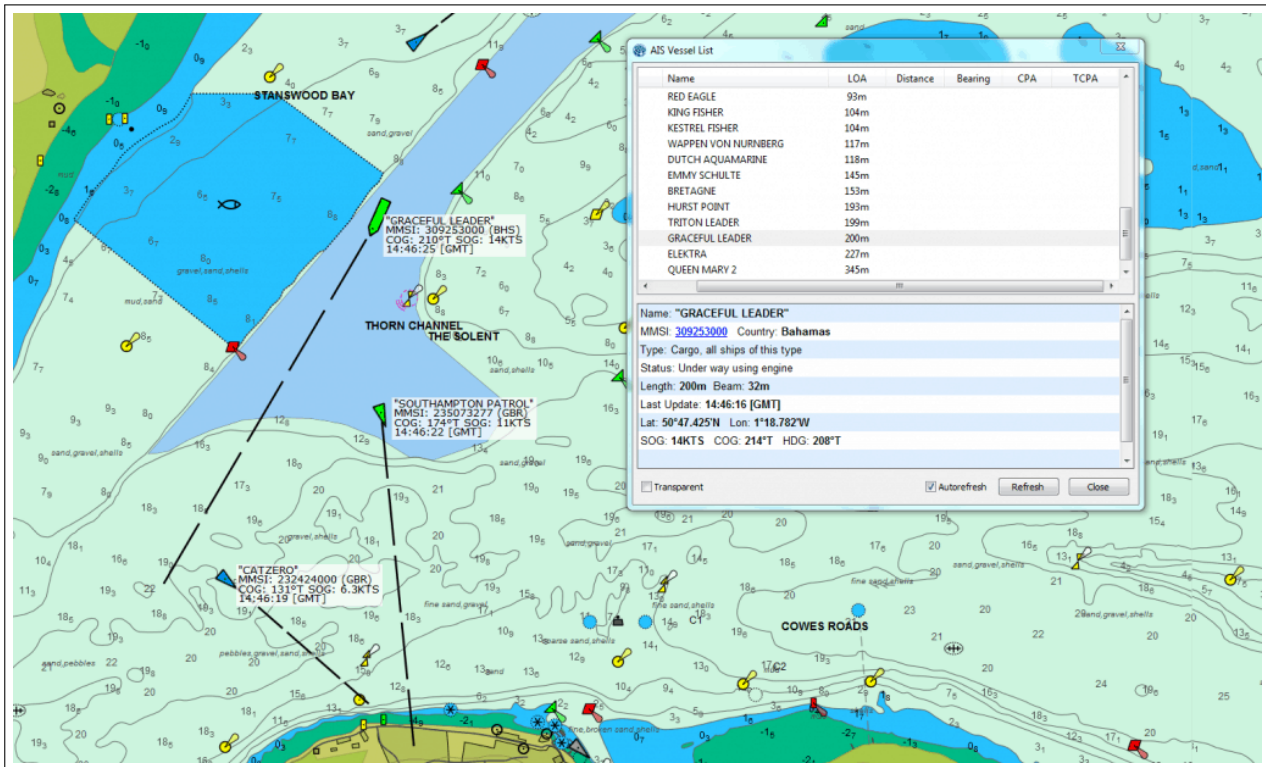


FIGURE 3.45: AIS ECDIS MMSI (by www.collins-marine.co.uk)

3.12.3 International Mobile Satellite Organization (IMSO).

The International Mobile Satellite Organization (IMSO)¹³⁰ is the intergovernmental organization that oversees certain public satellite safety and security communication services provided via the Inmarsat satellites. Some of these services concern:

- Global Maritime Distress Safety System (GMDSS) established by the International Maritime Organization (IMO)
- Search and rescue co-ordinating communications
- Maritime safety information (MSI) broadcasts
- Aeronautical mobile satellite (route) service, or AMS(R)S, through compliance with the Standards and Recommended Practices (SARPs) established by the International Civil Aviation Organization (ICAO)
- General communications

IMSO also serves as the Coordinator for the Long Range Identification and Tracking of Ships (LRIT), appointed by the Safety of Life at Sea (SOLAS) party States at IMO to ensure the worldwide operation of the system.

¹³⁰<https://imso.org/>

Inmarsat.

Inmarsat is a British satellite telecommunications company¹³¹, offering global mobile services. It provides telephone and data services to users worldwide, via portable or mobile terminals which communicate with ground stations through fourteen geostationary telecommunications satellites. Inmarsat's network provides communications services to a range of governments, aid agencies, media outlets and businesses (especially in the shipping, airline and mining industries) with a need to communicate in remote regions or where there is no reliable terrestrial network.

Iridium.

Iridium is a global satellite communications company¹³², providing access to voice and data services anywhere on Earth. With its constellation of satellites, Iridium's network connects people and devices in the world's most remote places — and close to home.

Iridium - GMDSS: After January 2020, there are two certified providers of GMDSS satellite services: **Inmarsat**, with commsats in equatorial geosynchronous orbits, and **Iridium Communications**, with their 66-satellite constellation in low-Earth orbit (LEO) that can cover higher latitudes and operate with lower communications latency.

The certification of Iridium in 2020 ended a monopoly on the provision of the satellite-based portion of maritime distress services that had previously been held by Inmarsat since the system became operational in 1999.^{133 134}

3.12.4 LTE/4G and 5G in Maritime Connectivity.

With the ongoing developments in 4G/LTE networks, Router Equipment and Antennas, vessels can now comfortably connect more than 20 nautical miles offshore. Furthermore, in countries where operators are using new LTE bands in the 600-800 MHz (e.g. Australia, Netherlands and North Sea area), the range can double!¹³⁵

4G/LTE services deliver fast bandwidth with low latency, as long as vessels remain within network coverage. Outside this coverage, vessels need to revert to satellite communications. [78, 117, 107, 115, 137]

3.12.5 Internet

The most common mode of providing internet¹³⁶ on ships is through satellite¹³⁷. Providing a direct connection with the satellite services through some hardware installations on the ship, easy access to internet can be provided for all the on boarders. The hardware installations are a must so that internet signals can be tapped from anywhere.¹³⁸

¹³¹<https://www.inmarsat.com>

¹³²<https://www.iridium.com/>

¹³³https://en.wikipedia.org/wiki/Global_Maritime_Distress_and_Safety_System

¹³⁴<https://www.nasaspaceflight.com/2020/01/iridium-milestone-maritime-safety-breaks-monopoly/>

¹³⁵<https://marpoint.gr/4g-lte-answering-top-5-questions/>

¹³⁶The Internet is the global system of interconnected computer networks that uses the Internet protocol suite (TCP/IP) to communicate between networks and devices

¹³⁷Alternative connectivity for coast navigation routes, via 4G/LTE-5G Networks

¹³⁸<https://www.marineinsight.com/life-at-sea/maritime-internet-options-how-is-internet-provided-on-ships/>

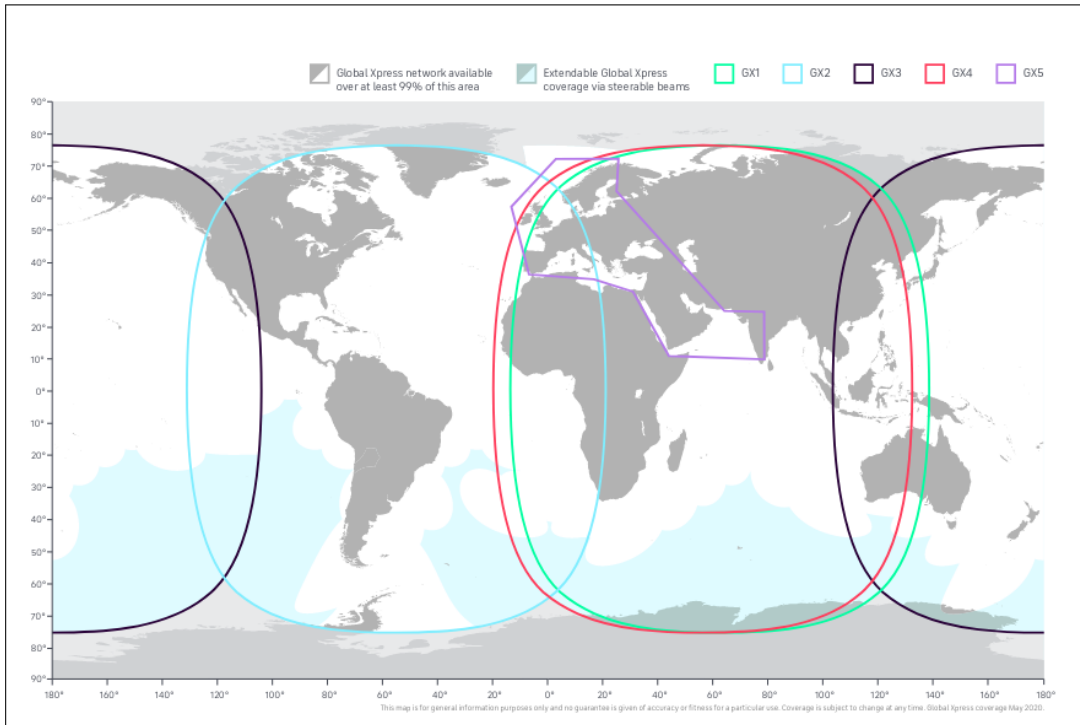


FIGURE 3.46: Inmarsat Global Xpress. Ka-Band.

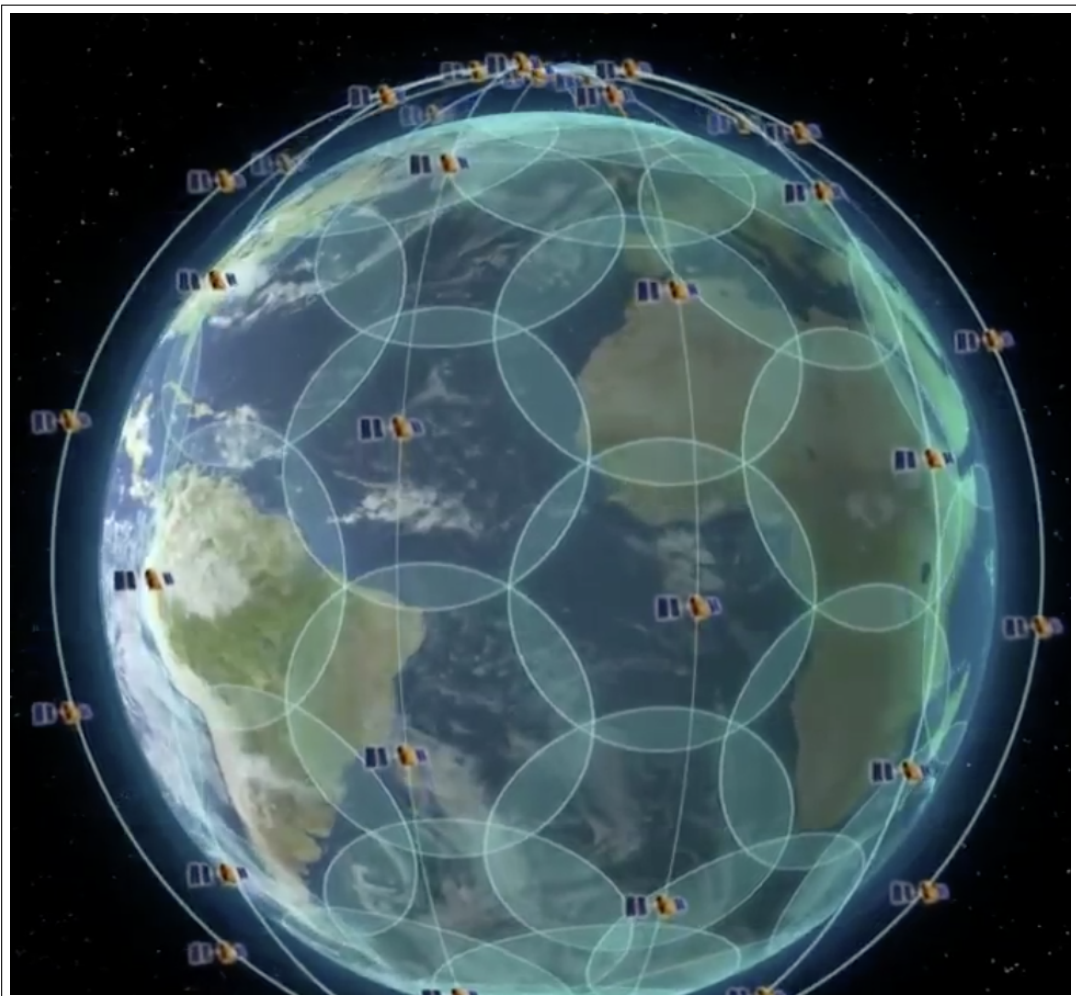


FIGURE 3.47: Iridium constellation coverage footprint's, by Mobile Internet Resource Center.

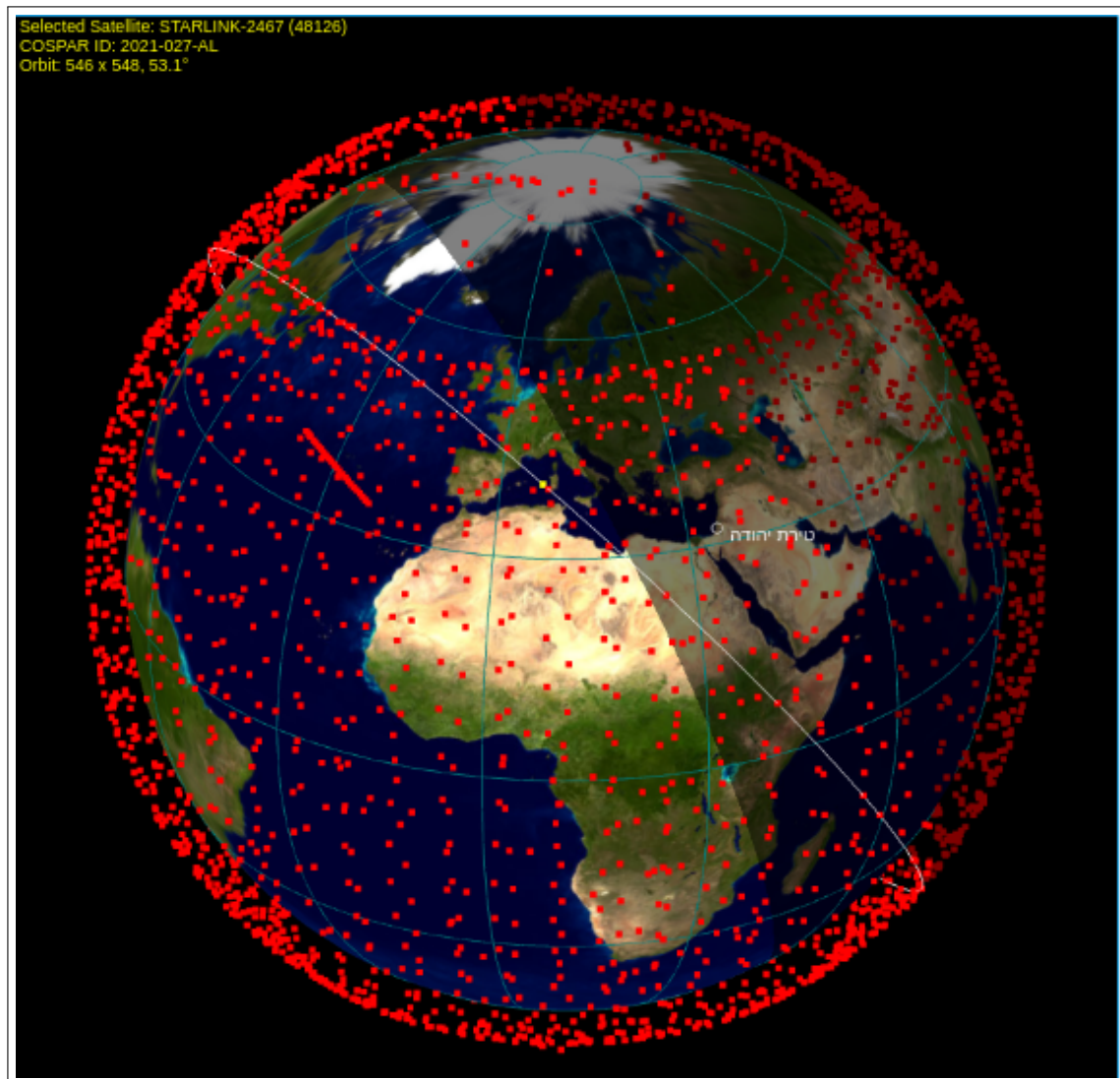


FIGURE 3.48: Starlink Constellation by <https://satellitemap.space>

3.12.6 Starlink

Starlink¹³⁹ is a satellite internet constellation operated by SpaceX^{140 141}, providing satellite Internet access coverage to 45 countries. It also aims for global mobile phone service after 2023. SpaceX started launching Starlink satellites in 2019. As of December 2022, Starlink consists of over 3,300 mass-produced small satellites in low Earth orbit (LEO), which communicate with designated ground transceivers.

Starlink Maritime. From merchant vessels, oil rigs to premium yachts, Starlink Maritime¹⁴² allow to connect from the most remote waters across the world, just like in the office or at home.

¹³⁹<https://www.starlink.com/>

¹⁴⁰Space Exploration Technologies Corp. (SpaceX) is an American spacecraft manufacturer, launcher, and a satellite communications corporation headquartered in Hawthorne, California.

¹⁴¹<https://www.spacex.com>

¹⁴²<https://www.starlink.com/maritime>

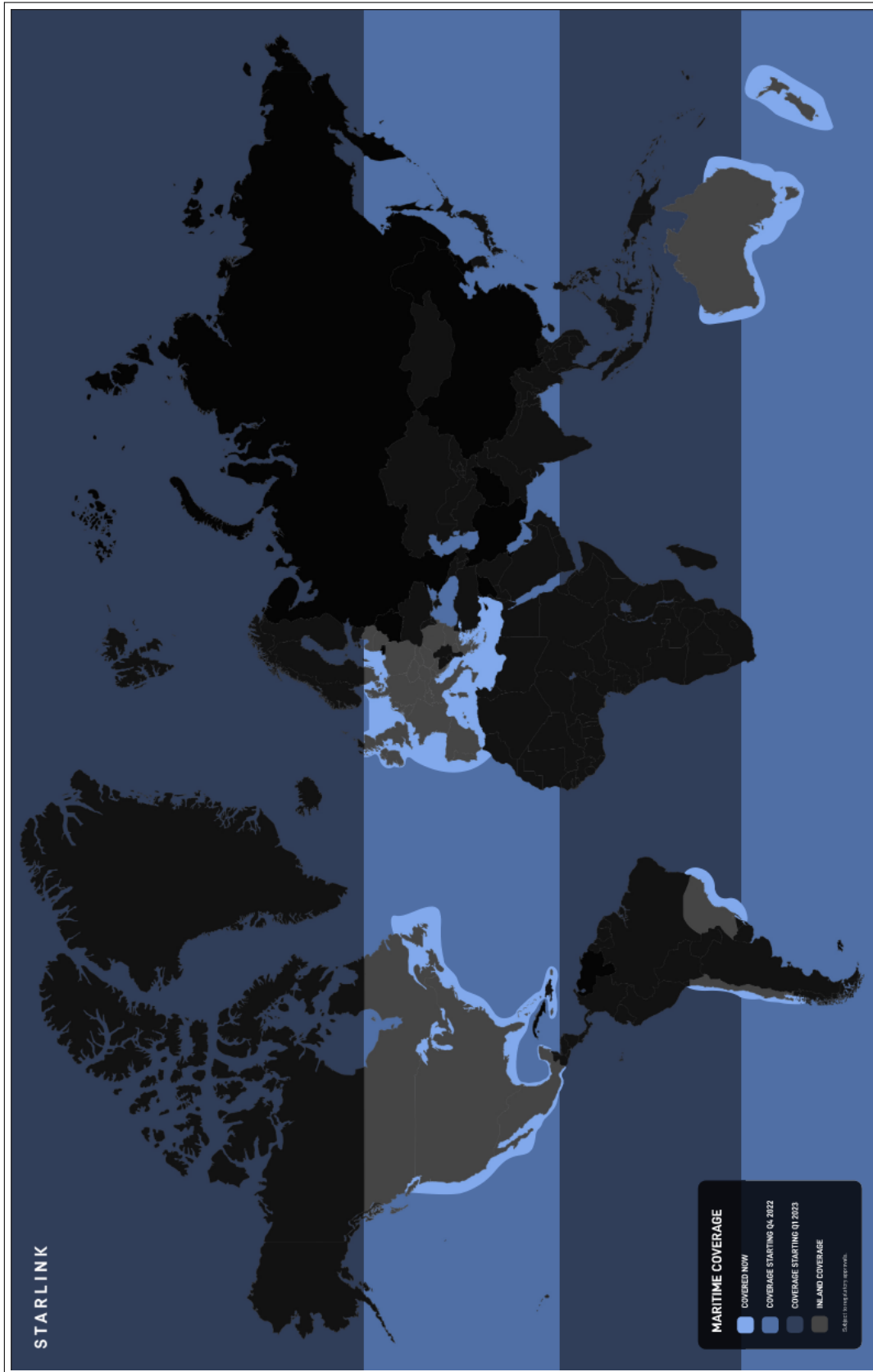


FIGURE 3.49: Starlink Maritime Coverage Map.

Communication Provider.	Min-max Latency.	Height from Earth Surface
Inmarsat	750-1300 msec	35.800Km
Iridium	100-150 msec	780Km
Starlink	45-80 msec	500Km
LTE/4G -5G	20-35 msec	5-1500m
Fiber Optic Networks	10 msec	1000Km
* Node's Processing Time	1-4 msec	

FIGURE 3.50: Latency Time from Maritime Communications Systems.

3.12.7 Latency

Latency¹⁴³ is physically a consequence of the limited velocity at which any physical interaction can propagate. The magnitude of this velocity is always less than or equal to the speed of light. Therefore, every physical system with any physical separation (distance) between cause and effect will experience some sort of latency, regardless of the nature of the stimulation at which it has been exposed to.

Communication Latency In communications, the lower limit of latency is determined by the medium being used to transfer information. In reliable two-way communication systems, latency limits the maximum rate that information can be transmitted, as there is often a limit on the amount of information that is "in-flight" at any given moment. Perceptible latency has a strong effect on user satisfaction and usability in the field of human-machine interaction.

Satellite's Transmission Latency Satellites¹⁴⁴ in geostationary orbits are far enough away from Earth that communication latency becomes significant¹⁴⁵ from one Earth station to another and then back to the first. Low Earth orbit is sometimes used to cut this delay, at the expense of more complicated satellite tracking on the ground and requiring more satellites in the satellite constellation to ensure continuous coverage.

Audio/video latency, is the delay between when a signal enters and when it emerges from a system.

Network latency,¹⁴⁶ in a packet-switched network is measured as either one-way, or round-trip delay time. Round-trip latency is more often quoted, because it can be measured from a single point.

¹⁴³[https://en.wikipedia.org/wiki/Latency_\(engineering\)](https://en.wikipedia.org/wiki/Latency_(engineering))

¹⁴⁴<https://satoms.com/satellite-latency/>

¹⁴⁵A quarter of a second for a trip from one ground-based transmitter to the satellite and back to another ground-based transmitter; close to half a second for two-way communication..!!

¹⁴⁶<https://www.satelliteinternet.com/resources/what-is-internet-latency/>

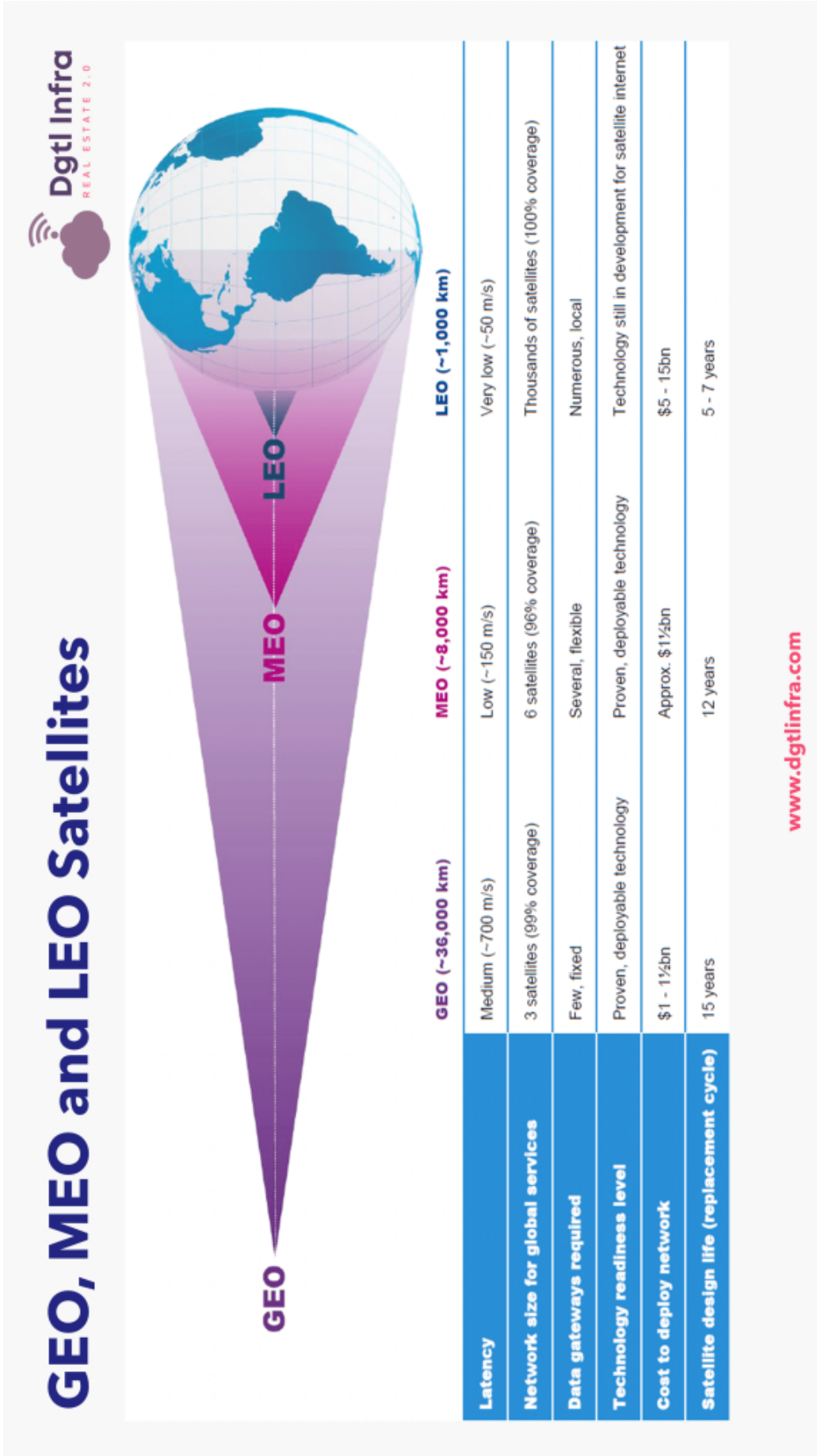


FIGURE 3.51: Comparison of Latency for basic Satellite Types. By dgtlinfra [dot] com

Chapter 4

Collecting, Manipulating and Compute Data for VPM.

The Data of the Vessel ,from IOT, for VPM

In this chapter, I present briefly definitions About Data (as data), Data Manipulating and the relative Computing Definitions.

4.1 About Data.

4.1.1 Data Definitions.

Data refers to information's. This Information's are individual facts, statistics, or items of information, often numeric and / or strings . In a more technical sense, data are a set of values of qualitative or quantitative variables about one or more persons or objects, while a datum (singular of data) is a single value of a single variable.

Data are measured, collected, reported, analyzed and used to create data visualizations¹⁴⁷.

4.1.2 Big Data.

The definition of big data is data that contains greater variety, arriving in increasing volumes and with more velocity¹⁴⁸. It can be defined as data sets whose size or type is beyond the ability of traditional relational databases to capture, manage and process the data with low latency. Characteristics of big data include high volume, high velocity and high variety. Sources of data are becoming more complex than those for traditional data because they are being driven by artificial intelligence (AI), mobile devices, social media and the Internet of Things (IoT). For example the different types of data originate from sensors, devices, video/audio, networks, log files, transactional applications, web and social media — much of it generated in real time and at a very large scale¹⁴⁹.

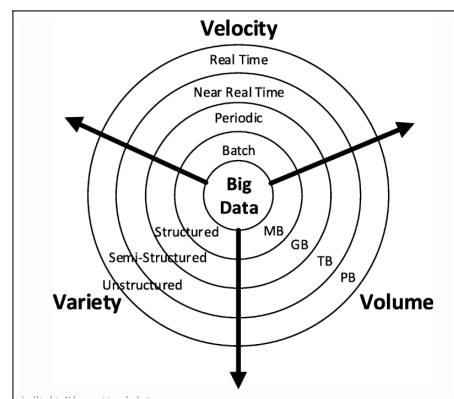


FIGURE 4.1: Big Data 3V's.(by springer.com)

¹⁴⁷<https://en.wikipedia.org/wiki/Data>

¹⁴⁸<https://www.oracle.com/big-data/what-is-big-data/>

¹⁴⁹<https://www.ibm.com/analytics/hadoop/big-data-analytics>

4.1.3 Big Data Characteristics.

Big data have the following characteristics¹⁵⁰:

Volume, is the quantity of generated and stored data. The size of the data determines the value and potential insight, and whether it can be considered big data or not. The size of big data is usually larger than terabytes and petabytes.

Velocity, is the speed at which the data is generated and processed to meet the demands and challenges that lie in the path of growth and development. Big data is often available in real-time. Compared to small data, big data is produced more continually. Two kinds of velocity related to big data are **the frequency of generation** and **the frequency of handling, recording, and publishing**

Variety, refers to the many types of data that are available. Traditional data types were structured and fit neatly in a relational database. With the rise of big data, data comes in new unstructured data types. Unstructured and semi-structured data types, such as text, audio, and video, require additional preprocessing to derive meaning and support metadata.

4.1.4 The Collected Data Types.

There are three main types of data that we meeting Numeric, Text and Logic-state (Boolean) data. Also there is and multimedia data type.¹⁵¹

Numeric data types include integers and floats¹⁵². A floating point (known as a float) number has decimal points even if that decimal point value is 0. (eg. 1.0, 0.12 etc)

Text data type is known as Strings . Strings can contain numbers and / or characters. For example, a string might be a word, a sentence¹⁵³, or several sentences¹⁵⁴.

Logic-state type is known as Boolean . In computer science, the Boolean data type¹⁵⁵ is a data type that has one of two possible values (usually denoted true and false) which is intended to represent the two truth values of logic and Boolean algebra.

Multimedia data refers to data that consist of various media types like text, audio, video, and animation etc., are typically the elements for the building blocks of the generalized multimedia environments, platforms, or integrating tools.^{156 157}

¹⁵⁰https://en.wikipedia.org/wiki/Big_data

¹⁵¹<https://datacarpentry.org/python-ecology-lesson/04-data-types-and-format/>

¹⁵²Both in decimal and binary base

¹⁵³NMEA sentence.

¹⁵⁴JSON, XML, CSV or SQL data files.

¹⁵⁵https://en.wikipedia.org/wiki/Boolean_data_type

¹⁵⁶Video/Web camera, Radar, Ecdis,e-mails etc..

¹⁵⁷<https://www.igi-global.com/dictionary/towards-multimedia-digital-libraries/19578>

4.2 Data Manipulating (Managing)

4.2.1 Data Managing general overview.

Data management is the process of collecting, cleaning, storing, analyzing and using data securely, efficiently, and cost-effectively. The purpose of data management is to drive the optimization of the use of data, so that the authorized persons can make decisions and take actions that maximize the benefit to the Data Driven Organization/Company/Vessel.

4.2.2 IoMaT Collecting Big Data from Multi-sources.

Machine's data created from IoT constitute a valuable source of big data.¹⁵⁸ This data is usually generated from the smart sensors that are embedded to electronic devices. The sourcing capacity depends on the ability of the sensors to provide real-time accurate information. IoMaT is now gaining momentum and includes big data generated from own Vessel ecosystem. [89]

4.2.3 Raw Data ("unprocessed").

Raw data refers to tables of data where each row contains an observation and each column represents a variable that describes some property of each observation. Raw data is the data that is collected from a source, but in its initial state. It has not yet been processed — or cleaned, organized, and visually presented. We can find raw data in a variety of places, including databases, files, spreadsheets, and even on source devices, such as a camera.^{159 160}

4.2.4 Cleaning Data.

Data cleaning is the streamlining process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset at the collecting/ fetching this dataset . When combining multiple data sources, there are many duplicated or mislabeled data. If data is incorrect, outcomes and algorithms are unreliable, even though they may look correct.

Cleansing data is an integral part of data processing and maintaining. This has lead to the development of a broad range of methods intending to enhance the accuracy and thereby the usability of existing data.

There is no one absolute way to prescribe the exact steps in the data cleaning process because the processes will vary from dataset to dataset. But it is crucial to establish a template for our data cleaning for right processing way .^{161 162 163}

We can follow next basic steps to establish out a framework for digital-ship.[9]

1. Remove duplicate or irrelevant observations
2. Fix structural errors
3. Filter unwanted outliers
4. Handle missing data
5. Validate the data before actually presenting

¹⁵⁸<https://www.allerin.com/blog/top-5-sources-of-big-data>

¹⁵⁹<https://www.displayr.com/what-is-raw-data/>

¹⁶⁰<https://www.datamation.com/big-data/raw-data/>

¹⁶¹<https://www.tableau.com/learn/articles/what-is-data-cleaning>

¹⁶²<https://www.iteratorshq.com/blog/data-cleaning-in-5-easy-steps/>

¹⁶³<https://www.codemotion.com/magazine/dev-hub/big-data-analyst/data-cleaning/>

4.2.5 Data Analytics.

Data analytics refers to the process and practice of analyzing data to answer questions, extract insights, and identify trends. This is done using an array of tools, techniques, and frameworks that vary depending on the type of analysis being conducted.[157]

The four major types of analytics include:^{164 165}

Descriptive analytics, which looks at data to examine, understand, and describe something that's already happened.

Diagnostic analytics, which goes deeper than descriptive analytics by seeking to understand the why behind what happened.

Predictive analytics, which relies on historical data, past trends, and assumptions to answer questions about what will happen in the future.

Prescriptive analytics, which aims to identify specific actions that an individual or organization should take to reach future targets or goals.

4.2.6 Presenting Data in VPM.

The visualization of present and past data is done through the "Dashboard Panel" systems. With various graphs 4.2.

For example:

- histograms,
- lines,
- pies,
- bar & column graphs,
- radar,
- scatter,
- bubble etc.

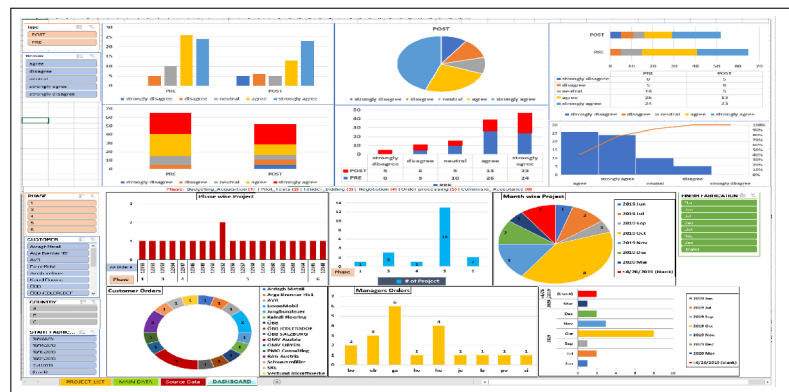


FIGURE 4.2: General View of Dashboard Graphs.
(by fiverr.com)

¹⁶⁴<https://online.hbs.edu/blog/post/data-analytics-vs-data-science>

¹⁶⁵<https://www.investopedia.com/terms/d/data-analytics.asp>

4.3 Computing Definitions.

4.3.1 Computing and Super Computing.

Computing, is activity requiring, benefiting from, or creating computing machinery. It includes the study and experimentation of algorithmic processes and development of both hardware and software. It has scientific, engineering, mathematical, technological and social aspects. Major computing disciplines include computer engineering, computer science, cyber-security, data science, information systems, information technology and software engineering¹⁶⁶.

Supercomputing, term refers to the processing of massively complex or data-laden problems using the concentrated compute resources of multiple computer systems working in parallel (i.e. a "supercomputer"). Supercomputing involves a system working at the maximum potential performance of any computer, typically measured in Petaflops. Sample use cases include weather, energy, life sciences, manufacturing and transportation etc..¹⁶⁷

4.3.2 Cloud Computing

Cloud computing, is computing resources services through the Internet. These resources include hardware and software, tools and applications like data storage, servers, databases, networking, software high speed processing capabilities.¹⁶⁸ [26, 33, 52, 54, 71, 99, 103, 120]

Cloud computing it's a system primarily comprised of three services:

- **Software-as-a-service (SaaS)** involves the licensure of a software application to customers. Licenses are typically provided through a pay-as-you-go model or on-demand.¹⁶⁹
- **Infrastructure-as-a-service (IaaS)** involves a method for delivering everything from operating systems to servers and storage through IP-based connectivity as part of an on-demand service. Clients can avoid the need to purchase software or servers, and instead procure these resources in an outsourced, on-demand service.¹⁷⁰
- **Platform-as-a-service (PaaS)** is considered the most complex of the three layers of cloud-based computing. PaaS shares some similarities with SaaS, the primary difference being that instead of delivering software online, it is actually a platform for creating software that is delivered via the Internet.¹⁷¹

Cloud Computing Advantages ^{172 173}:

- **Easy implementation.** Cloud hosting allows business to retain the same applications and business processes without having to deal with the backend technicalities.
- **Accessibility.** Access your data anywhere, anytime. This allows for easy collaboration and sharing among users in multiple locations.

¹⁶⁶<https://en.wikipedia.org/wiki/Computing>

¹⁶⁷https://www.hpe.com/emea_europe/en/what-is/supercomputing.html

¹⁶⁸<https://www.investopedia.com/terms/c/cloud-computing.asp>

¹⁶⁹This type of system can be found in Microsoft Office's 365.

¹⁷⁰Popular examples of the IaaS system include IBM Cloud and Microsoft Azure.

¹⁷¹This model includes platforms like Salesforce.com and Heroku.

¹⁷²<https://www.stratospherenetworks.com/advantages-and-disadvantages-of-cloud.html>

¹⁷³<https://interfaz.io/cloud-computing-which-are-its-advantages-and-disadvantages/?lang=en>

- **No hardware required.** Since everything will be hosted in the cloud, a physical storage center is no longer needed.
- **Cost per head.** Overhead technology costs are kept at a minimum with cloud hosting services, enabling businesses to use the extra time and resources for improving the company infrastructure.
- **Flexibility for growth.** The cloud is easily scalable so companies can add or subtract resources based on their needs. As companies grow, their system will grow with them.
- **Efficient recovery.** Cloud computing delivers faster and more accurate retrievals of applications and data. With less downtime, it is the most efficient recovery plan.

Disadvantages of Cloud Computing ¹⁷⁴:

- **Performance Can Vary.** When you are working in a cloud environment, your application is running on the server which simultaneously provides resources and to other businesses. Any greedy behavior or DDOS attack on our tenant could affect the performance of your shared resource.
- **Security Threat in the Cloud.** Before adopting cloud technology, we should be well aware of the fact that we will be sharing all our company's (vessel's) sensitive information to a third-party cloud computing service provider. Furthermore, the Continuing On-line connection with Cloud Service Provider could be vulnerable to Hackers Attacks.
- **Downtime.** Downtime should also be considered while working with cloud computing. That's because your cloud provider may face power loss, low internet connectivity, service maintenance, DDOS attacks etc.
- **Lower Bandwidth.** Many cloud storage service providers limit bandwidth usage of their users. So, in case if your organization surpasses the given allowance, the additional charges could be significantly costly.
- **Flexibility for growth.** The cloud is easily scalable so companies can add or subtract resources based on their needs. As companies grow, their system will grow with them.
- **Internet Connectivity in Vessels.** Good Internet connectivity is a must in cloud computing. We can't access cloud without an internet connection. The Interconnection of an IoT, that was installed on a ship, with the world of the Internet and the Cloud Computing. It is usually not technically possible due to geographical changes (voyages), because the average distance from coast's is out of 4G/LTE/5G coverage. And if we have the ability make connection through satellites, **it is very expensive** ¹⁷⁵ and **not cover all globe waters** from the satellite communications providers.

¹⁷⁴<https://www.guru99.com/advantages-disadvantages-cloud-computing.html>

¹⁷⁵eg. Price of 50GB for typical providers: with 4G/LTE 40 euro/month and with maritime sat connection 1000 euro/month (2021)

4.3.3 Vessel Inter-networking and Disadvantages of Cloud Computing.

The difficulties of the internet connectivity on ships, and the other disadvantages as previously mentioned for the Cloud Computing. Driven as the only option, the use of Edge Computing Philosophy for the transition to Shipping 4.0 and the efficient usage of the Internet of Maritime Things (IoMaT) .

4.3.4 Computing and Edge Computing in IoMaT.

Edge computing is a networking computing philosophy. Focused on bringing computing, as close to the source of data as possible. Edge computing means execution of fewer processes in the cloud data centers and moving those processes to local devices, such as on a end user's computer, an IoT device, or an edge server.

Bringing of computation to the network's edge, minimizes the amount of long-distance communication, reduce the latecy of feedback in computation level , increase data security and privacy, consersing networking and computing resources, that has to happen between a edge client and cloud server. Also has options for unlimited scalability. [49, 64, 69, 71, 72, 74, 80, 85, 89, 92, 95, 97, 128, 141, 148]

Chapter 5

Case studies for Edge Computing

In this chapter, I refer in three case studies of IoT systems for VPM, based to Edge Computing Technologies, as an Complete Application. That are Generation of IoMaT & NMEA data, NMEA Gateway Server Application and Web-based IoMaT Dashboard.

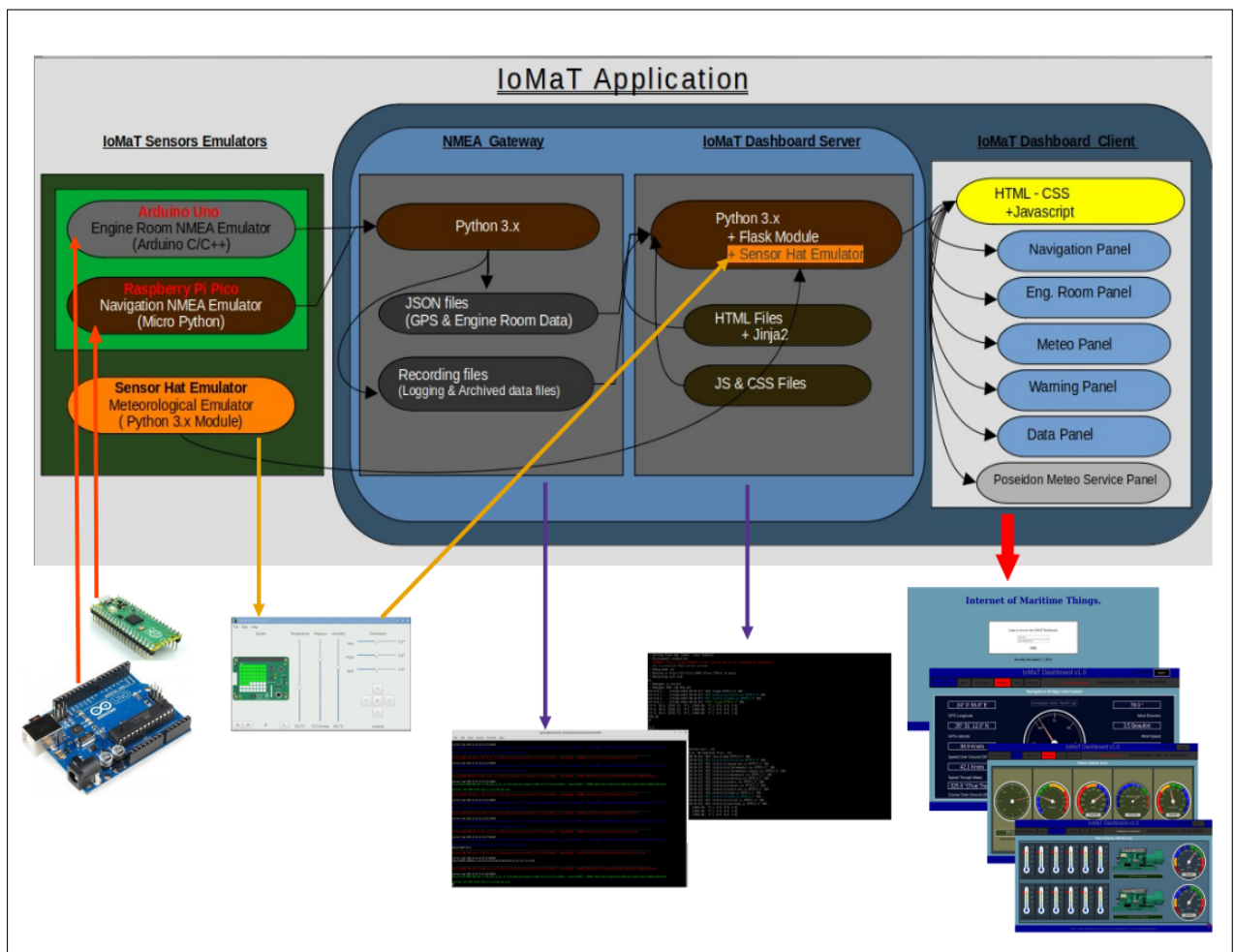


FIGURE 5.1: IoMaT Application Overview Structure

5.1 Case Study 1: Generating IoMaT and NMEA Data.

Creating / generating data (including NMEA Data) for a IoMaT system has three distinct parts. Meteorological data, satellite navigation data and engine room data.

5.1.1 Meteorological Data from Sensor Hat Emulator and Python3 scripting.

On Vessels Meteorological Stations the collected data is on two categories . The first category include : air temperature, wind intensity-direction, air humidity, barometric pressure (auto recording) and coverage rate, types of clouds (visual observation). The second category is the optional collection of data like: temperature - salinity of sea water, intensity and direction of sea currents without limitations for other relative data fields.

In this study, I use for the creation of the basic meteorological measurements the **Sensor Hat Emulator**¹⁷⁶ and **pure Python3 code with Flask Python module**¹⁷⁷.

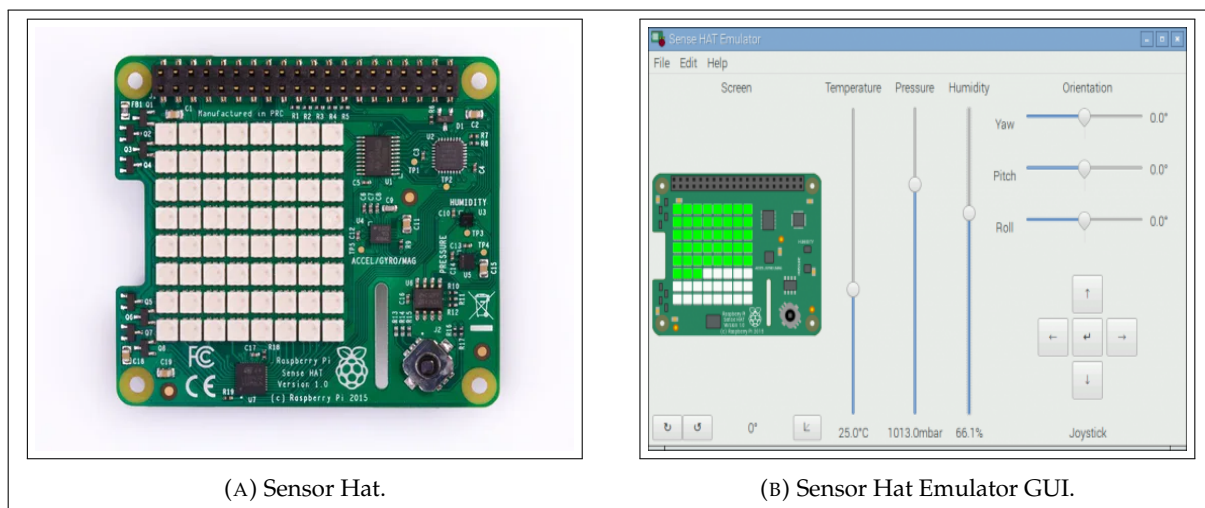


FIGURE 5.2: Hat and Sensor Hat Emulator.(by www.raspberrypi.org)

The Sensor Hat Emulator is a development python-module from **The Raspberry Pi Foundation**¹⁷⁸, which uses the same set of library commands and functions that have been developed for the **Sense HAT**^{179 180}.

In the first code list 5.1, I show the importing from the flask module, the user "global variable" parameter "g"¹⁸¹ and from the sense_emu module of the SenseHat class respectively. And how these elements are combined to get emulation data from the sensor hat to the user parameter "g" of the flask .

¹⁷⁶<https://www.raspberrypi.org/blog/desktop-sense-hat-emulator/>

¹⁷⁷Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. More info's in A.2

¹⁷⁸<https://www.raspberrypi.org/about/>

¹⁷⁹The Sense HAT is an add-on board for Raspberry Pi, made especially for the Astro Pi mission

¹⁸⁰<https://astro-pi.org/about/>

¹⁸¹g is an object for storing data during the application context of a running Flask web app.

```

1 #In IoMaT Server Application Python3 Code with Flask module
2 from flask import g #from module flask importing "global" variable "g"
3 #for application user "object".
4 from sense_emu import SenseHat #importing from "sense_emu"
5 #module the "Sensehat" class
6
7 s = SenseHat() # setting the variable "s"
8 g.s = s # passing the variable "s" to user-object global variable

```

LISTING 5.1: Python 3.7 Import sense_emu module

In the 2nd list 5.2, I show the Jinja2 ¹⁸² code included in the html meteo-page code, to handle the data sent by the code 5.1.

```

1 #In Html code with Jinja2, getting and formatting numeric values for meteo data
2 {{ '%0.0f'|format(g.s.humidity)}}% #Calling function for humidity
3 {{ '%0.1f'|format(g.s.temperature)}}C #Calling function for temperature
4 {{ '%0.0f'|format(g.s.pressure)}} mBar #Calling function for air pressure

```

LISTING 5.2: Using sense-emu methods with Jinja2 in Html code.

With the following 5.3 pure code of Python3 and with flask module, classes and methods. I generate the rest of ship meteo data. First i initialize meteo variables, define wind random speed **wind_s()** and random direction **wind()** function's.

```

1 #Defining meteo emulation parameters
2 #Initializing Meteo Variables
3 windmain = random.randint(0, 359) # Initialize random wind direction
4 sp = random.randint(2, 8) # Initialize random wind speed
5 ws = [0]*100 # Initialize auxiliary table/list
6 av = [0]*100 # Initialize auxiliary table/li
7
8 # Wind Speed Function
9 def wind_s(sp, v=ws, a=av):
10 sp += round(random.uniform(-1, +1), 1)
11 if sp <= 0:
12 sp += 2
13 elif sp >= 12:
14 sp -= 2
15 v.append(sp)
16 print(len(v))
17 v = v[-101:-1]
18 average = round((sum(v[-6:-1])/5), 2)
19 a.append(average)
20 a = a[-101:-1]
21 return [v, a]
22
23 # Wind Direction Function
24 def wind(windmain):
25 windmain = windmain+random.randint(-1, 1)
26 return windmain

```

LISTING 5.3: Python 3.7 pure code for meteo emulating

¹⁸²Jinja is a fast, expressive, extensible templating engine. Special placeholders in the template allow writing code similar to Python syntax. More info's in A.1

5.1.2 Satellite Navigation NMEA Data by Raspberry Pi Pico and Micro Python.

Satellite positioning is the most important part of navigation. In this area, information's transmitted to the vessel's local networks with NMEA sentence's.

My choice to emulate a simple GPS system for generating and transmitting NMEA sentence's. I used the **Raspberry Pi Pico**¹⁸³ microcontroller board with programming in **Micro-Python** language¹⁸⁴. The RPi Pico, transmit¹⁸⁵ the GPS-NMEA sentences 5.3 eg. "\$GPRMC ,191855.645 , A ,3531.137 , N ,02400.980 , E ,038.9 ,325.9 ,100921 ,000.0 , W *7 E".These sentences are in a log file¹⁸⁶ as read by the main program 5.4.

```

1 # Micropython v1.16
2 # "NMEA GPS Sentence's Emulator"
3 # Author: Athanasios G. Ritas
4 # MSc TeleAutoS 2018-2021
5
6 import utime
7 from machine import UART
8
9 #Setting UART port parameters
10 uart1=UART(1,4800)
11
12 #Setting blinking led parameters
13 led=machine.Pin(25, machine.Pin.OUT, machine.Pin.PULL_UP)
14 led.value(0)
15
16 #Main program: reading NMEA records, writing sentences and blinking the led
17 with open("nmea1.txt") as nmea_em:
18     Lines =nmea_em.readlines()
19     count=0
20     for i in range(1000):
21         for line in Lines:
22             count+=1
23             led.toggle()
24             utime.sleep(.1)
25             print(line)
26             led.toggle()
27             utime.sleep(2.1)

```

LISTING 5.4: Micro Python Code for emulating GPS NMEA sentences.

¹⁸³Raspberry Pi Pico is a tiny, fast, and versatile board built using RP2040 processor. More info's in A.3

¹⁸⁴MicroPython is a full implementation of the Python 3 programming language that runs directly on embedded hardware like Raspberry Pi Pico. More info's in A.4

¹⁸⁵Write to UART port, in 4800 baud rate.

¹⁸⁶This file generated from NMEA Generating Organization in url: <https://www.nmeagen.org/>

```

/dev/ttyACM2
Αποστολή
18:34:24.948 -> $GPGSA,A,3,01,02,03,04,05,06,07,08,09,10,11,12,1.0,1.0,1.0*30
18:34:24.948 ->
18:34:27.146 -> $GPRMC,191857.645,A,3531.156,N,02400.967,E,038.9,325.9,100921,000.0,W*72
18:34:27.146 ->
18:34:29.343 -> $GPGGA,191858.645,3531.166,N,02400.961,E,1,12,1.0,0.0,M,0.0,M,,*68
18:34:29.343 ->
18:34:31.543 -> $GPGSA,A,3,01,02,03,04,05,06,07,08,09,10,11,12,1.0,1.0,1.0*30
18:34:31.543 ->
18:34:33.743 -> $GPRMC,191858.645,A,3531.166,N,02400.961,E,038.9,325.9,100921,000.0,W*78
18:34:33.743 ->
18:34:35.937 -> $GPGGA,191859.645,3531.175,N,02400.954,E,1,12,1.0,0.0,M,0.0,M,,*6D
18:34:35.937 ->
18:34:38.137 -> $GPGSA,A,3,01,02,03,04,05,06,07,08,09,10,11,12,1.0,1.0,1.0*30
18:34:38.137 ->
18:34:40.336 -> $GPRMC,191859.645,A,3531.175,N,02400.954,E,038.9,325.9,100921,000.0,W*7D
18:34:40.336 ->
18:34:42.528 -> $GPGGA,191900.645,3531.185,N,02400.948,E,1,12,1.0,0.0,M,0.0,M,,*62
18:34:42.528 ->
18:34:44.755 -> $GPGSA,A,3,01,02,03,04,05,06,07,08,09,10,11,12,1.0,1.0,1.0*30
18:34:44.755 ->
18:34:46.952 -> $GPRMC,191900.645,A,3531.185,N,02400.948,E,038.9,325.9,100921,000.0,W*72
18:34:46.952 ->

 Αυτόματη κύλιση  Επίδειξη χρονοσήμανσης
Αμφότερα NL & CR 4800 baud Εκκαθάριση εξόδου

```

FIGURE 5.3: NMEA Sentences from Satellite Positioning System

```

/dev/ttyACM0
Αποστολή
18:17:12.673 -> $ERMED,0,79.04,78.55,80.02,80.02,79.53,80.02,79.53,*11
18:17:12.772 ->
18:17:13.668 -> $ERMED,1,82.04,81.55,83.02,83.02,82.53,83.02,79.53,*11
18:17:13.768 ->
18:17:15.664 -> $ERMED,0,79.04,79.04,79.53,79.53,79.53,79.53,79.37,*11
18:17:15.797 ->
18:17:16.659 -> $ERMED,1,82.04,82.04,82.53,82.53,82.53,82.53,79.37,*11
18:17:16.791 ->
18:17:18.684 -> $ERMED,0,79.53,79.53,79.53,79.53,79.53,79.53,79.53,*11
18:17:18.783 ->
18:17:19.679 -> $ERMED,1,82.53,82.53,82.53,82.53,82.53,82.53,79.53,*11
18:17:19.779 ->
18:17:21.678 -> $ERMED,0,79.53,79.53,79.53,79.53,80.02,79.53,79.61,*11
18:17:21.777 ->
18:17:22.677 -> $ERMED,1,82.53,82.53,82.53,82.53,83.02,82.53,79.61,*11
18:17:22.777 ->
18:17:24.671 -> $ERMED,0,79.53,80.02,79.53,79.53,79.53,79.04,79.53,*11
18:17:24.804 ->
18:17:25.667 -> $ERMED,1,82.53,83.02,82.53,82.53,82.53,82.04,79.53,*11
18:17:25.802 ->
18:17:27.662 -> $ERMED,0,79.53,79.53,79.53,79.53,79.53,79.53,79.53,*11
18:17:27.794 ->
18:17:28.690 -> $ERMED,1,82.53,82.53,82.53,82.53,82.53,82.53,79.53,*11
18:17:28.790 ->
18:17:30.680 -> $ERMED,0,79.53,79.53,79.53,79.53,79.53,79.53,79.53,*11
18:17:30.780 ->

 Αυτόματη κύλιση  Επίδειξη χρονοσήμανσης
Αμφότερα NL & CR 4800 baud Εκκαθάριση εξόδου

```

FIGURE 5.4: NMEA Sentences form Engine Room (ERMED)

5.1.3 Engine Room NMEA Data by Arduino Uno and Arduino C/Cpp.

Another important area of data collection for evaluation and extraction useful information's, are the Engine's Room.

My choice to emulate an IoT data collection system from the engine room. It relied on the development **Arduino Uno R3 A.5** development board with the **official programming language**¹⁸⁷. The program 5.5 generates pseudo-random temperature values for 2 hypothetical engines (engine 0 and engine 1) of the ship propulsion. Temperature values are including in NMEA sentences messages 5.4 and transmitted (writing on the serial port).

```

1  /* Arduino NMEA Generator
2     Author : Athanasios G. Ritas
3  */
4
5  //.....importing libraries.....
6  #include <SoftwareSerial.h>
7
8  //.....declare millis() check time periods.....
9  unsigned long previousMillis1 = 0; //store last time LED1 was blinked
10 const long period1 = 150;          // period at which led1 blinks in ms
11
12 unsigned long previousMillis2 = 0; //store last time LED2 was blinked
13 const long period2 = 2000;         // period at which led1 blinks in ms
14
15 unsigned long previousMillis3 = 0; //store last time LED2 was blinked
16 const long period3 = 3000;         // period at which led1 blinks in ms
17
18 //.....declare general var for Arduino Uno .....
19 float temp = 0;
20 const int numReadings = 6;
21 int readIndex = 0; // the index of the current reading
22 float total = 0;   // the running total
23 float average = 0, aver_pr = 0;
24 int tempPin = A1;
25 int idx = 0;
26
27 //.....declare Software Serial Port .....
28 SoftwareSerial softserial(2, 3); // RX, TX
29
30 //.....declare var for MNEA msg.....
31 String msg_NMEA = "$", msg_NMEA1 = "$", NMEA_Dev = "ER", NMEAMsgType = "MED,2",
    NMEAMsgType1 = "MED,3", msg_NMEA0 = "", msg_NMEA01 = "", CheckSum = "*11",
    EndmsgNMEA = "<CR><LF>";
32
33 //.....
34
35 void setup()
36 {
37
38     Serial.begin(4800);
39     softserial.begin(4800);
40
41     pinMode(LED_BUILTIN, OUTPUT);
42     digitalWrite(LED_BUILTIN, LOW);
43
44     delay(1000); /// ??
45

```

¹⁸⁷Arduino code is written in C++ with an addition of special methods and functions


```

46 msg_NMEA = msg_NMEA + NMEA_Dev + NMEAMsgType;
47 msg_NMEA1 = msg_NMEA1 + NMEA_Dev + NMEAMsgType1;
48 msg_NMEA0 = msg_NMEA;
49 msg_NMEA01 = msg_NMEA1;
50
51 Serial.println("Initializing Tocken Generator MSGs");
52 Serial.println(msg_NMEA);
53 Serial.println(msg_NMEA1);
54
55 delay(2000);
56 }
57
58 void loop()
59 {
60   unsigned long currentMillis = millis(); // store the current time
61
62   if (currentMillis - previousMillis1 >= period1) // check if 100ms passe
63   {
64     // Serial.println("check#1");
65     previousMillis1 = currentMillis; // save the last time
66     if (readIndex < numReadings)
67     {
68       // read from the sensor:
69       temp = 60 + analogRead(tempPin) * (5000 / 1024.0) / 10;
70
71       // add the reading to the total:
72       total = total + temp;
73       // advance to the next position in the array:
74       readIndex = readIndex + 1;
75       Serial.println(readIndex);
76
77       // Serial.println(numReadings);
78       delay(20);
79       msg_NMEA = msg_NMEA + "," + temp;
80       temp = temp + 3;
81       msg_NMEA1 = msg_NMEA1 + "," + temp;
82     }
83
84   }
85
86
87   if (currentMillis - previousMillis2 >= period2)
88   {
89
90     if (idx == 0) {
91       // calculate the average:
92       average = total / numReadings;
93
94       // send it to the computer as ASCII digits
95       msg_NMEA = msg_NMEA + "," + average + "," + CheckSum+"wemos";
96       msg_NMEA1 = msg_NMEA1 + "," + average + "," + CheckSum;
97
98       //writting MSG's to serial ports
99       Serial.println(msg_NMEA);
100      Serial.println("");
101      // softserial.println("Soft_Serial>>>Arduino Working....!!!");
102      digitalWrite(LED_BUILTIN, HIGH); // turn the LED on
103      delay(50);
104      digitalWrite(LED_BUILTIN, LOW);
105      idx = 1;
106
107    }
108  }

```

```

109 }
110
111 if (currentMillis - previousMillis3 >= period3)
112 {
113
114     previousMillis2 = currentMillis; // save the last time
115     previousMillis3 = currentMillis; // save the last time
116
117     Serial.println(msg_NMEA1);
118     Serial.println("");
119     // softserial.println(msg_NMEA1);
120
121     //reset variables
122     readIndex = 0;
123     msg_NMEA = msg_NMEA0;
124     msg_NMEA1 = msg_NMEA01;
125     // delay(50);
126     total = 0;
127     idx = 0;
128     // aver_pr = average;
129
130 }
131 }

```

LISTING 5.5: Arduino C/C++ code for generating NMEA sentences.

```

agritas@master20: /media/agritas/6236-3963/loMat
File Edit View Search Terminal Help
current_time 2021-12-13 21:11:31.434495
Writing Main Engines NMEA RAW data in Engine_Room_2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:31.434495----/dev/ttyACM0----SERMED,1,82.53,82.53,82.04,82.53,82.53,82.53,79.45,*11
Writing Main Engines JSON format data @ >>>>>nmea_eng_room.json
.....
Writing NMEA RAW data in 2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:33.245028----/dev/ttyACM0----SERMED,0,79.53,79.53,79.53,79.53,80.02,79.53,79.61,*11
current_time 2021-12-13 21:11:33.245028
Writing Main Engines NMEA RAW data in Engine_Room_2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:33.245028----/dev/ttyACM0----SERMED,0,79.53,79.53,79.53,79.53,80.02,79.53,79.61,*11
Writing Main Engines JSON format data @ >>>>>nmea_eng_room.json
.....
Writing NMEA RAW data in 2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:33.533626----/dev/ttyACM1----$GPRMC,192121.645,A,3532.027,N,02402.221,E,038.9,030.9,100921.000,0.0,W*71
current_time 2021-12-13 21:11:33.533626
Writing GPS-NMEA RAW data in GPS_2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:33.533626----/dev/ttyACM1----$GPRMC,192121.645,A,3532.027,N,02402.221,E,038.9,030.9,100921.000,0.0,W*71
Writing GPS JSON format data @ >>>>>nmea_gps.json
.....
Writing NMEA RAW data in 2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:34.244556----/dev/ttyACM0----SERMED,1,82.53,82.53,82.53,82.53,83.02,82.53,79.61,*11
current_time 2021-12-13 21:11:34.244556
Writing Main Engines NMEA RAW data in Engine_Room_2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:34.244556----/dev/ttyACM0----SERMED,1,82.53,82.53,82.53,82.53,83.02,82.53,79.61,*11
Writing Main Engines JSON format data @ >>>>>nmea_eng_room.json
.....
Writing NMEA RAW data in 2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:36.247566----/dev/ttyACM0----SERMED,0,79.53,79.53,79.53,80.02,79.53,79.61,*11
current_time 2021-12-13 21:11:36.247566
Writing Main Engines NMEA RAW data in Engine_Room_2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:36.247566----/dev/ttyACM0----SERMED,0,79.53,79.53,79.53,80.02,79.53,79.61,*11
Writing Main Engines JSON format data @ >>>>>nmea_eng_room.json
.....
Writing NMEA RAW data in 2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:37.561942----/dev/ttyACM0----SERMED,1,82.53,82.53,82.53,83.02,82.53,82.53,79.61,*11
current_time 2021-12-13 21:11:37.561942
Writing Main Engines NMEA RAW data in Engine_Room_2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:37.561942----/dev/ttyACM0----SERMED,1,82.53,82.53,82.53,83.02,82.53,82.53,79.61,*11
Writing Main Engines JSON format data @ >>>>>nmea_eng_room.json
.....
Reload UART Ports
Writing NMEA RAW data in 2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:37.936784----/dev/ttyACM1----$GPGSA,A,3,01,02,03,04,05,06,07,08,09,10,11,12,1.0,1.0,1.0*30
current_time 2021-12-13 21:11:37.936784
/dev/ttyACM1 $GPGSA,A,3,01,02,03,04,05,06,07,08,09,10,11,12,1.0,1.0,1.0*30
.....
Writing NMEA RAW data in 2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:40.139341----/dev/ttyACM1----$GPRMC,192122.645,A,3532.036,N,02402.227,E,038.9,030.9,100921.000,0.0,W*74
current_time 2021-12-13 21:11:40.139341
Writing GPS-NMEA RAW data in GPS_2021_12_13_21_11_00_nmea_msg.txlog>>>>2021-12-13 21:11:40.139341----/dev/ttyACM1----$GPRMC,192122.645,A,3532.036,N,02402.227,E,038.9,030.9,100921.000,0.0,W*74
Writing GPS JSON format data @ >>>>>nmea_gps.json

```

FIGURE 5.5: NMEA Gateway Server

5.2 Case Study 2: NMEA Gateway Server Application.

In the Second Case Study, i have make an Linux terminal application (GateWay Server) for handling the incoming NMEA data sentences, from the vessel local network. First must be identified and setting the ports of incoming messages,so that they can be read, clean and after will be log and archive of them. Follows the analysis of the NMEA sentences, for extraction and organization of the necessary information. And finally to be converted into a more user-friendly format and stored in the corresponding. And finally the conversion of data into a more human readable format and stored in the corresponding formation.

5.2.1 Identify and Reading USB ports.

Identifying and reading ports for incoming NMEA messages is done using a 3-part Python3 code library-module (library.ports). ¹⁸⁸ My code with use this library is provided at 5.6.

First the ports are identified¹⁸⁹, then adjusted for read / write speed 4800 baud.

And finally, the data **value** is read, for each port in a loop.

```

1 import time,serial,os
2 from library.ports import serial_ports,def_ports,read
3
4
5 def connect_ports():
6     time.sleep(0.1)
7     ports=serial_ports()
8     #print("Connection Ports", ports)
9     return def_ports(ports)
10
11
12 def data_read():
13     Ports=connect_ports()
14     #print(Ports)
15     for porta in Ports:
16
17         try:
18             value=read(porta)
19
20             archive_nmea(value)
21             parse_data(value)
22
23         except:
24             Ports=connect_ports()
25         pass

```

LISTING 5.6: Identify and Reading USB/Serial NMEA Ports.

5.2.2 Logging incoming data.

The recording /logging of all incoming NMEA data is done with the following function who provided at 5.7.

Needing the incoming NMEA **value** 5.2.1 from each readable port¹⁹⁰, **port name** and a **timestamp**.

¹⁸⁸The library-module, has been granted for use in current research work, with the restriction Not to Publish the code it contains.

¹⁸⁹Example port list for linux like OS : ['/dev/ttyUSB0','/dev/ttyACM1','/dev/ttyACM0']

¹⁹⁰If the data can decoding in utf-8 symbol format.

```

1 import time
2
3 current_time = now.strftime("%Y_%m_%d_%H_%M_%S") #generate a string from
   current time eg. 2021_12_13_22_22_55.150511
4 logfilename=str(current_time)+"_"+nmea_msg.txlog #generate a part for final
   filename eg. 2021_12_13_22_22_55.150511_nmea_msg.txlog
5
6
7 def archive_nmea(value, current_time, logfilename):
8
9     if value:
10        current_time = datetime.now()
11
12        try:
13            v=value.decode("utf-8")
14
15            with open("General"+logfilename, 'a+') as data:
16                d=str(current_time)+"----"+str(porta.port)+"----"+v
17                #generate final filename eg. General2021_12_13_22_22_55.150511
   _nmea_msg.txlog
18                data.write(d)
19
20        except:
21            continue
22
23    else: return

```

LISTING 5.7: Logging incoming NMEA data.

5.2.3 Parse NMEA messages.

For parsing of NMEA sentences, I have made a function 5.8 who takes the incoming **value**, analyzes and returns the NMEA-message type (**msg_type**) and the message-data (**msg**) as a list¹⁹¹ of strings.

```

1 def parse_nmea_basic(value):
2     if value:
3         try:
4             v=value.decode("utf-8")
5             try:
6                 v0=v.split("$", 2)[1].split(",*")[0]
7                 msg_type=v0.split(",")[0]
8                 msg_constructor=msg_type[0:2]
9                 msg_spec=msg_type[2:5]
10                msg=v0.split(",")[1:-1]
11                CheckSum=v.split(",*")[1]
12                msg=v.split(",")
13                return msg_type, msg
14            except:
15                return "", ""
16
17        except:
18            return
19
20
21

```

LISTING 5.8: Parsing NMEA sentences.

¹⁹¹Example : msg_type>"ERMED", msg>["0","80.02","80.02","80.02","80.51","80.51","80.51","80.26"]

5.2.4 Transform NMEA messages to JSON format.

The NMEA data protocol is clearly defined for all its sentences. It has multiple options for sending data from the various sentences. This means an information (partial data), eg UTC time (hh-mm-ss.sss), can be included within more than one different type of sentences. They are not human readable, due to their formatting and for the rapid sequence of new sentences from each NMEA broadcasting device.

The last reason drove me to make the 5.9 python method, combining relative NMEA sentences in the JSON format files 5.10 and 5.11.¹⁹²

```

1 def nmea2json(nmea_parse_basic(value), porta.port):
2     if msg_type=="GPGGA":
3         d_gps.update({"UTC time":round(float(msg[1]),2)})
4         d_gps.update({"Lat":[round(float(msg[2]),2),msg[3]]})
5         d_gps.update({"Long": [round(float(msg[4]),2), msg[5]]})
6         d_gps.update({"Height":[round(float(msg[9]),2),msg[10]]})
7         d_gps.update({"GPS Quality indicator":int(msg[6])})
8         d_gps.update({"Number of Satellite's": int(msg[7])})
9
10        with open("nmea_gps.json", "w") as f:
11            dic={"GPS":[{"porta.port:msg_type},{"System Time": str(current_time)},
12                d_gps]}
13            json.dump(dic, f)
14
15        if int(msg[1])==0:
16            d_er0.update({"T#1":float(msg[2])})
17            d_er0.update({"T#2":float(msg[3])})
18            d_er0.update({"T#3":float(msg[4])})
19            d_er0.update({"T#4":float(msg[5])})
20            d_er0.update({"T#5":float(msg[6])})
21            d_er0.update({"T#6":float(msg[7])})
22
23            with open("nmea_eng_room.json", "w") as f:
24                dic={"Engine Room":[{"porta.port:msg_type},{"System Time": str(
25                    current_time)},
26                    {"Left Main Engine ": [0, {"Temperatures": d_er0}]},
27                    {"Rigth Main Engine ": [1, {"Temperatures": d_er1}]}
28                ]}
29                json.dump(dic, f)

```

LISTING 5.9: Combining NMEA in JSON Format.Sample code.

```

1 {
2   "GPS":
3     [
4       {"/dev/ttyACM0": "GPGGA"},
5       {"System Time": "2021-12-06 06:28:10.930286"},
6       {
7         "UTC time": 191929.64,
8         "Lat": [3531.45, "N"],
9         "Long": [2401.02, "E"],
10        "Height": [0.0, "M"],
11        "Speed over ground": [38.9, "knots"],

```

¹⁹²Those files used, as data sources by "IoMaT Dashboard" web-based application.5.3

```

12     "Track (true)": [37.7, "degrees"],
13     "Date": [100921, "dd/mm/yy"],
14     "Magnetic variation": [0.0, "degrees"],
15     "Mode 1": "A",
16     "Mode 2 Fix Type": 1,
17     "PDOP": [1.0, "m"],
18     "HDOP": [1.0, "m"],
19     "VDOP": [1.4, "m"],
20     "GPS Quality indicator": 1,
21     "Number of Satellite's": 12
22   }
23 ]
24 }

```

LISTING 5.10: NMEA-GPS Data in JSON Format.

```

1 {
2   "Engine Room":
3     [
4       {"/dev/ttyACM0": "ERMED"},
5       {"System Time": "2021-11-29 06:42:01.381919"},
6       {"Left Main Engine ":
7         [0, {"Temperatures":
8           {
9             "T#1": 81.97,
10            "T#2": 82.46,
11            "T#3": 81.97,
12            "T#4": 81.97,
13            "T#5": 81.97,
14            "T#6": 82.46
15          }
16        ]
17      },
18      {"Rigth Main Engine ":
19        [1, {"Temperatures":
20          {
21            "T#1": 84.97,
22            "T#2": 85.46,
23            "T#3": 84.97,
24            "T#4": 84.97,
25            "T#5": 84.97,
26            "T#6": 85.46
27          }
28        ]
29      }
30    ]
31  }
32 ]
33 }

```

LISTING 5.11: NMEA-ERMED Data in JSON Format.

5.3 Case Study 3: Web-based IoMaT Dashboard.

The Visualization of generated data from an Internet of Maritime Things (IoMaT) system for Vessel Performance Monitoring. It was performed using web-based technologies eg. HTML & JavaScript and free programming tools e.g. Python3, Flask and Jinja2.

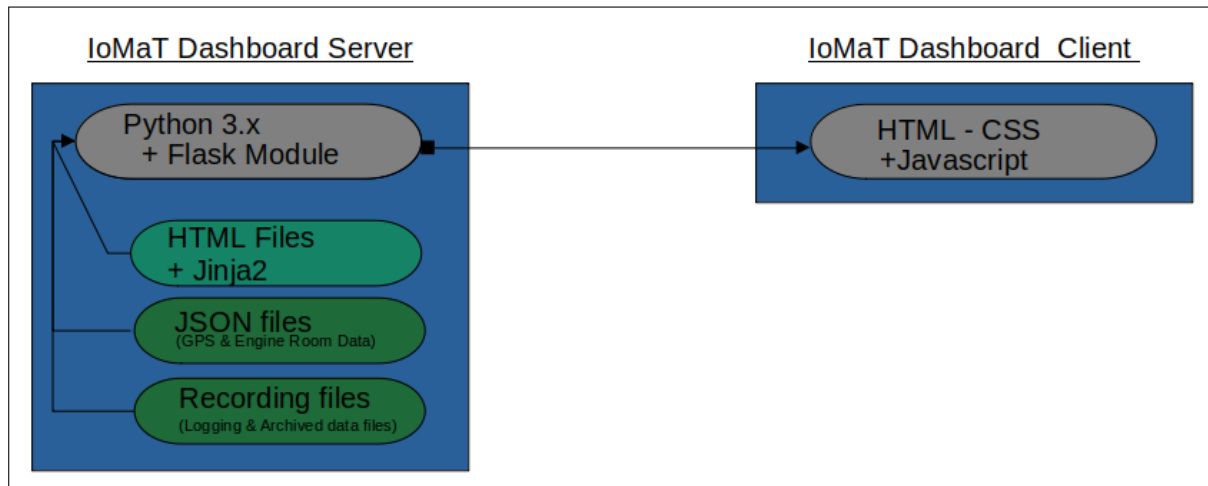


FIGURE 5.6: Web-based Dashboard overview diagram.

This is the base diagram (idea) which i used for build of the Web-Based Dashboard Application.

5.3.1 Python with Flask Web-server.

With the above diagram 5.6 and the following basic code 5.12 . Where in the python 3.x script i import the flask module and create a page with "/hello" routing, for our main message 'Hello World from Python Flask!'.

```

1 #####importing modules & functions#####
2
3 from flask import Flask
4
5 ##### Initialize the Application Server#####
6
7 app = Flask(__name__) # Initialize Web Server
8
9 @app.route('/hello')
10 def hello_world():
11     return 'Hello World from Python Flask!'
12
13
14 if __name__ == '__main__':
15     app.run(debug=True)
16

```

LISTING 5.12: Basic Python3 with Flask module sample web application.

5.3.2 User Authentication Screen.

In a general information security context: Verifying the identity of a user often as a prerequisite to allowing access to resources in an information system¹⁹³. This is achieved by a username and password¹⁹⁴, however several other types authentication bio-metrics^{195 196}, are used.

In the case of User Authentication for "IoMaT Dashboard", my choice is the use of simplest method of " Password-based authentication" 5.13 & 5.14. Passwords are the most common methods of authentication. Passwords can be in the form of a string of letters, numbers, or special characters.

```

1 #Python3 and flask code for user authentication
2 #Application Users:
3 users = []
4 users.append([0, 'AgRitas', 'AgRitas12345!&$-.'])
5 users.append([1, 'John', 'code_of_john'])
6 app.secret_key = "Secret Encryption Key" # Setting the Encrypt-Decrypt Key
7
8 ###Setting user global variable (cookie)
9 @app.before_request
10 def before_request():
11     g.user = None
12     if 'user_id' in session:
13         for user in users:
14             if user[0] == session['user_id']:
15                 g.user = user
16                 g.s = s
17
18 ### Defining "logme" dashboard panel and bind the relative function.
19 @app.route('/logme', methods=['POST', 'GET'])
20 def logme():
21     # users=get_Users()
22     session.pop('user_id', None)
23     if request.method == 'POST':
24         username = request.form['username']
25         password = request.form['password']
26         for user in users:
27             if user[1] == username and user[2] == password:
28                 session['user_id'] = user[0]
29                 return redirect(url_for('nav_data'))
30     return render_template('login.html')

```

LISTING 5.13: User authentication page.Python -Flask code.

```

1 <div class="container">
2 <br/><br>
3 <div class="login"> Login to Access the IoMaT Dashboard.</div>
4 <br/>
5 <form method="POST">
6 <input id="user" type="text" name="username" placeholder="User ID"><br />
7 <input id="pass" type="password" name="password" placeholder="User
  Password"><br /> <br>
8 <button type="submit">Login </button> <br/> <br>
9 </form>
10 </div>

```

LISTING 5.14: User authentication page.Html code.

¹⁹³<https://csrc.nist.gov/glossary/term/authentication>

¹⁹⁴<https://www.idrnd.ai/5-authentication-methods-that-can-prevent-the-next-breach/>

¹⁹⁵<https://en.wikipedia.org/wiki/Biometrics>

¹⁹⁶<https://www.biometricsinstitute.org/what-is-biometrics/types-of-biometrics/>

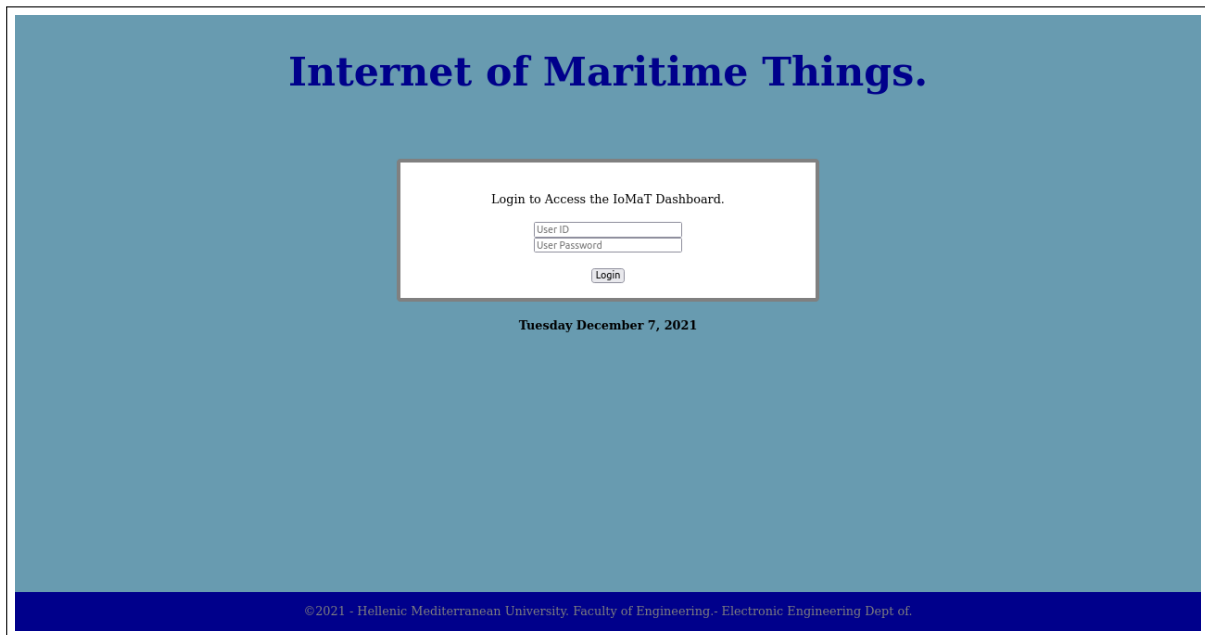


FIGURE 5.7: User authentication page.

5.3.3 Navigation Bridge Panel.

The most usable data and information's on the vessel are the visualized navigation data e.g. positioning data, speed, course and basic meteorological data e.g. wind direction & speed and air temperature. The source codes of Routing & the Panel for Navigation Bridge Information page, and a screen-shot are in 5.15, 5.16 & 5.8. I use recording data from "nmea_gps.json" file 5.10.

```

1  ### From individual modules & libraries
2  from getNMEA import nmea_gps, trans_degrees
3
4  ### Extract Vessel Position, from json file "nmea_gps.json"
5  def pos():
6      nmea_gps_data = nmea_gps()
7      return [float(nmea_gps_data[2][0]), float(nmea_gps_data[3][0])]
8
9  ### Defining "logme" dashboard panel , using the route () decorator to bind a
   function
10 @app.route('/nav_bridge', methods=['POST', 'GET'])
11 def nav_data():
12     if not g.user:
13         return redirect(url_for('logme'))
14     nmea_data = nmea_gps()
15     dat = pos()
16     pos2 = trans_degrees(dat[1])
17     pos1 = trans_degrees(dat[0])
18     w = wind(windmain)
19     return render_template('navigation.html', data=nmea_data, pos=dat, lat=pos1
   , long=pos2)

```

LISTING 5.15: Navigation Bridge Information Panel.Python -Flask code.

```

1 <div class="containerbridge_info ">
2
3 <p class="nav_gps_compass left_box">{{'%.1f'|format(data[0])}} (True Track)
  </p>
4 <p class="nav_nav_text nav_left" style=" top:700px;"> Course Over Ground
  (GPS) </p>
5
6 <p class="nav_gps_log left_box">{{long[0]}}{{long[1]}} {{long[2]}}{{long
  [3]}} {{long[4]}}{{long[5]}} {{data[3][1]}} </p>
7 <p class="nav_nav_text nav_left" style="top:240px;">GPS-Longitude </p>
8
9
10 <p class="nav_gps_lat left_box">{{lat[0]}}{{lat[1]}} {{lat[2]}}{{lat[3]}}
  {{lat[4]}}{{lat[5]}} {{data[2][1]}} </p>
11 <p class="nav_nav_text nav_left" style="top: 360px;">GPS-Latitude </p>
12
13 <p class="nav_gps_speed left_box" title="(Speed Over Ground) ">{{'%.1f'|
  format(data[1])}} Knots </p>
14 <p class="nav_nav_text nav_left" style="top: 480px;" title="(Speed Over
  Ground) ">Speed Over Ground (GPS) </p>
15
16
17 </div>

```

LISTING 5.16: Navigation Bridge Information Panel.Html sample code.

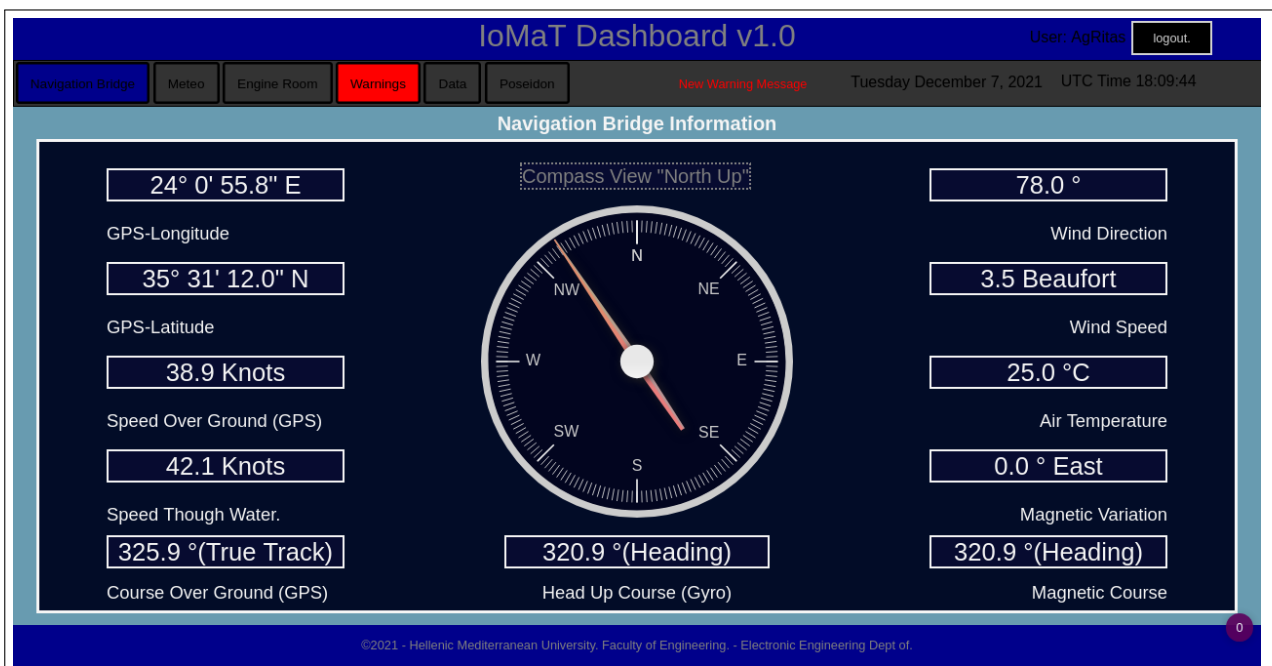


FIGURE 5.8: Navigation Bridge Information Panel.

5.3.4 Vessel Meteo Station Panel.

The measurements from meteo-station of the vessel, together with the meteorological forecasts from the international and national services. For navigation at sea in general. Playing very important role in the Captain's decisions for the safety of ship navigation, goods transportation and people life's safety during the voyage.

In this section, I use generated data measurements from Sensor Hat Emulator and pure Python3 code 5.1.1. The meteo panel 5.9, are building using the following source code's for Routing and visual instruments (gauges) 5.17 & 5.18.

```

1  ## Defining meteo dashboard panel, using the route() decorator to bind a
   function.
2  @app.route('/meteo', methods=['POST', 'GET'])
3  def sense_data():
4      s = ws, a = av, sp = ws[-1]
5      w = wind(windmain)
6      s = wind_s(sp, s, a)
7      return render_template('meteo.html', wind=w, wind_s=float(s[0][-1]))

```

LISTING 5.17: Meteo Station Panel.Python -Flask code.

```

1  <div style= " position: fixed; top:260px;left:355px;">
2      <canvas data-type="radial-gauge"
3          data-width="280"
4          data-height="280"
5          data-units="Beaufort"
6          data-title="false"
7          data-value={{ '%0.1f' | format(wind_s) }}
8          data-animate-on-init="false"
9          data-animated-value="true"
10         data-min-value="0"
11         data-max-value="13"
12         data-major-ticks="0,1,2,3,4,5,6,7,8,9,10,11,12,13"
13
14         data-stroke-ticks="false"
15         data-highlights=' [
16             { "from": 0, "to": 3, "color": "blue" },
17             { "from": 3, "to": 6, "color": "green" },
18             { "from": 6, "to": 9, "color": "orange" },
19             { "from": 9, "to": 12, "color": "red" },
20             { "from": 12, "to": 13, "color": "darkred" }
21         ]'
22         data-color-plate="rgb(55, 67, 34)"
23         data-color-major-ticks="#f5f5f5"
24         data-color-minor-ticks="#ddd"
25         data-color-title="#fff"
26         data-color-units="#ccc"
27         data-color-numbers="#eee"
28         data-color-needle-start="rgba(240, 128, 128, 1)"
29         data-color-needle-end="rgba(255, 160, 122, .9)"
30         data-value-box="false"
31         data-animation-rule="bounce"
32         data-animation-duration="500"
33         data-border-outer-width="3"
34         data-border-middle-width="3"
35         data-border-inner-width="3"
36 ></canvas >

```

LISTING 5.18: Meteo Station Panel.Html sample code.

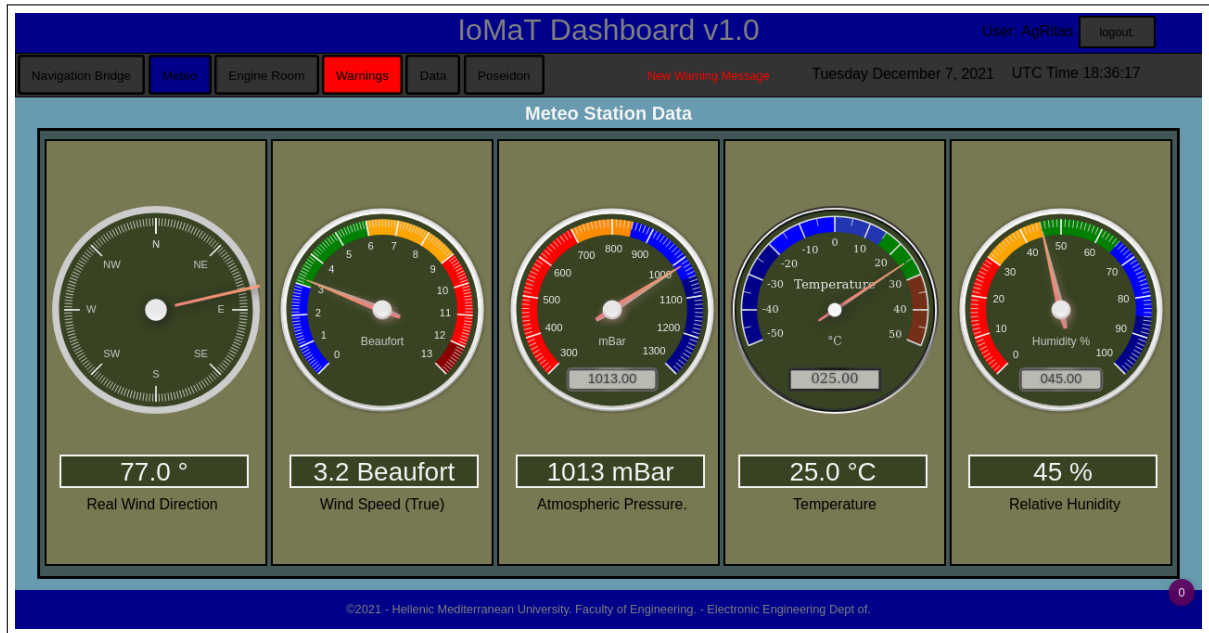


FIGURE 5.9: Meteo Station Panel.

5.3.5 Engines Room Control Panel.

The engine room is a vital area for the vessel operation. For example, the propulsion (main engine) and the electrical power supply (generators) of the ship are here.

In my case, I put in Engine Room Panel 5.10 temperatures and RPMs (revolutions per minutes) of 2 hypothetical Main Engines. Code listing in 5.19 & 5.20. I use also recorded data from "nmea_eng_room.json" file 5.11 .

```

1 ### From individual modules & libraries
2 from getNMEA import nmea_irmed
3
4 ##### Initialize Vessel Ecosystem Variables #####
5 _rpm = random.randint(150, 230) # Initialize random engines RPMs
6
7 # # Defining engine room panel route (), to bind a function
8 @app.route('/engine_room', methods=['POST', 'GET'])
9 def engine_data():
10     if not g.user:
11         return redirect(url_for('logme'))
12
13     nmea_data_irmed = nmea_irmed()
14
15     rpm = _rpm+random.randint(-3, 3)
16     rpm1 = _rpm+random.randint(-5, 8)
17     nmea_data_irmed.append([rpm, rpm1])
18     return render_template('engine_room.html', data=nmea_data_irmed)

```

LISTING 5.19: Engines Room Control Panel.Python -Flask code.

```

1 <div id="C_1" class="box_eng">
2   <div>
3     <div class="thermometer_eng" style='height: 200px;'>
4       <div class='draw-a'></div>
5       <div class='draw-b'></div>
6       <div class='meter'>
7         <div class='statistics'>
8           <div class='percent percent-a'>___100 </div>
9           <div class='percent percent-b'>___75 </div>
10          <div class='percent percent-c'>___50 </div>
11          <div class='percent percent-d'>___25 </div>
12          <div class='percent percent-e'>___0 </div>
13        </div>
14        <div class='mercury' style='height: {{(data[1])}}%'>
15          <div class='percent-current' style="font-family:Roboto, sans-serif;
16            font-size: 1px; background-color: red;">{{'%0.1f'|format(data[1])}}
17          C </div>
18          <div class='mask'>
19            <div class='bg-color'></div>
20          </div>
21        </div>
22      </div>
23      <br />
24      <p>C#1:{{'%0.1f'|format(data[1])}} C</p>
25    </div>
26  </div>

```

LISTING 5.20: Engines Room Control Panel.Html sample code.

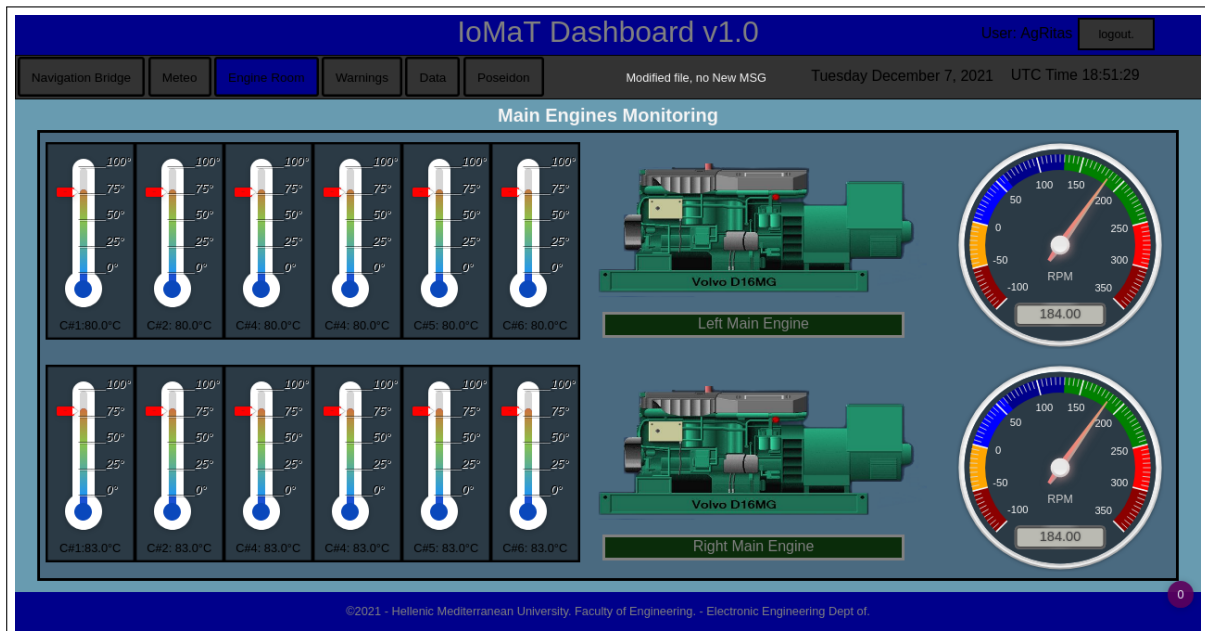


FIGURE 5.10: Engines Room Control Panel.

5.3.6 Warnings and Alerts, Panel.

The creation of warning or alarm messages from one or more points-areas of interest. Based on direct, indirect and self-controls eg "fire in a place", "impending bad weather", "bug or damage to an electromechanical, electronic or telecommunication system" etc, is very important. Most manufacturers have provided or built-in this feature in their machines.

The collection of all these messages in one place, facilitates the management and decision making by the Captain, the First Engineer and the managers of Shipping Company.

For this reason I created a panel 5.11, with which this information can be managed. The data was selected to be stored in JSON format 5.23. There are possibilities for warning in all panels, archiving, deleting, correcting (adding notes) 5.12 of the messages in it in the warning panel. The source codes of Routing & the Panel for Warnings are in 5.21 and 5.22.

```

1 ### From individual modules & libraries
2 from library.warn import *
3
4 @app.route('/warnings', methods=['GET', 'POST'])
5 def warnings():
6     if request.method == 'POST':
7         now = datetime.now()
8         warn_type = request.form.get('warn_type')
9         warn_msg = request.form.get('warn_msg')
10        warnDict = {
11            'id': int(datetime.timestamp(now)),
12            'type': warn_type,
13            'msg': warn_msg,
14            'status': 0,
15            'create_at': now.strftime("%Y/%m/%d_%H:%M:%S"),
16            'notes': None
17        }
18        addWarn(warnDict)
19
20
21    return render_template('warning.html', warns=getWarnings())
22
23
24 @app.route('/delete/<id>')
25 def delete(id):
26     deleteWarn(id)
27     return redirect(url_for('warnings'))
28
29
30 @app.route('/edit/<id>', methods=['GET', 'POST'])
31 def edit(id):
32     warn = getWarn(id)
33     if request.method == 'POST':
34         warn_note = request.form.get('warn_note')
35         warn['notes'] = warn_note
36         updateWarn(warn)
37         return redirect(url_for('warnings'))
38     dat = pos()
39     return render_template('warn_edit.html', warn=warn, pos=dat)

```

LISTING 5.21: Warnings and Alerts Panel.Python -Flask code.

```

1 <div id="C_1" class="box_eng">
2   <div>
3     <div class="thermometer_eng" style='height: 200px;'>
4       <div class='draw-a'></div>
5       <div class='draw-b'></div>
6       <div class='meter'>
7         <div class='statistics'>
8           <div class='percent percent-a'>___100 </div>
9           <div class='percent percent-b'>___75 </div>
10          <div class='percent percent-c'>___50 </div>
11          <div class='percent percent-d'>___25 </div>
12          <div class='percent percent-e'>___0 </div>
13        </div>
14        <div class='mercury' style='height: {{(data[1])}}%'>
15          <div class='percent-current' style="font-family:Roboto, sans-serif;
16            font-size: 1px; background-color: red;">{{'%0.1f'|format(data[1])}}
17        C </div>
18          <div class='mask'>
19            <div class='bg-color'></div>
20          </div>
21        </div>
22      </div>
23    <br />
24    <p>C#1:{{'%0.1f'|format(data[1])}} C</p>
25  </div>
26 </div>

```

LISTING 5.22: Warnings and Alerts Panel.Html sample code.

The screenshot displays the 'IoMaT Dashboard v1.0' interface. At the top, there is a navigation bar with tabs for 'Navigation Bridge', 'Meteo', 'Engine Room', 'Warnings' (which is active and highlighted in red), 'Data', and 'Poselidon'. To the right of the navigation bar, it shows 'User: AgRitas', a 'logout' button, and the date 'Tuesday December 7, 2021' along with 'UTC Time 21:28:37'. Below the navigation bar, the main content area is titled 'Monitoring System Warning's' and includes an 'Add Warn-Alert' button. The main content is divided into two columns:

- New Warning's & Alarm's Message's:** This column contains a list of recent alerts, each with a timestamp, a description, and a checkbox. The alerts include:
 - 2021/12/07 23:28:22 --Alarm Add Warn-Alert Message
 - 2021/12/07 23:27:25 --Alarm Add Warn-Alert Message
 - 2021/12/07 20:09:35 --test Add Warn-Alert Message
 - 2021/11/11 20:28:36 --test2 Add Warn-Alert Message
 - 2021/11/11 20:18:57 --Alert Alert Message >>> Fire on Engine Room
 - 2021/10/09 16:31:10 --Alarm Add Warn-Alert Message
- Viewed-Verified Warning's:** This column shows a list of alerts that have been viewed and verified. Each entry includes a timestamp, a description, and status icons (a green checkmark, a red X, and a trash icon). The alerts include:
 - 2021/09/30 23:31:10 -- Add Warn-Alert Message
 - 2021-09-27 18:27:10 --Alert_Fire
 - 2021-09-26 15:27:17 --Alert_Fire on Cusine
 - 2021-09-26 15:13:42 --Warning_Main Engine
 - 2021-09-26 15:12:37 --Warning_Main Engine
 - 2021-09-26 15:11:24 --Warning_Main Engine

At the bottom of the dashboard, there is a footer with the text: '©2021 - Hellenic Mediterranean University, Faculty of Engineering - Electronic Engineering Dept of.' and a small red circle with the number '0' in the bottom right corner.

FIGURE 5.11: Warnings and Alerts Panel.

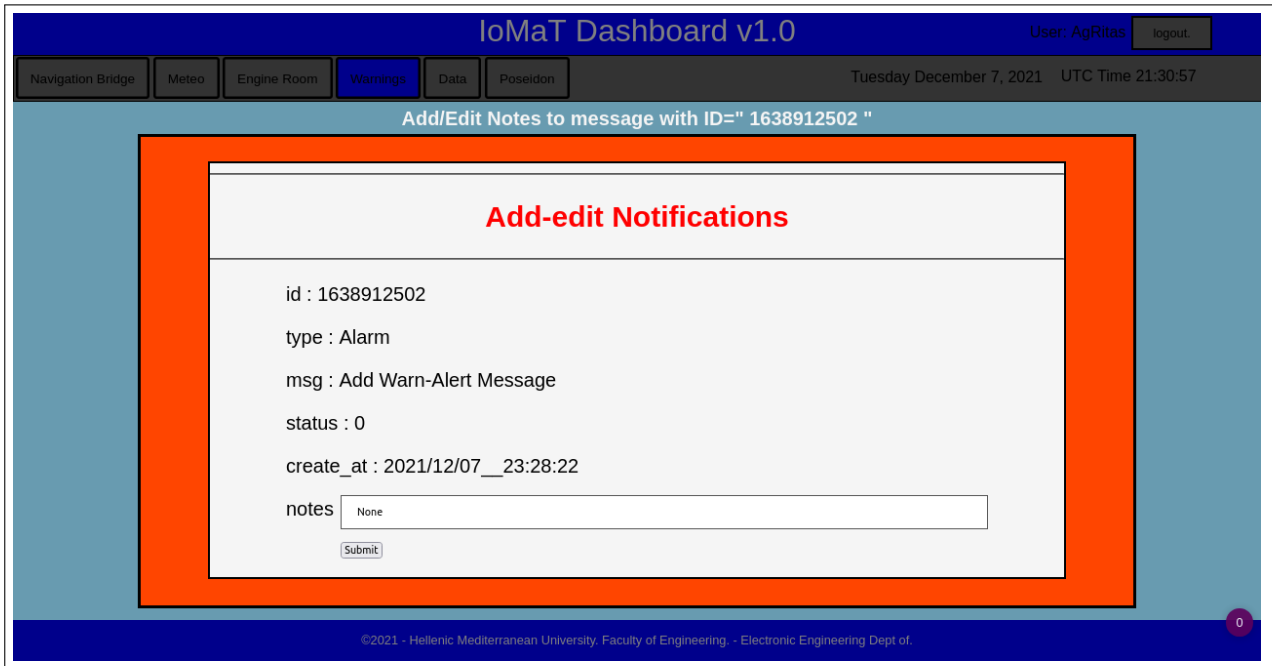


FIGURE 5.12: Edit Warnings and Alerts Panel.

```

1  [
2  {
3    "id": 1632658284,
4    "type": "Warning/Alert",
5    "msg": "What happen??",
6    "status": 0,
7    "create_at": "2021-09-26 15:11:24",
8    "notes": null
9  },
10 {
11  "id": 1632658357,
12  "type": "Warning",
13  "msg": "Main Engine 1",
14  "status": 1,
15  "create_at": "2021-09-26 15:12:37",
16  "notes": null
17 },
18 {
19  "id": 1632659237,
20  "type": "Alert",
21  "msg": "Fire on Cuisine",
22  "status": 1,
23  "create_at": "2021-09-26 15:27:17",
24  "notes": "The fire was extinguished in 5 minutes. "
25 },
26 {
27  "id": 1636654737,
28  "type": "Alert",

```



```

29     "msg": "Fire on Engine Room",
30     "status": 0,
31     "create_at": "2021/11/11__20:18:57",
32     "notes": "The fire was extinguished in 2 minutes."
33 },
34 {
35     "id": 1636655316,
36     "type": "test2",
37     "msg": "Add Warn-Alert Message"           ",
38     "status": 0,
39     "create_at": "2021/11/11__20:28:36",
40     "notes": null
41 },
42 ]

```

LISTING 5.23: Warnings and Alerts Json file.

5.3.7 Archiving Data Files,Access Panel.

Old logs, when the NMEA Gateway Server 5.2 application starts, are archived 5.24, and new logs 5.2.2 are created.

All files can be viewed at the "Data Archiving Panel" 5.13 , both for past and current files. The current files are viewed 5.15 in real time state¹⁹⁷. The past files also, can be fetched and saved from authorized the user, for further data analysis 5.14.

```

1 # Code from "NMEA Gateway Server"
2 #Importing of the necessary method
3 from checker import log_move
4
5 #Apply when need with call of..
6 log_move()

```

LISTING 5.24: Archiving Past Data Files.Python code.

```

1 #####importing modules & functions#####
2 import os
3 ##### individual modules & libraries #####
4 from checker import past_log_files
5
6 #####
7 @app.route('/data')
8 def data():
9     if not g.user:
10         return redirect(url_for('logme'))
11     paths = ["/static/log/", "."]
12     files = past_log_files(paths[0])
13     files2 = past_log_files(paths[1])
14     return render_template('data.html', paths=paths, files=files, files2=files2)
15
16 @app.route('/view/<id>')
17 def view(id):
18     path0 = os.getcwd()
19     filen = path0+'/static/log/'+id

```

¹⁹⁷When reloaded the panel

```

20 with open(filename) as f:
21     lines = f.readlines()
22     return render_template('view.html', lines=lines, file_id=id)
23
24 @app.route('/current/<id>')
25 def current(id):
26     path0 = os.getcwd()
27     filename = path0+'/' +id
28     with open(filename) as f:
29         lines = f.readlines()
30     return render_template('view.html', lines=lines, file_id=id)

```

LISTING 5.25: Logging Data Files Panel.Python -Flask code.

```

1 <div id="Past" class="tabcontent">
2 <div class="containerinfo">
3 <h3>Past Archiving File</h3>
4 <hr><hr> {% for idx in files %}
5 <p>{{idx}}
6 <a href="{{ url_for('static', filename='log/' +idx ) }}">
7  </a>
9 <a href="{{ url_for('view', id=idx ) }}">
10  </a>
12 </p>
13 <hr>
14 {% endfor %}
15 <hr> <h3> End of the file's List</h3>
16 </div>
17 </div>

```

LISTING 5.26: Logging Data Files Panel.Html sample code.

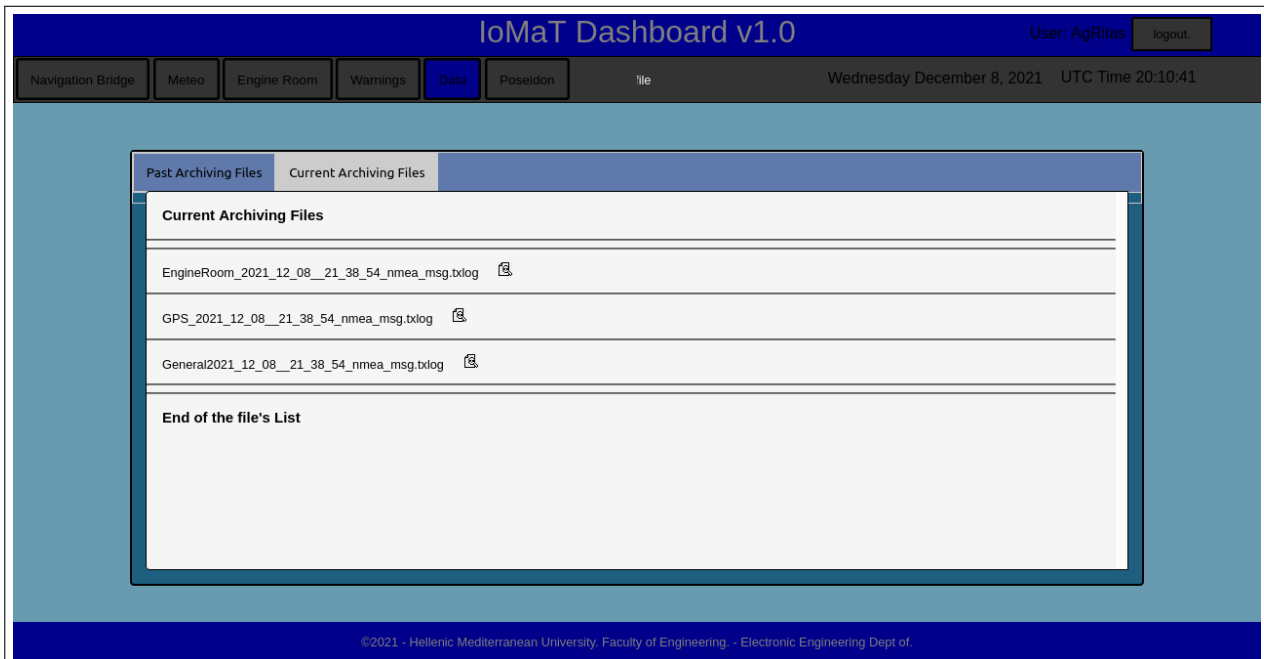


FIGURE 5.13: Current Data Files,Access Panel.

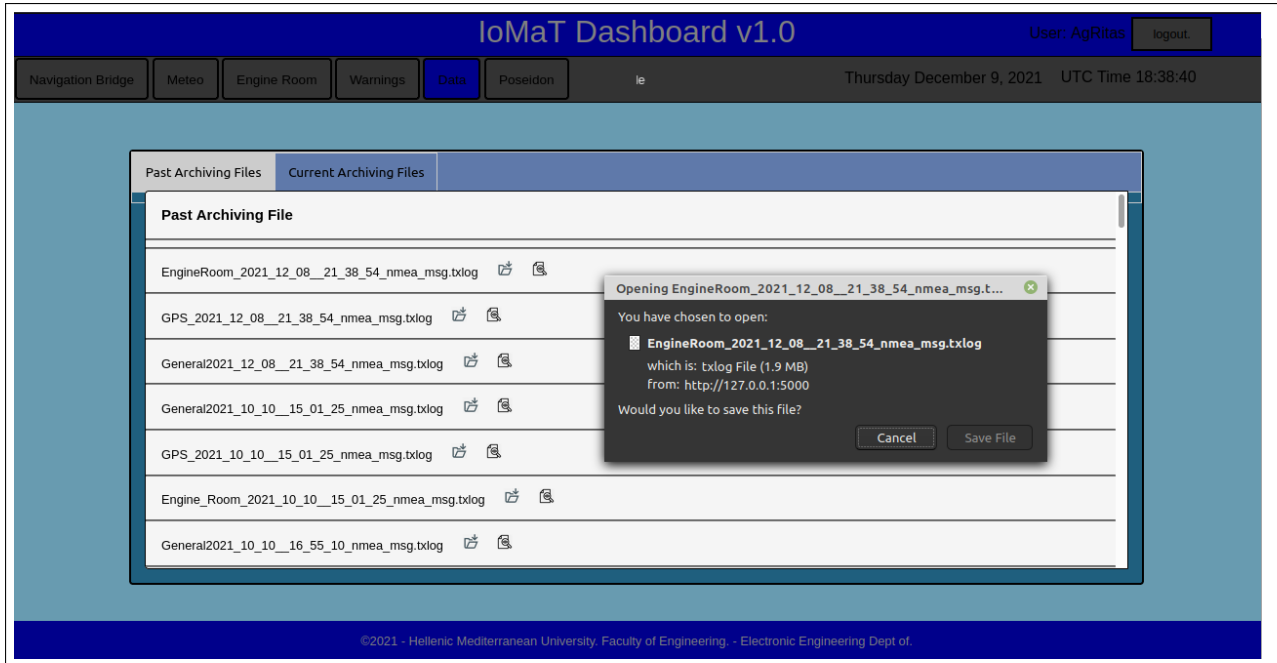


FIGURE 5.14: Fetched Archived Data Files,Access Panel.

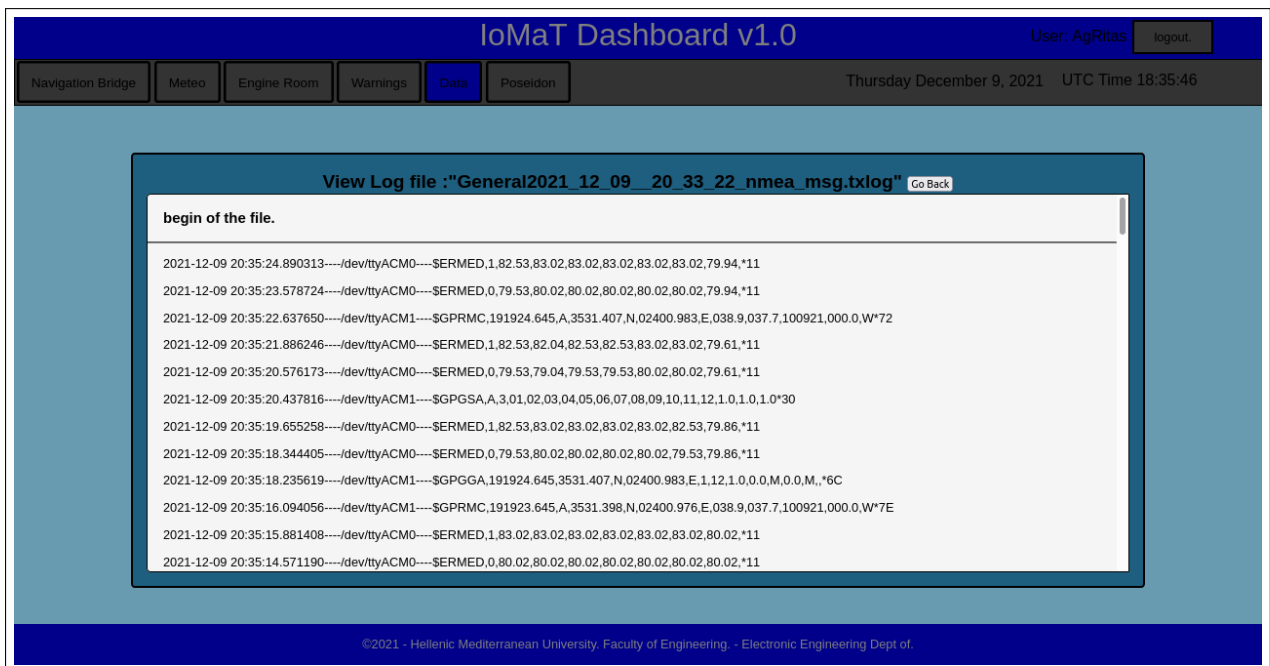


FIGURE 5.15: Direct View of Current and Archived Data Files.

5.3.8 Optional, by web-access Poseidon Meteo forecast System.

As is well known and I have mentioned above, the meteorological forecasts for shipping from international and national organizations is crucial for maritime safety.

Regarding for Greece and the wider region of the Eastern Mediterranean and the Black Sea¹⁹⁸. Two national research agencies, the Hellenic National Meteorological Service (HNMS/EMY)¹⁹⁹ and Hellenic Center for Marine Research(HCMR/ΕΛΚΕΘΕ)²⁰⁰ with the Poseidon System²⁰¹, publish forecasts for Eastern METAREA III²⁰².

The reference in this case study to Poseidon System are made, because except the data from EMY, WMO and the other collaborating networks. Handle data also from it's own advanced technological meteorological stations as a special network of the smart sensors (meteo-IoT)²⁰³. And the predictions for meteorological and environmental parameters, is visualized with advanced animating graphs in special ECDIS system. Focusing in human understandable images, mainly for maritime-naval usage 5.16, 5.17, 5.18, 5.19. But for this characteristics and very accurate prognosis, also people use Poseidon System for terrestrial and environmental activities. Has only one disadvantage, the limitation of 4G/LTE-5G connectivity²⁰⁴ in the open sea and oceans. For this reason is not able the uninterrupted use of this service in vessels.

In my application the code need to get from satellite navigation data the coordination's 5.27, for focusing the chart of Poseidon System 5.28.

```

1
2 ##### individual modules & libraries #####
3 from getNMEA import nmea_gps
4
5 ##### Vessel Position Function #####
6 def pos():
7     nmea_gps_data = nmea_gps()
8     return [float(nmea_gps_data[2][0]), float(nmea_gps_data[3][0])]
9
10
11
12 @app.route('/someroute', methods=['POST', 'GET'])
13 def someroute():
14     dat = pos()
15     return render_template('someroute_page.html', pos=dat)
16

```

LISTING 5.27: Calculating Position Coordinates for Focusing Poseidon App.
Python -Flask code.

```

1 <a title="Poseidon Meteo"
2   href="https://poseidon.hcmr.gr/map?model=METE0&coords={{ '%0.5f'|format(pos
   [0]/100)}} ,{{ '%0.5f'|format(pos[1]/100)}} ,9"
3   id="engine_room" target="_blank">Poseidon</a>

```

LISTING 5.28: Redirecting in "blank tab" Poseidon Meteo Page.Html sample code.

¹⁹⁸Sub-areas including :Adriatic, Ionian, Libyan, Cretan, Carpathian, Myrtoo, Aegean, Thracian, Marmara, etc.

¹⁹⁹http://www.emy.gr/emyl/navigation/guide_bulletins

²⁰⁰<https://www.hcmr.gr/en/>

²⁰¹<https://poseidon.hcmr.gr/>

²⁰²<https://community.wmo.int/global-maritime-distress-and-safety-system-services-gmdss/metarea-3>

²⁰³<https://poseidon.hcmr.gr/services/ocean-data/situ-data>

²⁰⁴This means limitation of internet connectivity.

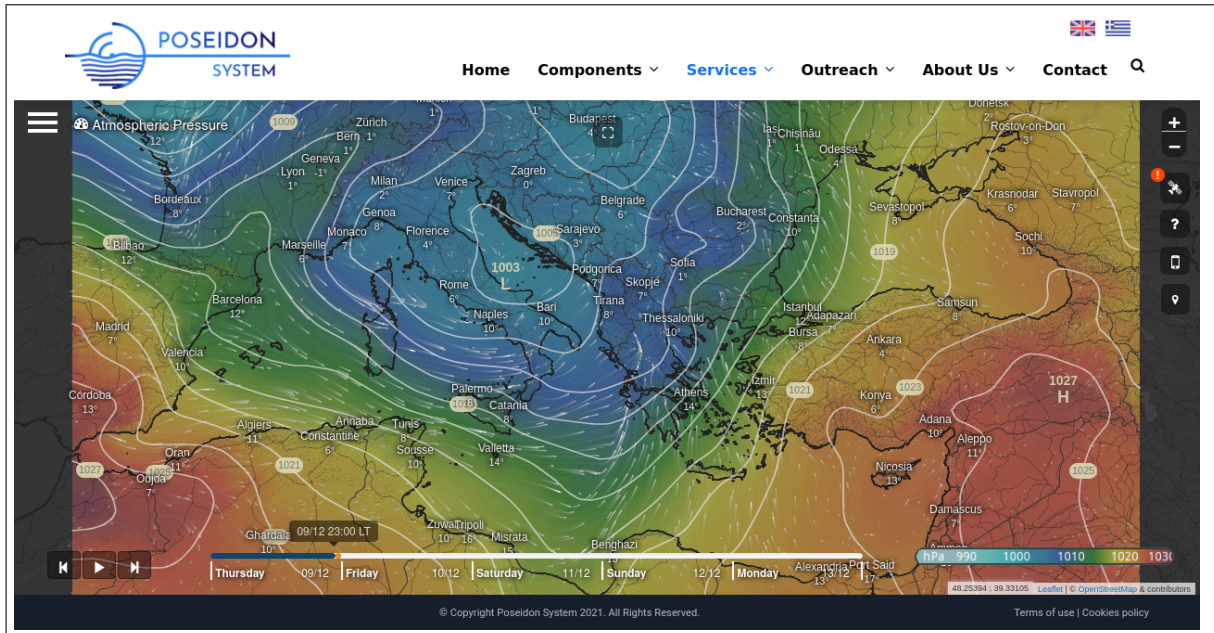


FIGURE 5.16: Poseidon Meteo Page, Atmospheric Pressure Forecast.

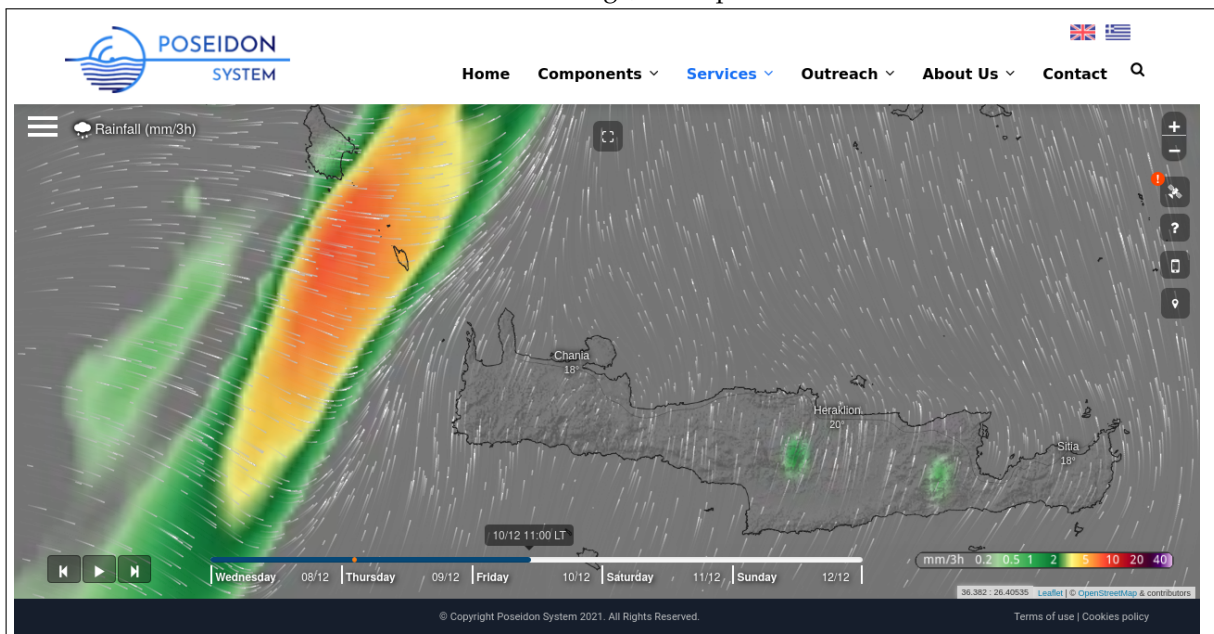


FIGURE 5.17: Poseidon Meteo Page, Rainfall Forecast.

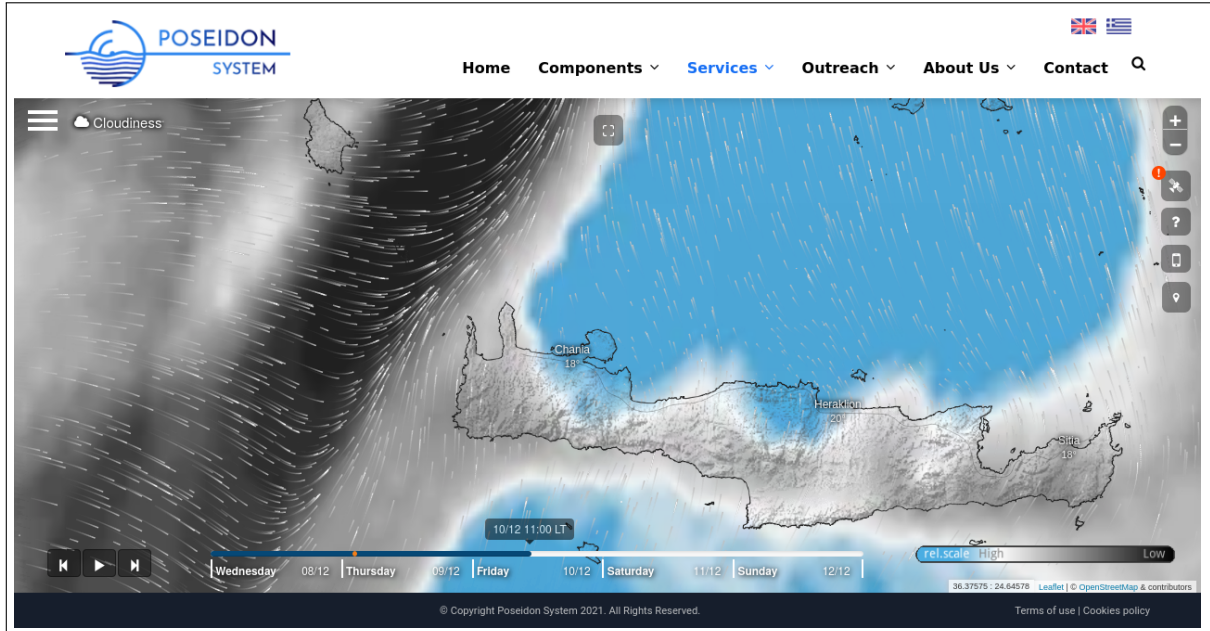


FIGURE 5.18: Poseidon Meteo Page, Cloudiness Forecast.

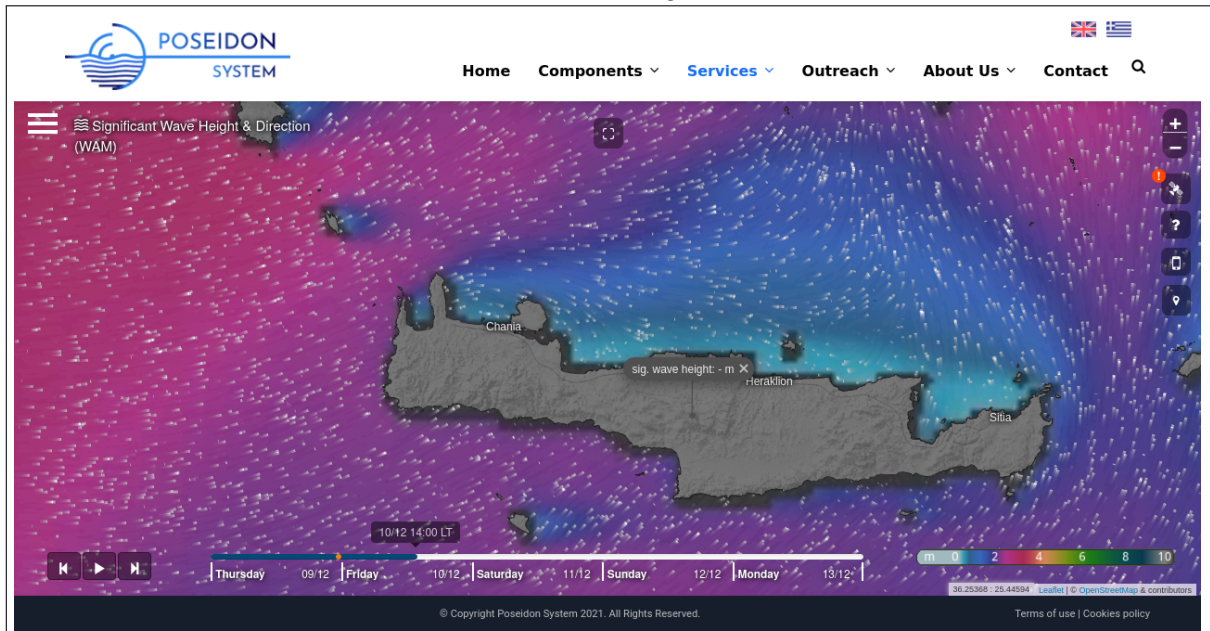


FIGURE 5.19: Poseidon Meteo Page, Wave Height and Direction Forecast.

Chapter 6

Conclusions

This is the last chapter and entitled "Conclusions", presents a brief evaluation of the contribution of this Master Thesis and lists the points that may be the subject of future research activity.

6.1 Sort review, of this Master Thesis.

In this master's thesis I studied the needs for using the Philosophy of Edge Computing Technologies (Chapter 3) for Internet of Things systems on Merchant Ships. These needs were created during the implementation of the requirements of Industry 4.0 in Merchant Shipping. What we now call Digital Shipping, Shipping 4.0 or Maritime 4.0.

Management of a Ship in transportation of cargoes and people, requires the utilization of information's produced inside and outside of the ship ecosystem.

This management is characterized as Data Driven and evaluates a set of Performance Indicators in the top level (Chapter 2). Needs an in-ship information system capable of collecting, storing, processing and visualizing (Chapter 4) a huge amount of data that is constantly being generated.

In the final I built up a software application (Chapter 5) to understand the operation and use of the various subsets of technologies and protocols. This application has 3 sections. 1st section, Generating IoMaT & NMEA data . 2nd section, Managing , Storage, Cleaning and Data Conversion into Human-readable format and the end the 3rd section, includes a Web-based IoMaT Dashboard.

6.2 Is real necessary the Edge Computing Technologies??

Yes, it is necessary to use Edge Computing Technologies in order to implement the specifications of Industry 4.0. Internet connectivity and the utilization of the technological advantages of cloud computing are not always possible. Specially when the ships are faraway from the shores and is out of the range of the 4G / LTE / 5G networks. Also, the cost of satellite connection is very high, for real-time applications.

6.3 Usage Advantages of Edge Computing Philosophy within IoMaT, for Vessel Performance Monitoring Case.

Traditional cloud computing has serious disadvantages including data security threats, performance issues, and growing operational costs. Because most data saved in the cloud has little significance and is rarely used, it becomes a waste of resources and storage space.

In many instances, it would be incredibly beneficial handling data on the device where it's generated. That's where edge computing comes in helping the decentralization of data processing and reduce the dependence from cloud computing.[149]

Edge computing has several advantages, such as:

- Increasing data security and privacy
- Better, more responsive and robust application performance
- Reducing operational costs
- Improving shipping-business efficiency and reliability
- Unlimited scalability
- Conserving network and computing resources
- Reducing latency

6.4 Technological Challenges in the IoT and IoMaT Field's.

However, there are other technological challenges[58, 165] on which research is focusing to promote the development and diffusion of IoT in maritime, powered by edge computing philosophy. A number of aspects of the work in this dissertation that can benefit from additional work, including but not limited to:

- **Standardization:**

In order to ensure the development of the IoT, it is crucial to have open standards for the connectivity of systems, interoperability of the various elements, etc. This process would facilitate both technological innovation (thanks to the public availability of standards) and independence from specific technologies or vendors;

- **Availability and reliability:**

Data must be available anytime and anywhere for each authorized object. Therefore, mechanisms are needed for the object's interoperability and the transfer and restoration (in case of unexpected data events). Additionally, the mobility of devices must be managed appropriately;

- **Data storage, processing, and visualization:**

New methods must be found to efficiently manage and visualize the vast amount of data coming from smart objects;

- **Scalability:**

Research on this topic serves to make it possible to add new services and devices to existing IoT systems without degrading their presentations. In particular, it is necessary to take into account constraints such as memory, computing power, bandwidth, etc.;

- **Management and self-configuration:**
The user can efficiently manage a large number of devices. Additionally, smart objects must be able to self-configure in response to external events as much as possible, always in order to simplify management;
- **Unique identification of smart objects:**
Each smart object must have unique identification in order to be reached by all the others. This process represents a problem, especially concerning the considerable increase of smart objects present;
- **Energy consumption:**
smart objects must manage energy efficiently. They often communicate with other devices via wireless technologies and have batteries as a power source that cannot always be replaced easily;
- **Security and privacy:**
As far as security is concerned, in general, we have that: Communications between devices are often wireless, and this makes it easy to “eavesdrop”; the low computing capacity they have makes it challenging to implement elaborate countermeasures. As far as privacy is concerned, one of the main problems lies in the fact that smart objects collect a considerable amount of information about users; a possible attacker could have access not only to users’ data but also to their habits and information about their health, etc. Moreover, in an IoT application, the concepts of “safety” (security of physical objects and people) and “security” (security of data and information systems) tend to converge on the same level, since the objects have gained the ability to interact with the surrounding world.
- **Ship -Shore Connectivity Ensuring:**
With the development and integration of 5G technologies with satellite networks eg Inmarsat & Iridium, connectivity must be possible not only for the requirements of GMDSS at any time. But the on-demand connectivity (despite the financial cost) to support operations of intelligent systems on board. The connectivity must have all the guarantees for availability in global coverage, all time reliability, enhanced capacity , speed (from 50 Mbps to over 1 Gbps), increased bandwidth, very low latency (below 10 milliseconds).

6.5 Contribution in Applied Electronic Engineering Science.

In the present work an attempt was made to connect on Applied Academic Level, the Edge computing Technologies on IoT systems with the requirements of Industry 4.0, as they have begun to be applied in Digital Shipping. So that an Engineer (reader) who working in the field of maritime / naval electronics, to understand how his work (design, installation, maintenance, upgrade, etc.) affects at the final target. This target is Optimal Ship Management based on the Data Driven Decisions Philosophy.

The Toolbox offered by the modern Science of Electronic Engineering, includes multiple options in hardware, software, technologies and methods for device connectivity, and the management of huge amounts of data that are constantly generated. Of course, while the focus today is on implementing IoT systems on ships and supporting them with connectivity and edge computing technologies. However, there are open issues that await solutions such as Cybersecurity, Official Standardization, etc. And until all these technologies mature, coexistence awaits us the biggest challenge.

Appendix A

Software and Hardware Technologies, who I used in Case Studies in Chapter 5.

A.1 About Jinja2.

Jinja is a fast, expressive, extensible templating engine. Special placeholders in the template allow writing code similar to Python syntax. Then the template is passed data to render the final document.²⁰⁵ It includes:

- Template inheritance and inclusion.
- Define and import macros within templates.
- HTML templates can use autoescaping to prevent XSS from untrusted user input.
- A sandboxed environment can safely render untrusted templates.
- Async support for generating templates that automatically handle sync and async functions without extra syntax.
- I18N support with Babel.
- Templates are compiled to optimized Python code just-in-time and cached, or can be compiled ahead-of-time
- Exceptions point to the correct line in templates to make debugging easier.
- Extensible filters, tests, functions, and even syntax.

Jinja's philosophy is that while application logic belongs in Python if possible, it shouldn't make the template designer's job difficult by restricting functionality too muue.²⁰⁶

²⁰⁵[https://en.wikipedia.org/wiki/Jinja_\(template_engine\)](https://en.wikipedia.org/wiki/Jinja_(template_engine))

²⁰⁶<https://jinja.palletsprojects.com/en/3.0.x/intro/>



FIGURE A.1: Jinja2 engine logo.

A.2 About Flask.

Flask is a customizable Python framework that gives developers complete control over how users access data. Flask is a "micro-framework" based on Werkzeug's WSGI toolkit and Jinja 2's templating engine. It is designed as a web framework for RESTful API development²⁰⁷.

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools.²⁰⁸



FIGURE A.2: Fask web api logo.

²⁰⁷<https://opensource.com/article/19/11/python-web-api-flask>

²⁰⁸[https://en.wikipedia.org/wiki/Flask_\(web_framework\)](https://en.wikipedia.org/wiki/Flask_(web_framework))

A.3 About Raspberry Pi Pico.

Raspberry Pi Pico is a tiny, fast, and versatile board built using RP2040, a brand new microcontroller chip designed by Raspberry Pi in the UK. ²⁰⁹ [145]

RP2040 Designed by Raspberry Pi, RP2040 features a **dual-core Arm Cortex-M0+** processor with 264KB internal RAM and support for up to 16MB of off-chip Flash. A wide range of flexible I/O options includes I2C, SPI, and — uniquely — Programmable I/O (PIO). These support endless possible applications for this small and affordable package.

Can be programmed in Thonny-IDE with MicroPython(more at A.4) or in Arduino-IDE with Arduino C (more at A.1)

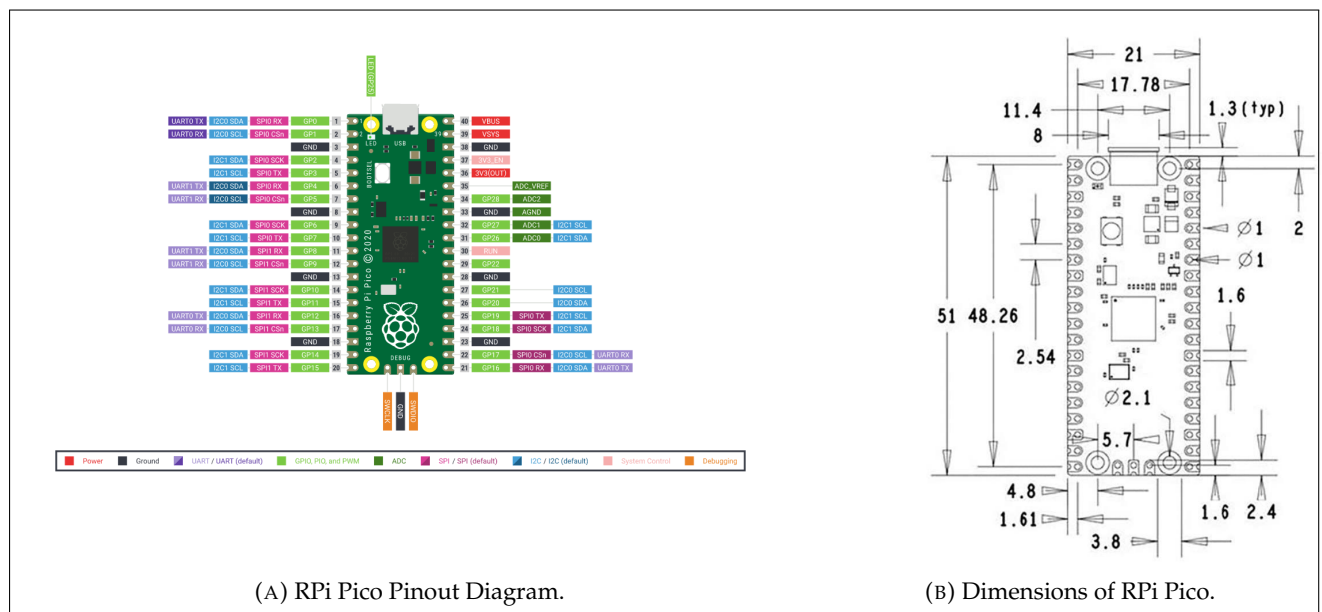


FIGURE A.3: Raspberry Pi Pico Pinout and Dimensions.(by www.raspberrypi.org)

```

1 // Demonstrates a simple use of the setup1()/loop1() functions
2 // for Raspberry Pi Pico Dual Core Programming
3 // Released to the public domain
4 float temp = 0;
5 const int numReadings = 10;
6 int readings[numReadings];
7 int readIndex = 0;           // the index of the current reading
8 int total = 0;              // the running total
9 int average = 0;
10
11 // The normal, core0 setup
12 void setup() {
13     Serial.begin(115200);
14     for (int thisReading = 0; thisReading < numReadings; thisReading++) {
15         readings[thisReading] = 0;
16     }
17     delay(5000);
18 }
19
20 void loop() {
21     temp = analogReadTemp();
22     // subtract the last reading:

```

²⁰⁹<https://www.raspberrypi.com/documentation/microcontrollers/raspberry-pi-pico.html>

```

23     total = total - readings[readIndex];
24     // read from the sensor:
25     readings[readIndex] = analogReadTemp();
26     // add the reading to the total:
27     total = total + readings[readIndex];
28     // advance to the next position in the array:
29     readIndex = readIndex + 1;
30
31     // if we are at the end of the array...
32     if (readIndex >= numReadings) {
33
34         // ...wrap around to the beginning:
35         // calculate the average:
36         average = total / numReadings;
37         // send it to the computer as ASCII digits
38         Serial.println(average);
39         readIndex = 0;
40     }
41
42     Serial.println(readIndex);
43     //Serial.printf("C0: Blue leader standing by...\n");
44     Serial.printf("Core temperature: %2.1fC\n", temp );
45     delay(1000);
46 }
47
48 // Running on core1
49 void setup1() {
50     delay(5000);
51     pinMode(LED_BUILTIN, OUTPUT);
52     Serial.printf("C1: Red leader standing by...\n");
53 }
54
55 void loop1() {
56     Serial.printf("Core2 ReadIndex: %2.1fC\n", readIndex );
57     digitalWrite(LED_BUILTIN, HIGH);
58     Serial.printf("Builtin LED is ON...\n");
59     delay(1000);
60     digitalWrite(LED_BUILTIN, LOW);
61     Serial.printf("Builtin LED is OFF...\n");
62     delay(1000);
63
64 }

```

LISTING A.1: RPi Pico dual-core programming example in Arduino-C.

A.4 About Micro Python.

MicroPython is a full implementation of the Python 3 programming language that runs directly on embedded hardware like Raspberry Pi Pico. You get an interactive prompt (the REPL) to execute commands immediately via USB Serial, and a built-in filesystem. The Pico port of MicroPython includes modules for accessing low-level chip-specific hardware.²¹⁰

MicroPython is packed full of advanced features such as an interactive prompt, arbitrary precision integers, closures, list comprehension, generators, exception handling and more. Yet it is compact enough to fit and run within just 256k of code space and 16k of RAM.

MicroPython aims to be as compatible with normal Python as possible to allow you to transfer code with ease from the desktop to a microcontroller or embedded system.²¹¹

The main disadvantage of the micro python language, is not support yet the multicore programming²¹². For this reason if needed , a developer can use the arduino c/cpp programming framework, whose except myltithreading (eg. Free RTOS library) support and multicore programming, see an example in A.1.



FIGURE A.4: Micro Python logo.

²¹⁰<https://www.raspberrypi.com/documentation/microcontrollers/micropython.html>

²¹¹<https://micropython.org/>

²¹²multithreading is available for single core

A.5 About Arduino UNO.

Arduino/Genuino Uno is a microcontroller board based on the ATmega328P [44]. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again. ²¹³

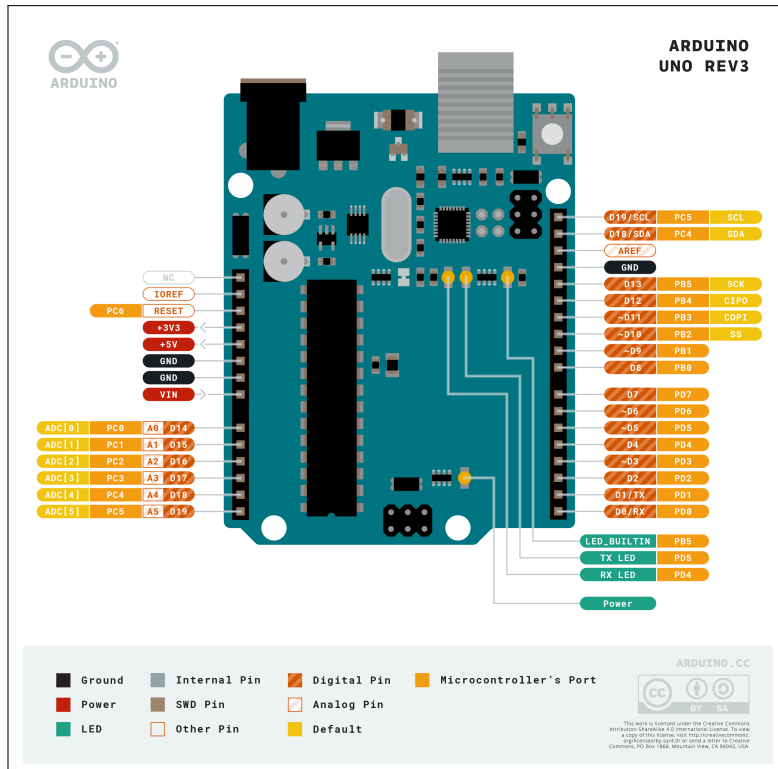


FIGURE A.5: Arduino Uno R3 Pinout Diagram.

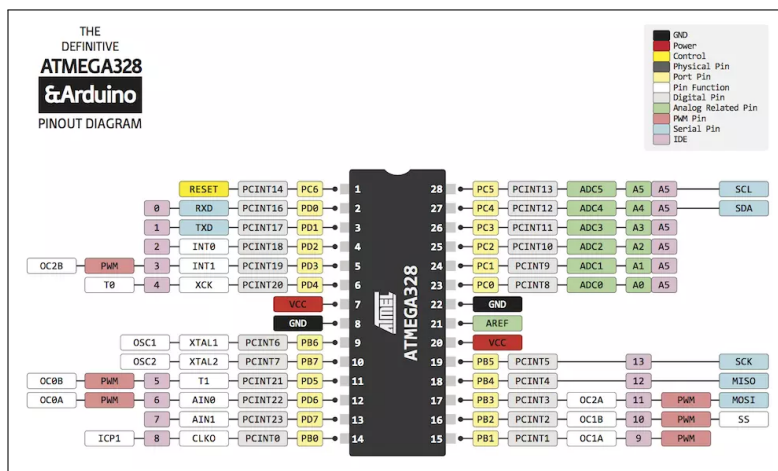


FIGURE A.6: Atmega328P Pinout Diagram.

²¹³<https://www.arduino.cc/en/pmwiki.php?n=Main/arduinoBoardUno>

Appendix B

Relative Maritime Organizations, Authorities and Systems.

Here i include the main list of the Relative Organizations and Authorities with short description for each. Which have a very important role in the Evolution of the Shipping Industry and lead to the new era of Shipping 4.0

B.1 BIMCO

The Baltic and International Maritime Council Organization ²¹⁴ has NGO status with offices in Copenhagen, Singapore, Shanghai, Athens and London and our members range from the largest shipowners in the world to small local port agents and law firms. BIMCO's mission is to be at the forefront of global developments in shipping, providing expert knowledge and practical advice to safeguard and add value to our members' businesses and our vision is to be the chosen partner trusted to provide leadership to the global shipping industry.

B.2 EMSA

The European Maritime Safety Agency (EMSA) ²¹⁵ Founding Regulation states that the purpose of the Agency is to ensure a high, uniform and effective level of maritime safety, maritime security, prevention of, and response to, pollution caused by ships as well as response to marine pollution caused by oil and gas installations and, where appropriate, to contribute to the overall efficiency of maritime traffic and maritime transport so as to facilitate the establishment of a European Maritime Transport Space without Barriers.

B.3 GMDSS

The Global Maritime Distress and Safety System ²¹⁶ is the technical, operational and administrative structure for maritime distress and safety communications worldwide. It was established in 1988 by the International Maritime Organization (IMO) which adopted a revised text of Chapter IV of the International Convention for the Safety of Life at Sea, 1974, (SOLAS) – dealing with Radiocommunications – and was implemented globally between 1992 and 1997. The GMDSS establishes the radiocommunications equipment that ships are required to carry, how this equipment shall be maintained and how it is used, and provides the context within which governments should establish the appropriate shore-based facilities to support GMDSS communications.

²¹⁴<https://www.bimco.org/about-us-and-our-members>

²¹⁵<http://www.emsa.europa.eu>

²¹⁶<https://imso.org/gmdss/>

B.4 IMSO

The International Mobile Satellite Organization (IMSO)²¹⁷ is the inter-governmental organization whose Primary Purpose is the oversight of certain public satellite safety and security communication services provided by mobile satellite communication systems. IMSO also serves as the Coordinator for the Long Range Identification and Tracking of Ships (LRIT), appointed by the Safety of Life at Sea (SOLAS) party States at IMO to ensure the worldwide operation of the system.

B.5 IMO

The International Maritime Organization²¹⁸ – is the United Nations specialized agency with responsibility for the safety and security of shipping and the prevention of marine and atmospheric pollution by ships. IMO's work supports the UN sustainable development goals.

B.6 INMARSAT

The International Maritime Satellite²¹⁹ was set up in 1979 by the International Maritime Organization (IMO) to develop a satellite communications network for protecting lives at sea and we are deeply proud of our safety heritage.

We were the first satellite operator to meet the stringent requirements of the Global Maritime Distress and Safety System (GMDSS) and International Civil Aviation Organization (ICAO) for global safety communications

B.7 NMEA

The National Marine Electronics Association²²⁰ is a US-based marine electronics trade organization setting standards of communication between marine electronics.

B.8 OCIMF

The Oil Companies International Marine Forum (OCIMF)²²¹ is a voluntary association of oil companies with an interest in the shipment and terminalling of crude oil, oil products, petrochemicals and gas. The association was formed in April 1970 in response to the growing public concern about marine pollution, particularly by oil. In the 50 years since, OCIMF has grown to become a leading authority on safety for the global marine industry, and today has member companies and consultancy status at the International Maritime Organization (IMO)s.²²²

B.9 STO

The Science and Technology Organization²²³ is a NATO subsidiary body having the same legal status than the NATO itself, and created within the framework of the North Atlantic Treaty signed in Washington in 1949. It has been established with a view to meeting to the best advantage

²¹⁷<https://imso.org/>

²¹⁸<https://www.imo.org/>

²¹⁹<https://www.inmarsat.com/en/about/who-we-are.html>

²²⁰<http://www.nmea.org/>

²²¹<https://www.ocimf.org/about-ocimf>

²²²<http://www.nmea.org/>

²²³<https://www.sto.nato.int>

the collective needs of NATO, NATO Nations and partner Nations in the fields of Science and Technology. The STO is operated under the authority of the North Atlantic Council which has delegated the operations of the STO to a Board of Directors (the Science & Technology Board – STB) comprising the NATO Nations S&T managers. The STB is chaired by the NATO Chief Scientist who is a high level recognized S&T leader of a NATO Nation, being permanently assigned to the NATO headquarters in Brussels and also serving as the senior scientific advisor to the NATO leadership.

B.10 SOLAS

The International Convention for the Safety of Life at Sea (SOLAS) is an international maritime treaty that sets minimum safety standards in the construction, equipment and operation of merchant ships. The convention requires signatory flag states to ensure that ships flagged by them comply with at least these standards.

The current version of SOLAS is the 1974 version, known as SOLAS 1974, which came into force on 25 May 1980. As of November 2018, SOLAS 1974 had 164 contracting states, which flag about 99% of merchant ships around the world in terms of gross tonnage.

SOLAS in its successive forms is generally regarded as the most important of all international treaties concerning the safety of merchant ships.²²⁴

B.11 AIS.

The automatic identification system (AIS) is an automatic tracking system that uses transceivers on ships and is used by vessel traffic services (VTS) and the vessels ECDIS system. With AIS, static and dynamic vessel information can be electronically exchanged between AIS-receiving stations (onboard, ashore or satellite). Information provided by AIS equipment, such as unique identification, position, course, and speed, can be displayed on a screen or an electronic chart display and information system (ECDIS). AIS is intended to assist a vessel's watch-standing officers and allow maritime authorities to tracking and monitoring vessel movements.

When satellites are used to detect AIS signatures, the term Satellite-AIS (S-AIS) is used. AIS information supplements marine radar, which continues to be the primary method of collision avoidance for water transport.^{225 226 227 228}

B.12 LRIT

The Long-Range Identification and Tracking (LRIT) system provides for the global identification and tracking of ships to enhance security of shipping and for the purposes of safety and marine environment protection.

The obligations of ships to transmit LRIT information and the rights and obligations of SOLAS Contracting Governments and of Search and rescue services to receive LRIT information are established in regulation V/19-1 of the 1974 SOLAS Convention.^{229 230 231}

²²⁴<https://www.imo.org/en/KnowledgeCentre/ConferencesMeetings/Pages/SOLAS.aspx>

²²⁵<https://artes.esa.int/satellite-\T1\textendash-automatic-identification-system-satais-overview>

²²⁶https://en.wikipedia.org/wiki/Automatic_identification_system

²²⁷<https://www.navcen.uscg.gov/?pageName=aismain>

²²⁸<https://help.marinetraffic.com/hc/en-us/articles/204581828-What-is-the-Automatic-Identification-System-AIS->

²²⁹<https://www.imo.org/en/OurWork/Safety/Pages/LRIT.aspx>

²³⁰<https://nauticalclass.com/difference-between-lrit-and-ais-system/>

²³¹<https://www.marineinsight.com/maritime-law/the-long-range-tracking-and-identification-lrit-system-tracking->

Appendix C

Relative Technological Challenges.

C.1 Cybersecurity on Maritime 4.0.

As IMO states in its article ²³²:

Maritime cyber risk refers to a measure of the extent to which a technology asset could be threatened by a potential circumstance or event, which may result in shipping-related operational, safety or security failures as a consequence of information or systems being corrupted, lost or compromised.

Cyber risk management means the process of identifying, analyzing, assessing and communicating a cyber-related risk and accepting, avoiding, transferring or mitigating it to an acceptable level, considering costs and benefits of actions taken to stakeholders

The overall goal is to support safe and secure shipping, which is operationally resilient to cyber risks.

More reading at [50, 51, 59, 80, 91, 96, 108, 116, 143, 148, 150, 154, 163, 170], but not limited. ^{233 234}



FIGURE C.1: Cyber Security weak point in a vessel, by gard.no.

²³²<https://www.imo.org/en/OurWork/Security/Pages/Cyber-security.aspx>

²³³<https://www.mitags.org/guide-ship-cybersecurity/>

²³⁴[https://www.gard.no/Content/25634225/Cyber20Security_Presentation20\(ID201418279\).pdf](https://www.gard.no/Content/25634225/Cyber20Security_Presentation20(ID201418279).pdf)

C.2 Maritime Autonomous Surface Ships (MASS).

As EMSA states in its article ²³⁵:

Increased automation on-board ships, which could reach ultimately full autonomy or become remotely controlled unmanned vessels, are not a new maritime safety issue. As a matter of fact, the IMO's Maritime Safety Committee discussed automated ships as early as 1964. However, the recent technological breakthroughs in the fields of information technologies, digitization and machine learning, notably supported by EU funded research, have opened the possibility of a practical implementation of some of these solutions to MASS (Maritime Autonomous Surface Ships).

More reading at [109, 123, 124, 131, 148].

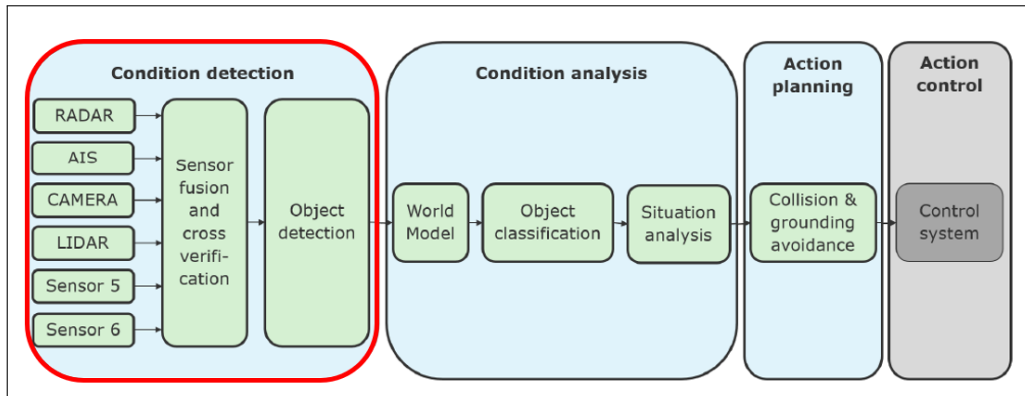


FIGURE C.2: MASS Control Scheme by EMSA.

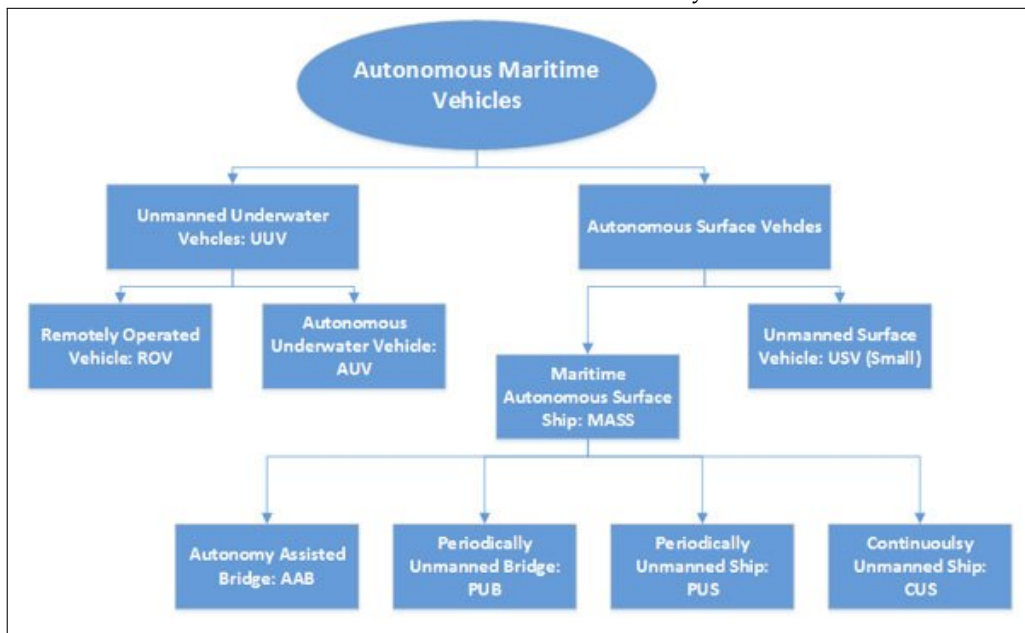


FIGURE C.3: Classification of autonomous maritime system and autonomous ship types (O.J. Rødseth, H. Nordahl, 2017).

²³⁵<http://emsa.europa.eu/mass.html>

C.3 How can use Blockchain Technology.

C.3.1 Blockchain definition.

A blockchain is a distributed database that is shared among the nodes of a computer network. Blockchains are best known for their crucial role in cryptocurrency systems, such as Bitcoin, for maintaining a secure and decentralized record of transactions. The innovation with a blockchain is that it guarantees the fidelity and security of a record of data and generates trust without the need for a trusted third party.

One key difference between a typical database and a blockchain is how the data is structured. A blockchain collects information together in groups, known as blocks, that hold sets of information. Blocks have certain storage capacities and, when filled, are closed and linked to the previously filled block, forming a chain of data known as the blockchain. All new information that follows that freshly added block is compiled into a newly formed block that will then also be added to the chain once filled. ²³⁶

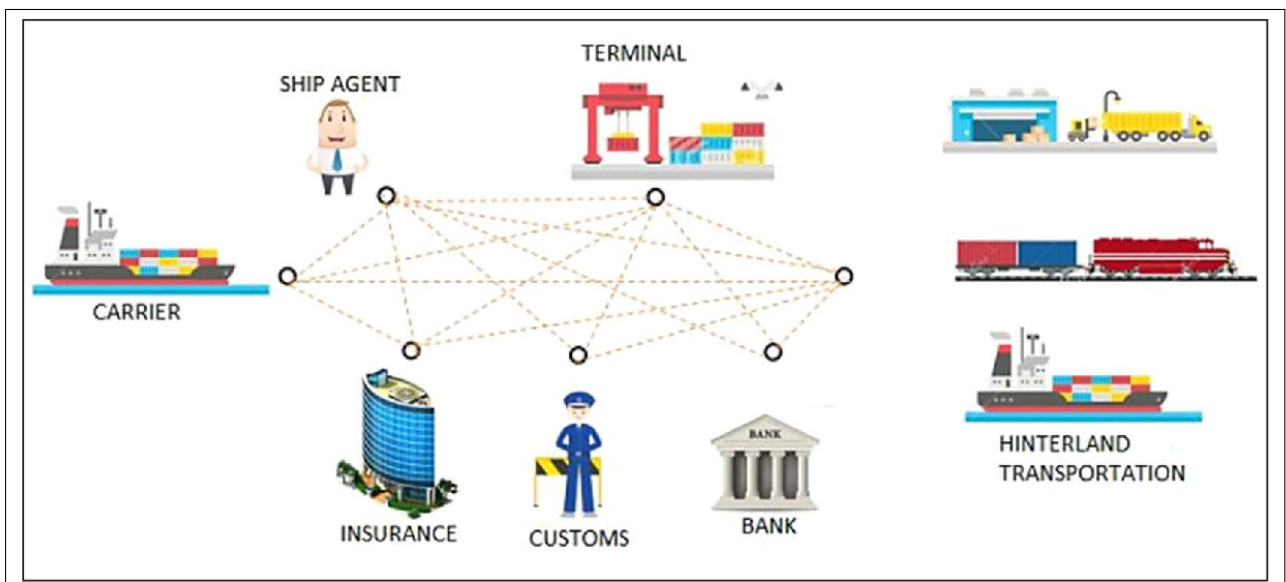


FIGURE C.4: Blockchain information flow in the import carrier process (Oude Weernink et al., 2017)

²³⁶<https://www.investopedia.com/terms/b/blockchain.asp>

C.3.2 Blockchain in Maritime 4.0

The blockchain technology will transform the logistics industry, include the shipping.

The shipping industry is beset with trust problems. Whether it is fake seafarer certificates, bad bunkers, or forged bills of lading, blockchain offers a viable solution.

On top of the trust problem, blockchain can streamline the bloated paperwork requirement that comes with operating ships around the world. Looking in nearby future, as adoption grows in other industries, blockchain will continue to be adopted in maritime too.

More reading at [139, 148]²³⁷, but not limited.

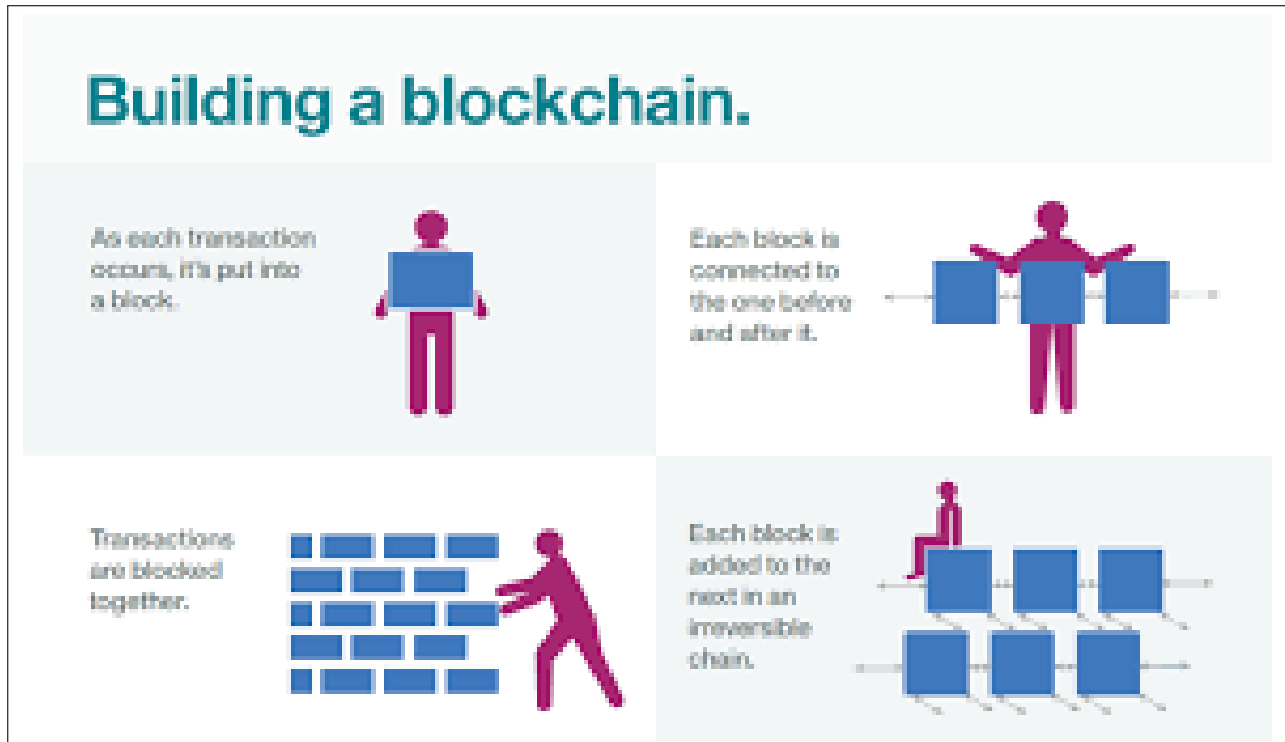


FIGURE C.5: Building a Blockchain. (by faoglobal.com)

²³⁷<https://www.hellenicshippingnews.com/blockchain-and-shipping-can-it-work/>

Appendix D

Bibliography

Notes for Bibliography.

The bibliography is classified based by fields "Type/Year/Name/Title".

The [xxx] before references, is an time sorting index number from older to newest.

The (Cited X times), is a count number, how many times i have use each refer in my thesis.

D.1 Article's.

- [4] Ornulf Jan Rodseth, Ommund Ogaard, and Jan Olav Hallset. "Integrated Ship Control and Open Systems". In: *IFAC Proc. Vol.* 25.3 (1992), pp. 5–14. ISSN: 14746670. DOI: [10.1016/s1474-6670\(17\)50266-9](https://doi.org/10.1016/s1474-6670(17)50266-9). (Cited 1 time)
- [5] E Haaland and J R dseth. "Maritime Information Technology Standard (MITS)". In: *Marit. Commun. Control.* (1993), pp. 1–12. URL: <http://www.mits-forum.org/resources/icmes93-proof.pdf>. (Cited 1 time)
- [8] I. F. Akyildiz et al. "Wireless sensor networks: A survey". In: *Comput. Networks* 38.4 (2002), pp. 393–422. ISSN: 13891286. DOI: [10.1016/S1389-1286\(01\)00302-4](https://doi.org/10.1016/S1389-1286(01)00302-4). (Cited 1 time)
- [9] Heiko Müller and Johann-christoph Freytag. "Problems, Methods, and Challenges in Comprehensive Data Cleansing". In: *Challenges HUB-IB-164* (2003), pp. 1–23. URL: http://www.dbis.informatik.hu-berlin.de/fileadmin/research/papers/techreports/2003-hub{_}ib{_}164-mueller.pdf. (Cited 1 time)
- [13] Andrzej Felski Josef Urbanski Waclaw Morgas. "Maritime Navigation Safety and Security Management". In: *Annual of Navigation.* (2007), p. 12. URL: https://www.researchgate.net/publication/233906647_Maritime_navigation_Its_safety_and_security_management. (Cited 1 time)
- [15] Chung-Chiun Liu S Gary W. Hunter, Joseph R. Stetter, Peter J. Hesketh. "Smart Sensor Systems". In: *Electrochem. Soc.* (2008), pp. 1–6. DOI: [10.1002/9780470866931](https://doi.org/10.1002/9780470866931). URL: https://www.electrochem.org/dl/interface/wtr/wtr10/wtr10_p029-034.pdf. (Cited 1 time)
- [16] Marco Guerriero et al. "Radar/AIS data fusion and SAR tasking for maritime surveillance". In: *11th International Conference on Information Fusion* (2008), pp. 1650–1654. URL: <https://ieeexplore.ieee.org/document/4632409?arnumber=4632409>. (Cited 1 time)
- [17] Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. "Wireless sensor network survey". In: *Comput. Networks* 52.12 (2008), pp. 2292–2330. ISSN: 13891286. DOI: [10.1016/j.comnet.2008.04.002](https://doi.org/10.1016/j.comnet.2008.04.002). (Cited 3 times)
- [20] K. E. Fjørtoft, B. Kvamstad, and F. Bekkadal. "Maritime communication to support safe navigation". In: *Mar. Navig. Saf. Sea Transp.* (2009), pp. 285–290. ISSN: 2083-6473. DOI: [10.1201/9780203869345.ch51](https://doi.org/10.1201/9780203869345.ch51). (Cited 1 time)
- [21] F. Bekkadal. "Novel maritime communications technologies". In: *Conference: Microwave Symposium (MMS), 2010 Mediterranean* (2010). DOI: [10.1109/MMW.2010.5605161](https://doi.org/10.1109/MMW.2010.5605161). URL: https://www.researchgate.net/publication/224183644_Novel_maritime_communications_technologies. (Cited 1 time)
- [23] Kyay Mone et al. "Clustering Analysis and Identification of Marine Traffic Congested Zones at Wusongkou, Shanghai". In: *Zeszyty Naukowe Akademii Morskiej w Gdyni* 67. January 2010 (2010), pp. 101–113. (Cited 1 time)
- [25] Ornulf Jan Rodseth, Fjørtoft Kay Endre, and Maria A Lambrou. "Web technologies for Maritime Single Windows". In: *3rd Int. Marit. Technol. Dev. Conf. MTEC 2011* (2011), p. 7. URL: <http://www.mits-forum.org/resources/MTEC2011-sw.pdf>. (Cited 1 time)
- [26] Bhaskar Prasad Rimal et al. "Architectural Requirements for Cloud Computing Systems: An Enterprise Cloud Approach". In: *J Grid Comput.* 9 (2011), pp. 3–26. DOI: [10.1007/s10723-010-9171-y](https://doi.org/10.1007/s10723-010-9171-y). URL: https://faculty.e-ce.uth.gr/dkatsar/papers/Springer_JNL_GRID11rjkg.pdf. (Cited 1 time)
- [28] H. Kdouh et al. "Wireless sensor network on board vessels". In: *2012 19th Int. Conf. Telecommun. ICT 2012* April (2012). DOI: [10.1109/ICTEL.2012.6221242](https://doi.org/10.1109/ICTEL.2012.6221242). (Cited 2 times)

- [33] Kamalendu Pal and Bill Karakostas. "The Use of Cloud Computing in Shipping Logistics". In: *E-Logistics E-Supply Chain Manag. Appl. Evol. Bus.* January 2013 (2013), pp. 104–124. DOI: [10.4018/978-1-4666-3914-0.ch006](https://doi.org/10.4018/978-1-4666-3914-0.ch006). URL: https://www.researchgate.net/publication/289149877_The_Use_of_Cloud_Computing_in_Shipping_Logistics. (Cited 1 time)
- [34] Giuliana Pallotta, Michele Vespe, and Karna Bryan. "Vessel pattern knowledge discovery from AIS data: A framework for anomaly detection and route prediction". In: *Entropy* 15.6 (2013), pp. 2218–2245. ISSN: 10994300. DOI: [10.3390/e15062218](https://doi.org/10.3390/e15062218). (Cited 1 time)
- [35] Michael Till Beck et al. "Mobile Edge Computing : A Taxonomy". In: *The Sixth International Conference on Advances in Future Internet*. c (2014), pp. 48–54. DOI: [10.1.1.670.9418](https://doi.org/10.1.1.670.9418). (Cited 1 time)
- [40] Xin Yan Xiong et al. "Vibration monitoring system of ships using wireless sensor networks". In: *2014 IEEE Int. Conf. Mechatronics Autom. IEEE ICMA 2014* August (2014), pp. 90–94. DOI: [10.1109/ICMA.2014.6885677](https://doi.org/10.1109/ICMA.2014.6885677). (Cited 1 time)
- [41] L. Aldous et al. "Uncertainty analysis in ship performance monitoring". In: *Ocean Engineering* 110 (2015), pp. 29–38. ISSN: 00298018. DOI: [10.1016/j.oceaneng.2015.05.043](https://doi.org/10.1016/j.oceaneng.2015.05.043). URL: <http://dx.doi.org/10.1016/j.oceaneng.2015.05.043>. (Cited 2 times)
- [46] B. Habtemariam et al. "Measurement level AIS/radar fusion". In: *Signal Processing* 106 (2015), pp. 348–357. ISSN: 01651684. DOI: [10.1016/j.sigpro.2014.07.029](https://doi.org/10.1016/j.sigpro.2014.07.029). URL: <http://dx.doi.org/10.1016/j.sigpro.2014.07.029>. (Cited 1 time)
- [47] Bo-Kyeong Lee et al. "Survey and Analysis of User Opinion for the Review and Modernization of GMDSS and Implementation of e-Navigation". In: *J. Korean Soc. Mar. Environ. Saf.* 21.4 (2015), pp. 381–388. ISSN: 1229-3431. DOI: [10.7837/kosomes.2015.21.4.381](https://doi.org/10.7837/kosomes.2015.21.4.381). (Cited 1 time)
- [49] Ola Salman et al. "Edge computing enabling the Internet of Things". In: *IEEE World Forum Internet Things, WF-IoT 2015 - Proc.* June 2016 (2015), pp. 603–608. DOI: [10.1109/WF-IoT.2015.7389122](https://doi.org/10.1109/WF-IoT.2015.7389122). (Cited 2 times)
- [52] Alessio Botta et al. "Integration of Cloud computing and Internet of Things: A survey". In: *Futur. Gener. Comput. Syst.* (2016), pp. 1–17. ISSN: 0167739X. DOI: [10.1016/j.future.2015.09.021](https://doi.org/10.1016/j.future.2015.09.021). (Cited 1 time)
- [54] Victor Chang, Yen Hung Kuo, and Muthu Ramachandran. "Cloud computing adoption framework: A security framework for business clouds". In: *Futur. Gener. Comput. Syst.* 57 (2016), pp. 24–41. ISSN: 0167739X. DOI: [10.1016/j.future.2015.09.031](https://doi.org/10.1016/j.future.2015.09.031). URL: <http://dx.doi.org/10.1016/j.future.2015.09.031>. (Cited 1 time)
- [55] Ghyslane Cherradi, Adil El Bouziri, and Azedine Boulmakoul. "Smart Data Collection Based on IoT Protocols". In: *Jdsi'16, Issn 2509-2103* November 2017 (2016). URL: https://www.researchgate.net/publication/320865914_Smart_Data_Collection_Based_on_IoT_Protocols. (Cited 1 time)
- [57] Erico N. De Souza et al. "Improving fishing pattern detection from satellite AIS using data mining and machine learning". In: *PLoS One* 11.7 (2016), pp. 1–20. ISSN: 19326203. DOI: [10.1371/journal.pone.0158248](https://doi.org/10.1371/journal.pone.0158248). (Cited 1 time)
- [58] Manuel Díaz, Cristian Martín, and Bartolomé Rubio. "State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing". In: *J. Netw. Comput. Appl.* 67 (2016), pp. 99–117. ISSN: 10958592. DOI: [10.1016/j.jnca.2016.01.010](https://doi.org/10.1016/j.jnca.2016.01.010). URL: <http://dx.doi.org/10.1016/j.jnca.2016.01.010>. (Cited 1 time)
- [59] Joseph Dizenzo, Dana A. Goward, and Fred S. Roberts. "The little-known challenge of maritime cyber security". In: (2016), pp. 1–5. DOI: [10.1109/IISA.2015.7388071](https://doi.org/10.1109/IISA.2015.7388071). (Cited 1 time)
- [60] Robert Grandin, Tim Gray, and Ron Roberts. "Simulating UT measurements from bolthole cracks". In: *AIP Conf. Proc.* 1706 (2016). ISSN: 15517616. DOI: [10.1063/1.4940503](https://doi.org/10.1063/1.4940503). (Cited 2 times)
- [62] Luca Oneto et al. "Vessel monitoring and design in industry 4.0: A data driven perspective". In: *2016 IEEE 2nd Int. Forum Res. Technol. Soc. Ind. Leveraging a Better Tomorrow, RTSI 2016* (2016). DOI: [10.1109/RTSI.2016.7740594](https://doi.org/10.1109/RTSI.2016.7740594). (Cited 1 time)
- [63] Luca Oneto et al. "Vessel monitoring and design in industry 4.0: A data driven perspective". In: *2016 IEEE 2nd International Forum on Research and Technologies for Society and Industry Leveraging a Better Tomorrow, RTSI 2016* (2016), p. 7740594. DOI: [10.1109/RTSI.2016.7740594](https://doi.org/10.1109/RTSI.2016.7740594). (Cited 1 time)
- [64] Xiang Sun and Nirwan Ansari. "EdgeIoT: Mobile Edge Computing for the Internet of Things". In: *IEEE Commun. Mag.* 54.12 (2016), pp. 22–29. ISSN: 01636804. DOI: [10.1109/MCOM.2016.1600492CM](https://doi.org/10.1109/MCOM.2016.1600492CM). (Cited 2 times)
- [65] Dimitrios Zissis, Elias K. Xidias, and Dimitrios Lekkas. "Real-time vessel behavior prediction". In: *Evol. Syst.* 7.1 (2016), pp. 29–40. ISSN: 18686486. DOI: [10.1007/s12530-015-9133-5](https://doi.org/10.1007/s12530-015-9133-5). (Cited 2 times)
- [70] Panos Deligiannis. "Ship performance indicator". In: *Marine Policy* 75 (2017), pp. 204–209. ISSN: 0308597X. DOI: [10.1016/j.marpol.2016.02.027](https://doi.org/10.1016/j.marpol.2016.02.027). URL: <http://dx.doi.org/10.1016/j.marpol.2016.02.027>. (Cited 2 times)

- [71] Koustabh Dolui and Soumya Kanti Datta. "Comparison of edge computing implementations: Fog computing, cloudlet and mobile edge computing". In: *GloTS 2017 - Glob. Internet Things Summit, Proc.* (2017). DOI: [10.1109/GIOTS.2017.8016213](https://doi.org/10.1109/GIOTS.2017.8016213). (Cited 3 times)
- [72] Corentin Dupont, Raffaele Giaffreda, and Luca Capra. "Edge computing in IoT context: Horizontal and vertical Linux container migration". In: *GloTS 2017 - Glob. Internet Things Summit, Proc.* (2017), pp. 2–5. DOI: [10.1109/GIOTS.2017.8016218](https://doi.org/10.1109/GIOTS.2017.8016218). (Cited 1 time)
- [78] Oltjon Kodheli, Alessandro Guidotti, and Alessandro Vanelli-Coralli. "Integration of Satellites in 5G through LEO Constellations". In: 2018-Janua (2017), pp. 1–6. DOI: [10.1109/GLocom.2017.8255103](https://doi.org/10.1109/GLocom.2017.8255103). arXiv: [1706.06013](https://arxiv.org/abs/1706.06013). (Cited 1 time)
- [79] Maria Lambrou. "Shipping 4.0: Technology Stack and Digital Innovation Challenges". In: *IAME 2017 Conf. June* (2017), pp. 1–20. URL: https://www.researchgate.net/publication/320102036_Shipping_40_Technology_Stack_and_Digital_Innovation_Challenges. (Cited 1 time)
- [80] Jie Lin et al. "A Survey on Internet of Things: Architecture, Enabling Technologies, Security and Privacy, and Applications". In: *IEEE Internet Things J.* (2017), pp. 1–17. ISSN: 23274662. DOI: [10.1109/JIOT.2017.2683200](https://doi.org/10.1109/JIOT.2017.2683200). (Cited 2 times)
- [82] N.G.H. Meyendorf et al. "NDE 4.0 - NDE for the 21st Century – The internet of things and cyber physical systems will revolutionize NDE". In: *15th Asia Pacific Conf. Non-Destructive Test.* (2017), pp. 1–8. URL: https://lib.dr.iastate.edu/cnde_conf/117/. (Cited 2 times)
- [83] Norbert Meyendorf et al. "Using Remote NDE, including External Experts in the Inspection Process, to Enhance Reliability and address Today's NDE Challenges". In: *7th Eur. Work. Reliab. NDE* (2017), pp. 1–7. URL: https://lib.dr.iastate.edu/cnde_conf/118. (Cited 2 times)
- [84] Kostas Patroumpas et al. "Online event recognition from moving vessel trajectories". In: *Geoinformatica 21.2* (2017), pp. 389–427. ISSN: 13846175. DOI: [10.1007/s10707-016-0266-x](https://doi.org/10.1007/s10707-016-0266-x). arXiv: [1601.06041](https://arxiv.org/abs/1601.06041). (Cited 2 times)
- [85] Mahadev Satyanarayanan. "The emergence of edge computing". In: *Computer (Long. Beach. Calif.)*. 50.1 (2017), pp. 30–39. ISSN: 00189162. DOI: [10.1109/MC.2017.9](https://doi.org/10.1109/MC.2017.9). (Cited 1 time)
- [87] Ruolan Zhang and Masao Furusho. "Conversion Timing of Seafarer's Decision-making for Unmanned Ship Navigation". In: *TransNav, Int. J. Mar. Navig. Saf. Sea Transp.* 11.3 (2017), pp. 463–468. ISSN: 2083-6473. DOI: [10.12716/1001.11.03.11](https://doi.org/10.12716/1001.11.03.11). (Cited 1 time)
- [88] Nirwan Ansari and Xiang Sun. "Mobile edge computing empowers internet of things". In: *IEICE Trans. Commun.* E101B.3 (2018), pp. 604–619. ISSN: 17451345. DOI: [10.1587/transcom.2017NRI0001](https://doi.org/10.1587/transcom.2017NRI0001). arXiv: [arXiv:1709.00462v2](https://arxiv.org/abs/1709.00462v2). (Cited 1 time)
- [89] Hany Atlam, Robert Walters, and Gary Wills. "Fog Computing and the Internet of Things: A Review". In: *Big Data Cogn. Comput.* 2.2 (2018), p. 10. ISSN: 2504-2289. DOI: [10.3390/bdcc2020010](https://doi.org/10.3390/bdcc2020010). (Cited 3 times)
- [90] Dominik Filipiak, Milena Stró, and W Krzysztof. "Anomaly Detection in the Maritime Domain : Comparison of Traditional and Anomaly Detection in the Maritime Domain : Comparison of Traditional and Big Data Approach". In: *Sto-Mp-Ist-160 June* (2018), pp. 1–13. URL: https://www.researchgate.net/publication/325848020_Anomaly_Detection_in_the_Maritime_Domain_Comparison_of_Traditional_and_Big_Data_Approach. (Cited 1 time)
- [91] Rwei Hau Hsu et al. "Reconfigurable Security: Edge-Computing-Based Framework for IoT". In: *IEEE Netw.* 32.5 (2018), pp. 92–99. ISSN: 1558156X. DOI: [10.1109/MNET.2018.1700284](https://doi.org/10.1109/MNET.2018.1700284). arXiv: [1709.06223](https://arxiv.org/abs/1709.06223). (Cited 1 time)
- [92] He Li, Kaoru Ota, and Mianxiong Dong. "Learning IoT in Edge: Deep Learning for the Internet of Things with Edge Computing". In: *IEEE Netw.* (2018), pp. 96–101. ISSN: 08908044. DOI: [10.1109/MNET.2018.1700202](https://doi.org/10.1109/MNET.2018.1700202). URL: <http://hdl.handle.net/10258/00009928>. (Cited 1 time)
- [93] Shangbo Mao et al. "An Automatic Identification System (AIS) Database for Maritime Trajectory Prediction and Data Mining". In: (2018), pp. 241–257. DOI: [10.1007/978-3-319-57421-9_20](https://doi.org/10.1007/978-3-319-57421-9_20). arXiv: [1607.03306](https://arxiv.org/abs/1607.03306). (Cited 1 time)
- [94] Judit Nagy et al. "The role and impact of industry 4.0 and the internet of things on the business strategy of the value chain-the case of hungary". In: *Sustainability (Switzerland)* 10.10 (2018). ISSN: 20711050. DOI: [10.3390/su10103491](https://doi.org/10.3390/su10103491). (Cited 2 times)
- [95] Jianli Pan and James McElhannon. "Future Edge Cloud and Edge Computing for Internet of Things Applications". In: *IEEE Internet Things J.* 5.1 (2018), pp. 439–449. ISSN: 23274662. DOI: [10.1109/JIOT.2017.2767608](https://doi.org/10.1109/JIOT.2017.2767608). (Cited 2 times)
- [96] Jianli Pan and James McElhannon. "Future Edge Cloud and Edge Computing for Internet of Things Applications". In: *IEEE Internet Things J.* 5.1 (2018), pp. 439–449. ISSN: 23274662. DOI: [10.1109/JIOT.2017.2767608](https://doi.org/10.1109/JIOT.2017.2767608). (Cited 1 time)

- [97] Jianli Pan and Zhicheng Yang. "Cybersecurity challenges and opportunities in the new "edge computing + iot" world". In: *SDN-NFVSec 2018 - Proc. 2018 ACM Int. Work. Secur. Softw. Defin. Networks Netw. Funct. Virtualization, Co-located with CODASPY 2018* 2018-Janua (2018), pp. 29–32. DOI: [10.1145/3180465.3180470](https://doi.org/10.1145/3180465.3180470). (Cited 3 times)
- [98] Ivan Panić, Jasmin Ćelić, and Aleksandar Cuculić. "Wireless condition monitoring of machinery and equipment in maritime industry: An overview". In: *Pomorstvo* 32.2 (2018), pp. 201–210. ISSN: 13320718. DOI: [10.31217/p.32.2.5](https://doi.org/10.31217/p.32.2.5). (Cited 1 time)
- [99] Kai Peng et al. "A survey on mobile edge computing: Focusing on service adoption and provision". In: *Wirel. Commun. Mob. Comput.* 2018 (2018), pp. 1–17. ISSN: 15308677. DOI: [10.1155/2018/8267838](https://doi.org/10.1155/2018/8267838). URL: <https://www.hindawi.com/journals/wcmc/2018/8267838/>. (Cited 1 time)
- [101] Andreas Schütze, Nikolai Helwig, and Tizian Schneider. "Sensors 4.0 - Smart sensors and measurement technology enable Industry 4.0". In: *J. Sensors Sens. Syst.* 7.1 (2018), pp. 359–371. ISSN: 2194878X. DOI: [10.5194/jsss-7-359-2018](https://doi.org/10.5194/jsss-7-359-2018). (Cited 1 time)
- [102] J Smallegange et al. "Big Data and Artificial Intelligence for Decision Making: Dutch Position Paper". In: *Spec. Meet. Big Data Artif. Intell. Mil. Decis. Making, 30th May to 1st June 2018* 1 (2018), pp. 1–11. URL: <http://resolver.tudelft.nl/uuid:b927ede6-28bf-4f06-9bab-76a4603facb0>. (Cited 1 time)
- [103] Christos Stergiou et al. "Secure integration of IoT and Cloud Computing". In: *Futur. Gener. Comput. Syst.* 78 (2018), pp. 964–975. ISSN: 0167739X. DOI: [10.1016/j.future.2016.11.031](https://doi.org/10.1016/j.future.2016.11.031). URL: <http://dx.doi.org/10.1016/j.future.2016.11.031>. (Cited 1 time)
- [107] Lu Yin, Qiang Ni, and Zhongliang Deng. "A GNSS/5G Integrated Positioning Methodology in D2D Communication Networks". In: *IEEE J. Sel. Areas Commun.* 36.2 (2018), pp. 351–362. ISSN: 07338716. DOI: [10.1109/JSAC.2018.2804223](https://doi.org/10.1109/JSAC.2018.2804223). (Cited 1 time)
- [108] Mohammed Al-Khalidi et al. "Securing marine data networks in an IoT environment". In: *Proc. - 2019 Int. Conf. Futur. Internet Things Cloud, FiCloud 2019* (2019), pp. 125–132. DOI: [10.1109/FiCloud.2019.00025](https://doi.org/10.1109/FiCloud.2019.00025). (Cited 1 time)
- [109] Karlo Bratić et al. "Review of autonomous and remotely controlled ships in maritime sector". In: *Transactions on Maritime Science* 8.2 (2019), pp. 253–265. ISSN: 18483313. DOI: [10.7225/toms.v08.n02.011](https://doi.org/10.7225/toms.v08.n02.011). (Cited 1 time)
- [113] Hanan Elazhary. "Review: Internet of Things (IoT), mobile cloud, cloudlet, mobile IoT, IoT cloud, fog, mobile edge, and edge emerging computing paradigms: Disambiguation and research directions". In: *J. Netw. Comput. Appl.* 128.November 2018 (2019), pp. 105–140. ISSN: 10958592. DOI: [10.1016/j.jnca.2018.10.021](https://doi.org/10.1016/j.jnca.2018.10.021). (Cited 1 time)
- [114] K. Formela, T. Neumann, and A. Weintrit. "Overview of definitions of maritime safety, safety at sea, navigational safety and safety in general". In: *TransNav* 13.2 (2019), pp. 285–290. ISSN: 20836481. DOI: [10.12716/1001.13.02.03](https://doi.org/10.12716/1001.13.02.03). (Cited 1 time)
- [115] Alessandro Guidotti et al. "Architectures and key technical challenges for 5G systems incorporating satellites". In: *IEEE Trans. Veh. Technol.* 68.3 (2019), pp. 2624–2639. ISSN: 00189545. DOI: [10.1109/TVT.2019.2895263](https://doi.org/10.1109/TVT.2019.2895263). arXiv: [1806.02088](https://arxiv.org/abs/1806.02088). (Cited 1 time)
- [116] Vikas Hassija et al. "A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures". In: *IEEE Access* 7 (2019), pp. 82721–82743. ISSN: 21693536. DOI: [10.1109/ACCESS.2019.2924045](https://doi.org/10.1109/ACCESS.2019.2924045). (Cited 2 times)
- [117] Nozhan Hosseini et al. "UAV Command and Control, Navigation and Surveillance: A Review of Potential 5G and Satellite Systems". In: *IEEE Aerosp. Conf. Proc.* 2019-March.March (2019), pp. 1–10. ISSN: 1095323X. DOI: [10.1109/AERO.2019.8741719](https://doi.org/10.1109/AERO.2019.8741719). (Cited 1 time)
- [118] Shuchen Liu et al. "Case Study: Networked Control for Optimal Maneuvering of Autonomous Vessels". In: *IFAC-PapersOnLine* 52.8 (2019), pp. 55–60. ISSN: 24058963. DOI: [10.1016/j.ifacol.2019.08.095](https://doi.org/10.1016/j.ifacol.2019.08.095). (Cited 1 time)
- [122] Manolis Pitsikalis et al. "Composite event recognition for maritime monitoring". In: *DEBS 2019 - Proc. 13th ACM Int. Conf. Distrib. Event-Based Syst.* March (2019), pp. 163–174. DOI: [10.1145/3328905.3329762](https://doi.org/10.1145/3328905.3329762). arXiv: [1903.03078](https://arxiv.org/abs/1903.03078). (Cited 2 times)
- [123] Tina Scheidweiler et al. "Dynamic 'Standing Orders' for Autonomous Navigation System by means of Machine Learning". In: *J. Phys. Conf. Ser.* 1357.1 (2019), pp. 1–10. ISSN: 17426596. DOI: [10.1088/1742-6596/1357/1/012046](https://doi.org/10.1088/1742-6596/1357/1/012046). (Cited 1 time)
- [124] Xiongfei Shan et al. "Sea-sky line and its nearby ships detection based on the motion attitude of visible light sensors". In: *Sensors (Switzerland)* 19.18 (2019), pp. 1–23. ISSN: 14248220. DOI: [10.3390/s19184004](https://doi.org/10.3390/s19184004). (Cited 1 time)
- [127] Darren Wright et al. "Marine observing applications using ais: Automatic identification system". In: *Front. Mar. Sci.* 6.AUG (2019), pp. 1–7. ISSN: 22967745. DOI: [10.3389/fmars.2019.00537](https://doi.org/10.3389/fmars.2019.00537). (Cited 1 time)

- [128] Huifeng Wu et al. "Edge Computing in an IoT Base Station System: Reprogramming and Real-Time Tasks". In: *Complexity* 2019 (2019). ISSN: 10990526. DOI: [10.1155/2019/4027638](https://doi.org/10.1155/2019/4027638). (Cited 1 time)
- [130] Ashkan Yousefpour et al. "All one needs to know about fog computing and related edge computing paradigms: A complete survey". In: *J. Syst. Archit.* 98.December 2018 (2019), pp. 289–330. ISSN: 13837621. DOI: [10.1016/j.sysarc.2019.02.009](https://doi.org/10.1016/j.sysarc.2019.02.009). URL: <https://doi.org/10.1016/j.sysarc.2019.02.009>. (Cited 1 time)
- [131] Xinyu Zhang et al. "Decision-making for the autonomous navigation of maritime autonomous surface ships based on scene division and deep reinforcement learning". In: *Sensors (Switzerland)* 19.18 (2019). ISSN: 14248220. DOI: [10.3390/s19184055](https://doi.org/10.3390/s19184055). URL: www.mdpi.com/journal/sensors. (Cited 1 time)
- [132] Sheraz Aslam, Michalis P. Michaelides, and Herodotos Herodotou. "Internet of Ships: A Survey on Architectures, Emerging Applications, and Challenges". In: *IEEE Internet of Things Journal* 7.10 (2020), pp. 9714–9727. ISSN: 23274662. DOI: [10.1109/JIOT.2020.2993411](https://doi.org/10.1109/JIOT.2020.2993411). (Cited 1 time)
- [137] Giovanni Giambene, Sastri Kota, and Prashant Pillai. "Satellite - 5G Integration : A Network Perspective". In: 2020 (2020), pp. 1–7. (Cited 1 time)
- [138] A. Goudosis and S. K. Katsikas. "Secure ais with identity-based authentication and encryption". In: *TransNav* 14.2 (2020), pp. 287–298. ISSN: 20836481. DOI: [10.12716/1001.14.02.03](https://doi.org/10.12716/1001.14.02.03). URL: <http://www.transnav.eu>. (Cited 1 time)
- [141] Sreeram Gutha. "Efficiency and Stationing in Edge Computing." In: *Int. J. Adv. Sci. Technol.* 29.May (2020), pp. 112–119. URL: https://www.researchgate.net/publication/341113365_Efficiency. (Cited 2 times)
- [147] Rushi Mahendrakumar Patel. "Cyber Security in Domain of IoT : A Review Threats and Security". In: December (2020), pp. 0–4. DOI: [10.13140/RG.2.2.20037.47841](https://doi.org/10.13140/RG.2.2.20037.47841). (Cited 0 times)
- [148] Jorge Pena Queralta et al. "Enhancing Autonomy with Blockchain and Multi-Access Edge Computing in Distributed Robotic Systems". In: (2020), pp. 180–187. DOI: [10.1109/fmec49853.2020.9144809](https://doi.org/10.1109/fmec49853.2020.9144809). arXiv: [2007.01156](https://arxiv.org/abs/2007.01156). (Cited 4 times)
- [150] Kewei Sha et al. "A survey of edge computing-based designs for IoT security". In: *Digit. Commun. Networks* 6.2 (2020), pp. 195–202. ISSN: 23528648. DOI: [10.1016/j.dcan.2019.08.006](https://doi.org/10.1016/j.dcan.2019.08.006). URL: <https://doi.org/10.1016/j.dcan.2019.08.006>. (Cited 1 time)
- [151] Andrzej Stateczny and Witold Kazimierski. "Radar and Sonar Imaging and Processing." In: (2020), pp. 1–9. DOI: [10.3390/rs12111811](https://doi.org/10.3390/rs12111811). (Cited 1 time)
- [153] Brendan P. Sullivan et al. "Maritime 4.0 - Opportunities in digitalization and advanced manufacturing for vessel development". In: *Procedia Manufacturing* 42 (2020), pp. 246–253. ISSN: 23519789. DOI: [10.1016/j.promfg.2020.02.078](https://doi.org/10.1016/j.promfg.2020.02.078). URL: <https://doi.org/10.1016/j.promfg.2020.02.078>. (Cited 1 time)
- [154] Jonathan Tournier et al. "A survey of IoT protocols and their security issues through the lens of a generic IoT stack". In: *Internet of Things* xxxx (2020), p. 100264. ISSN: 25426605. DOI: [10.1016/j.iot.2020.100264](https://doi.org/10.1016/j.iot.2020.100264). URL: <https://doi.org/10.1016/j.iot.2020.100264>. (Cited 1 time)
- [165] Marco Lombardi, Francesco Pascale, and Domenico Santaniello. "Internet of things: A general overview between architectures, protocols and applications". In: *Inf.* 12.2 (2021), pp. 1–21. ISSN: 20782489. DOI: [10.3390/info12020087](https://doi.org/10.3390/info12020087). (Cited 2 times)
- [170] Ky Tran et al. "Marine Network Protocols and Security Risks". In: *Cybersecurity Priv. Commun.* (2021), pp. 239–251. DOI: <https://doi.org/10.3390/jcp1020013>. URL: <https://www.mdpi.com/journal/jcp>. (Cited 1 time)

D.2 BSc-MSc Thesis and PhD Dissertations

- [10] Jimmy Pettersson and I. Wainwright. "Radar Signal Processing with Graphics Processors (GPUs)". Thesis. Uppsala University, 2003, pp. 1–129. URL: <http://uu.diva-portal.org/smash/get/diva2:292558/FULLTEXT01>. (Cited 1 time)
- [42] Lucy Aldous. "Ship operational Efficiency: Performance Models and Uncertainty Analysis". PhD Dissertation. University College London, 2015, pp. 1–277. URL: https://discovery.ucl.ac.uk/id/eprint/1477486/1/Lucy%20Aldous%20Thesis_correctedv2.pdf. (Cited 2 times)
- [48] Alexandros Nikolaides. "AIS System Security (in Greek Language)". MSc Thesis. University of Piraeus, 2015, pp. 1–69. URL: <https://www.unipi.gr/unipi/en/>. (Cited 1 time)
- [61] Konstantinos Koutras. "'Effects of lightning strikes on naval ships'(in Greek Language)". BSc Thesis. University of Patras, 2016, pp. 1–88. (Cited 2 times)
- [67] Magnus Asrud. "A Programming Language for the Internet of Things". University of Oslo, 2017, p. 90. URL: <https://www.duo.uio.no/handle/10852/56894>. (Cited 1 time)

- [69] Vernon K Bumgardner. "Contributions to Edge Computing". PhD Dissertation. College of Engineering at the University of Kentucky, 2017, pp. 1–206. URL: https://uknowledge.uky.edu/cs_etds/56/. (Cited 2 times)
- [76] Asklipios Iliades. "The Correct Use Of AIS System. Answer To Security Shipping And The Fight Against Cross-Border Threats." MSc Thesis. University of Piraeus, 2017, pp. 1–67. URL: <https://www.unipi.gr/unipi/en/>. (Cited 1 time)
- [119] Ilias Lykourezos. "Electronic Navigation Systems (in Greek Language)". BSc Thesis. University West Attica , Athens Greece, 2019. (Cited 1 time)

D.3 Books List.

- [1] Merrill I. Skolnik, ed. *Introduction To Radar System*. International 1981. McGraw-Hill Book Co., 1981, p. 590. ISBN: 0070579091. DOI: [10.4324/9781315716077-2](https://doi.org/10.4324/9781315716077-2). (Cited 1 time)
- [2] ASM Handbook Committee. *Metals Handbook: Nondestructive Evaluation and Quality Control*. Vol. 17. 1989, p. 1609. ISBN: 0871700077. URL: https://www.asminternational.org/search/-/journal_content/56/10192/33375464/PUBLICATION. (Cited 2 times)
- [3] Murray I. Cole. *Algorithmic Skeletons : Structured Management of Parallel Computation*. 1991, p. 170. ISBN: 0-262-53086-4. URL: <https://homepages.inf.ed.ac.uk/mic/Pubs/skeletonbook.pdf>. (Cited 1 time)
- [6] The Global and Maritime Distress. *Understanding GMDSS The Global Maritime Distress and Safety System*. Springer Science+Business Media, LLC, 1994, p. 315. ISBN: 9780340610428. (Cited 1 time)
- [14] C. H. Chen Ultrasonic NDE Lab. and University of Massachusetts Dartmouth, eds. *Ultrasonic And Advanced Methods For Nondestructive Testing And Material Characterization*. World Scientific Publishing Co. Pte. Ltd., 2007, pp. 1–682. ISBN: 9789812704092. DOI: <https://doi.org/10.1142/6327>. URL: <https://www.worldscientific.com/worldscibooks/10.1142/6327>. (Cited 2 times)
- [18] F. Bekkadal. *Marine Navigation and Safety of Sea Transportation*. 2009. Chap. 5, pp. 307–312. ISBN: 9780429206702. DOI: [10.1201/9780203869345](https://doi.org/10.1201/9780203869345). (Cited 1 time)
- [24] Giovanni Parmigiani, Lurdes Y.T. Inoue, and Hedibert F. Lopes. *Decision Theory: Principles and Approaches*. 2010, pp. 1–372. ISBN: 9780470746684. DOI: [10.1002/9780470746684](https://doi.org/10.1002/9780470746684). (Cited 1 time)
- [31] William L. Melvin and James A. Scheer. *Principles of modern radar: Advanced techniques*. 2013, pp. 1–846. ISBN: 9781613530245. DOI: [10.1049/SBRA020E](https://doi.org/10.1049/SBRA020E). (Cited 1 time)
- [32] Tammy. Noergaard. *Embedded Systems Architecture: A Comprehensive Guide for Engineers and Programmers*. Elsevier Inc., 2013, pp. 1–657. ISBN: 978-0-12-382196-6. URL: <https://www.sciencedirect.com/book/9780123821966/embedded-systems-architecture>. (Cited 2 times)
- [36] Siegmund Brandt and Computational Methods. *Data Analysis: Statistical and Computational Methods for Scientists and Engineers*. 2014, p. 512. ISBN: N 978-3-319-03762-2 (eBook). DOI: [10.1007/978-3-319-03762-2](https://doi.org/10.1007/978-3-319-03762-2). (Cited 1 time)
- [43] Jim Blandm. *The Rust Programming Language: Fast, Safe, and Beautiful*. 2015, p. 264. ISBN: 9781491925447. (Cited 1 time)
- [45] Xiacong Fan. *Real-Time Embedded Systems: Design Principles and Engineering Practices*. Elsevier Inc., 2015, pp. 1–662. ISBN: 9780128017180. DOI: [10.1016/C2014-0-00053-6](https://doi.org/10.1016/C2014-0-00053-6). (Cited 2 times)
- [120] Redowan Mahmud and Rajkumar Buyya. "Modeling and Simulation of Fog and Edge Computing Environments Using iFogSim Toolkit". In: *Fog Edge Comput*. January 2020. 2019, pp. 433–465. ISBN: 9781119525080. DOI: [10.1002/9781119525080.ch17](https://doi.org/10.1002/9781119525080.ch17). (Cited 1 time)
- [125] Marlene Sinclair. *The internet of things*. John Wiley and Sons Ltd, 2019, p. 403. ISBN: 9781119359685. (Cited 1 time)
- [129] Carlo Cattani Yeliz Karaca. *Computational Methods For Data Analysis*. De Gruyter, 2019, p. 398. ISBN: 9783110496352. (Cited 1 time)
- [136] Jonah Gamba. *Signals and Communication Technology Radar Signal Processing for Autonomous Driving*. 2020. ISBN: 9789811391927. DOI: [10.1007/978-981-13-9193-4](https://doi.org/10.1007/978-981-13-9193-4). URL: <http://www.springer.com/series/4748>. (Cited 1 time)
- [162] Hai Deng Geng and Zhe. *Radar Networks*. Taylor and Francis Group, LLC, 2021, p. 253. ISBN: 9781420076905. DOI: [10.1201/9780429139345](https://doi.org/10.1201/9780429139345). (Cited 1 time)

D.4 Reports, White Papers etc.

- [7] IEC. *Maritime navigation and radiocommunication equipment and systems - Automatic Identification Systems (AIS) - Part 2: Class A shipborne equipment of the universal automatic identification system (AIS) - Operational and performance requirements, methods of t.* Tech. rep. Geneva, Switzerland: International Electrotechnical Commission, 2001, pp. 1–122. (Cited 2 times)
- [11] UN/CEFACT. *Recommendation and Guidelines on establishing a Single Window: To enhance the efficient exchange of information between trade and government.* 33. 2005, pp. 1–33. URL: <http://www.unece.org/cefact/>. (Cited 1 time)
- [12] Kia Boggan and Daniel M Pressel. *GPUs : An Emerging Platform for General-Purpose Computation.* Tech. rep. Army Research Laboratory, Aberdeen Proving Ground, MD USA 21005-5067, 2007, p. 50. URL: <https://apps.dtic.mil/sti/citations/ADA471188>. (Cited 1 time)
- [19] Furuno Finland. *Electronic Chart Display and Information System (Ecdis).* Technical Manual. 2009, pp. 1–504. (Cited 2 times)
- [22] Cray Inc. *The Gemini Network.* Tech. rep. 2010, pp. 1–16. URL: <https://wiki.alcf.anl.gov/parts/images/2c/Gemini-whitepaper.pdf>. (Cited 1 time)
- [27] Kay Ramstad. *Maritime Information Centre Delivery B “ MIS Processes and Cooperation Description ”.* Tech. rep. 2011, pp. 1–167. URL: <http://www.mits-forum.org/resources/MIS-B.pdf>. (Cited 1 time)
- [29] Beate Kvamstad. *The role of digital communication technology in e-Navigation.* 2. 2012, pp. 1–23. URL: <http://www.marintek.sintef.no>. (Cited 1 time)
- [30] NVS Technology AG. *No08C Receivers. Nmea Protocol Specification.* Tech. rep. 2013, pp. 1–48. (Cited 1 time)
- [38] NMEA 2000® *Standard for serial-data networking of Marine Electronic Devices - Explained.* Tech. rep. NMEA and IMEA, 2014, pp. 1–62. URL: <http://www.nmea.org/Assets/2000-explained-white-paper.pdf>. (Cited 1 time)
- [39] Project MOPS "Managing Operational Performance in Ship Management" Norwegian Marine Technology Research Institute. *Public white paper : Implementation of Shipping KPI Reporting Regime -v14.* White paper. 2014. URL: <http://www.mits-forum.org/resources/skpi-implementation-v14.pdf>. (Cited 2 times)
- [44] Atmel Corporation. *Data Sheet ATmega328P.* Tech. rep. 2015, pp. 1–294. URL: http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7810-Automotive-Microcontrollers-ATmega328P\{}_Datasheet.pdf. (Cited 1 time)
- [50] BIMCO. *The Guidelines on Cyber Security onboard Ships.* Tech. rep. 2016, p. 64. URL: www.bimco.org. (Cited 1 time)
- [51] WORLD SHIPPING COUNCIL BIMCO, CLIA, ICS, INTERCARGO, INTERMANAGER, INTERTANKO, IUMI, OCIMF. *The Guidelines on Cyber Security onboard Ships.* Tech. rep. 2016, p. 36. URL: www.bimco.org. (Cited 1 time)
- [56] Steve Corrigan. *Introduction to the Controller Area Network (CAN) Application Report Introduction to the Controller Area Network (CAN).* Tech. rep. May. Texas Instruments, 2016, pp. 1–17. URL: <https://www.ti.com/lit/an/sloa101b/sloa101b.pdf>. (Cited 1 time)
- [66] Amazon Web Services. *The FreeRTOS™ Reference Manual API Functions and Configuration Options.* 1. 2017, pp. 1–400. URL: https://www.freertos.org/fr-content-src/uploads/2018/07/FreeRTOS_Reference_Manual_V10.0.0.pdf. (Cited 1 time)
- [68] Tim Bray. *The JavaScript Object Notation (JSON) Data Interchange Format.* Tech. rep. 8259. Dec. 2017. 16 pp. DOI: [10.17487/RFC8259](https://doi.org/10.17487/RFC8259). URL: <https://www.rfc-editor.org/info/rfc8259>. (Cited 1 time)
- [73] ECMA International. *The JSON Data Interchange Syntax.* Tech. rep. December 2017. 2017, p. 8. URL: <http://www.ecma-international.org/publications/standards/Ecma-404.htm>. (Cited 1 time)
- [74] Flavio Esposito et al. "Complete edge function onloading for effective backend-driven cyber foraging. Conference Article". In: *Int. Conf. Wirel. Mob. Comput. Netw. Commun.* Vol. 2017-October. 2017, pp. 1–8. ISBN: 9781538638392. DOI: [10.1109/WiMOB.2017.8115808](https://doi.org/10.1109/WiMOB.2017.8115808). (Cited 1 time)
- [77] International Organization for Standardization. *The JSON data interchange syntax (ISO/IEC 21778:2017).* Tech. rep. 2017, pp. 1–14. (Cited 1 time)
- [86] *Start Discovering the KPI Standard.* BIMCO, 2017. URL: <https://www.bimco.org/-/media/bimco/ships-ports-and-voyage-planning/shippingkpis/shipping-kpis-discover.ashx?la=en>. (Cited 2 times)
- [100] R. Link; N. Riess. "NDT 4.0 - Significance and Implications to NDT – Automated Magnetic Particle Testing as an Example". In: *12th European Conference on Non-Destructive Testing (ECNDT 2018), Gothenburg 2018, June 11-15 (ECNDT 2018).* 2018, pp. 1–6. URL: <http://www.ndt.net/?id=22997>. (Cited 2 times)

- [110] Cisco. *Establishing the Edge*. White paper 1. 2019, pp. 1–27. DOI: [10.1038/nrn1199](https://doi.org/10.1038/nrn1199). (Cited 1 time)
- [121] Rajan Mistry. "Introduction to Real-Time Operating Systems (RTOS)". Qualcomm Technologies, Inc. 2019. URL: <https://www.embeddedcomputing.com/technology/software-and-os/introduction-to-realtime-operating-systems-rtos>. (Cited 1 time)
- [126] John Wiley. *Making Critical Decisions*. Tech. rep. © CHIRP Maritime and University College London, 2019, pp. 1–20. (Cited 1 time)
- [133] Dominique Bonte et al. *Smart Manufacturing and How To Get Started: the Implementation and Roi of Industry 4.0 Use Cases*. White paper. 2020. URL: <https://www.abiresearch.com/>. (Cited 1 time)
- [139] Erin H Green et al. *Blockchain Technology and Maritime Shipping: An Exploration of Use Cases in the U.S.* Tech. rep. June. U.S. Department of Transportation Maritime Administration, 2020, pp. 1–53. URL: <https://www.maritime.dot.gov/innovation/meta/blockchain-technology-and-maritime-shipping-exploration-use-cases-us-maritime>. (Cited 1 time)
- [143] National Cybersecurity Authority Hellenic Republic. *National Cybersecurity Strategy. (Greek Language)*. Tech. rep. 2020, pp. 1–83. (Cited 1 time)
- [145] Raspberry Pi (Trading) Ltd. *Raspberry Pi Pico Datasheet*. Tech. rep. Raspberry Pi (Trading) Ltd., 2020, pp. 1–30. URL: <https://datasheets.raspberrypi.com/pico/pico-datasheet.pdf>. (Cited 1 time)
- [152] Balakrishnan Subramaniann. *8 Popular Open Source Programming languages for IOT*. 9624670. 2020, p. 11. URL: <https://datascience.foundation/datatalk/8-popular-open-source-programming-languages-for-iot>. (Cited 1 time)
- [157] Dalaklis Dimitrios et al. "Big Data Management in the Shipping Industry : Examining Strengths Vs Weaknesses and Highlighting Relevant Business Opportunities". In: November. 2021. (Cited 1 time)
- [163] *GUIDE FOR CYBERSECURITY IMPLEMENTATION FOR THE MARINE AND OFFSHORE INDUSTRIES ABS CyberSafety TM*. Tech. rep. February. American Bureau of Shipping, 2021, pp. 1–54. URL: <https://standards.globalspec.com/std/14364455/251>. (Cited 1 time)
- [166] *Maritime Communications - A look over the horizon*. Tech. rep. GTMARITIME, 2021, pp. 1–55. URL: <https://www.gtmartime.com>. (Cited 1 time)

D.5 On Line Sources.

- [37] Nick Heath. *How IBM's Node-RED is hacking together the internet of things*. © 2021 ZDNET, A RED VENTURES COMPANY. 2014. URL: <https://www.techrepublic.com/article/node-red/>. (Cited 1 time)
- [53] C++ AMP. Microsoft. 2016. URL: <https://docs.microsoft.com/en-us/cpp/parallel/amp/cpp-amp-cpp-accelerated-massive-parallelism?view=msvc-160>. (Cited 1 time)
- [75] Mark Harris. *An Even Easier Introduction to CUDA*. NVidia Developer Blog. 2017. URL: <https://developer.nvidia.com/blog/even-easier-introduction-cuda/>. (Cited 1 time)
- [81] *Metadata and Documentation*. Axiom Data Science. 2017. URL: <https://www.axiomdatascience.com/best-practices/MetadataandDocumentation.html#metadata>. (Cited 1 time)
- [104] Elizabeth Stratiotis. *Fuel Costs in Ocean Shipping*. 2018. URL: www.morethanshipping.com/fuel-costs-ocean-shipping/. (Cited 1 time)
- [105] *Types of Network Storages: DAS, SAN and NAS*. Router-switch. 2018. URL: <https://www.router-switch.com/faq/types-of-network-storages.html>. (Cited 1 time)
- [106] Tyler Wojciechowic. "Smart Sensor vs Base Sensor - What's the Difference?" Symmetry Blog. 2018. URL: <https://www.semiconductorstore.com/blog/2018/Smart-Sensor-vs-Base-Sensor-Whats-the-Difference-Symmetry-Blog/3538/>. (Cited 1 time)
- [111] Brett Cornwright. *What is NDE 4.0 and Why is it Important?* Floodlight Software. 2019. URL: <https://floodlightsoft.com/blog/what-is-nde-4-0-and-why-is-it-important/>. (Cited 1 time)
- [112] George Crump, ed. *Understanding the IO Needs of Edge Computing*. 2019. URL: <https://storageswiss.com/2019/10/03/understanding-the-io-needs-of-edge-computing/>. (Cited 1 time)
- [134] Selmer Bringsjord and Naveen Sundar Govindarajulu. *Artificial Intelligence*. Ed. by Edward N. Zalta. 2020. URL: <https://plato.stanford.edu/archives/sum2020/entries/artificial-intelligence/>. (Cited 1 time)
- [135] Zornitsa Dimitrova. *IoT Data Collection: A Quick Guide to Current Methods and Research*. Record Evolution GmbH. 2020. URL: <https://www.record-evolution.de/en/data-collection-in-industrial-iot-contexts-methods-and-trends/>. (Cited 1 time)

- [140] Carsten Gregersen. "A Complete Guide to IoT Protocols and Standards In 2021". Nabto SA, Denmark. 2020. URL: <https://www.nabto.com/guide-iot-protocols-standards/>. (Cited 1 time)
- [142] Arman Hasen. *Storage Types (DAS, NAS and SAN)*. Network Walks Academy. 2020. URL: <https://networkwalks.com/storage-types-das-nas-san/>. (Cited 1 time)
- [144] Will Kenton. *Quantitative Analysis (QA)*. Investopedia. Nov. 2020. URL: <https://www.investopedia.com/terms/q/quantitativeanalysis.asp>. (Cited 2 times)
- [146] *NMEA 0183 vs NMEA 2000*. © Casual Navigation. 2020. URL: <https://casualnavigation.com/nmea-0183-vs-nmea-2000/>. (Cited 1 time)
- [149] *Real-Life Use Cases for Edge Computing*. IEEE. 2020. URL: <https://innovationatwork.ieee.org/real-life-edge-computing-use-cases/>. (Cited 1 time)
- [155] "What is a Real-Time Operating System (RTOS)?" *Online-White Paper*. National Instruments S.A. 2020. URL: <https://www.ni.com/en-us/innovations/white-papers/07/what-is-a-real-time-operating-system--rtos--.html>. (Cited 1 time)
- [156] Muli Ben-Yehuda. *Gain an edge on IoT storage*. Tech Target. Jan. 2021. URL: <https://internetofthingsagenda.techtarget.com/blog/IoT-Agenda/Gain-an-edge-on-IoT-storage>. (Cited 1 time)
- [158] *Embedded Operating System*. 2021. URL: <https://www.techopedia.com/definition/30014/embedded-operating-system>. (Cited 1 time)
- [159] *Embedded operating system*. Wikipedia. 2021. URL: https://en.wikipedia.org/wiki/Embedded_operating_system. (Cited 1 time)
- [160] The Editors of Encyclopedia. "Software". Encyclopedia Britannica. 2021. URL: <https://www.britannica.com/technology/software>. (Cited 1 time)
- [161] *FreeRTOS™ Real-time operating system for microcontrollers*. Copyright (C) Amazon Web Services, Inc. 2021. URL: <https://www.freertos.org>. (Cited 1 time)
- [164] *Intelligent agent*. Wikipedia. 2021. URL: https://en.wikipedia.org/wiki/Intelligent_agent. (Cited 1 time)
- [167] Tim Smith. *Qualitative Analysis*. Investopedia. Apr. 2021. URL: <https://www.investopedia.com/terms/q/quantitativeanalysis.asp>. (Cited 2 times)
- [168] Marine Digital Team. *TOP 5 factors for lower fuel costs for Shipping companies*. 2021. URL: https://marine-digital.com/article_top5factors. (Cited 1 time)
- [169] *The National Marine Electronics Association*. 2021. URL: <https://www.nmea.org/>. (Cited 1 time)
- [171] *Why CUDA Python?* NVidia Developer Blog. 2021. URL: <https://developer.nvidia.com/cuda-python>. (Cited 1 time)
- [172] © The Khronos® Group Inc. 2021. © The Khronos® Group Inc. 2021. 2021. URL: <https://www.khronos.org/opencv/>. (Cited 1 time)