Employing HDF5 File Format for Marine Engine Systems Data Storage

Giuseppe Giannino, Michelangelo Tricarico, Andrea Orlando
Innovation and Development Centre
Isotta Fraschini Motori (IFM)
Bari, Italy
E-mail: {giuseppe.giannino, michelangelo.tricarico, andrea.orlando}@isottafraschini.it

***Abstract*—This early-stage paper describes a proposal for a standard approach to store data acquired from communication protocols mostly used in marine engines control systems for propulsion and power generation. The reasons to use Hierarchical Data Format version 5 (HDF5) are explained, and some concrete applications are presented based on real assets.**

Keywords - HDF5; Marine engines; Big data; Communication protocols; File format.

I. INTRODUCTION

A marine engine can be considered the heart of a vessel, independently from its specific application, i.e., propulsion or power generation for civil or institutional usage such as cruise or warship. Due to the high complexity of these engines and the very high reliability requirements as well as the large number of sensors, a lot of data is generated and exchanged via communication protocols.

These data can be considered the base for the development of innovative and always increasing diagnostic and predictive strategies, generally based on Knowledge-based models, Artificial Intelligence, and Statistical models. All of them are aimed to fault identification by replacing the simplest and largely used maintenance approaches [1].

These latter are mostly based on scheduled maintenance operations and threshold diagnostic and don't consider all historian data, so the result is not an optimized strategy.

In this context, Isotta Fraschini Motori (IFM) [2], a Fincantieri company, is investigating and developing custom diagnostic models to be integrated within the next generation of marine engine automation and control systems.

The developments aim to be suitable not just for classical and most used diesel Internal Combustion Engines (ICE) but also for new engines based on green fuels like methanol, hydrogen, and so on.

The main purpose of this research work is to explore the applicability and usage of the HDF5 file format as base for data storage.

Many current on-board storage systems provide ASCII/.txt/.csv files as output of the acquisition and storage process. These files guarantee easy access to data, but they were never designed for the massive scale of big data and tend to eat up resources unnecessarily (e.g., CPU-intensive) The reading and writing processes are not efficient and practical, especially when high performances are required during data processing phase [3] or real-time analysis.

The main idea behind this proposal is to define a standard approach for storing data acquired across multiple communication protocols installed on board of an engine for marine purposes that can help to make the information agnostic to the specific communication protocol and then speed up and facilitate the development of diagnostic and predictive algorithms and data analysis tools.

The structure of the paper is as follows: Section II introduces general material useful to understand the next sections, therefore an overview of HDF5 file format and the mainly used and investigated communication protocols is given. Section III explains why and how HDF5 can be a suitable solution for storing data acquired from marine engines over different communication protocols. Section IV briefly presents a benchmarking analysis, in terms of amount of used storage memory, between the proposed HDF5 and .csv file format, typically used in these contexts. Section V presents a real example, developed by IFM, of a software tool for visualizing and analyzing data stored into HDF5 files and acquired from marine engines. Conclusions and considerations for future steps are reported in Section VI.

II. GENERAL CONCEPTS

In this section, the main concepts orbiting around this proposal and useful to better understand the next results are reported.

A. HDF5 file format

Hierarchical Data Format (HDF) with the version HDF5 represents the most upgraded version of this data model. The word "hierarchical" in HDF refers to its tree-like structure.

When working with huge amounts of data, the availability of a conceptual model can help to organize and manage data, by visualizing mentally the structure itself especially in case of correlations among them [4].

More specifically, HDF is a data model, file format and I/O library designed for storing, exchanging, managing, and archiving complex data including scientific, engineering, and remote sensing data. An HDF file format has two main data objects, Groups and Datasets.

Copyright (c) IARIA, 2023. ISBN: 978-1-68558-111-4

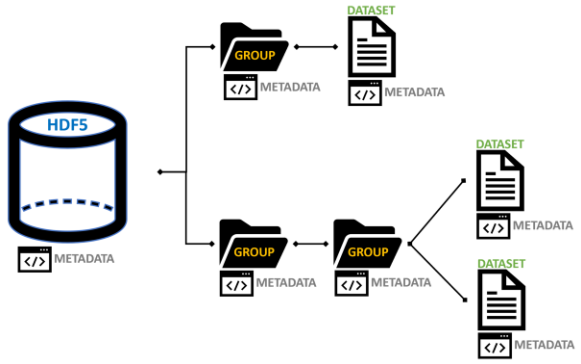

Figure 1. HDF5 - Basic components overview

Groups are the overarching structures aimed at creating and maintaining collections of related objects. Every file starts with a root group (*Root Group/)* [5].

Datasets, usually stored within groups, include a multidimensional array of elements together with additional information describing the dataset. This allows the HDF file format to be self-describing, which means that all groups and datasets can have additional information that describes their content [6]. This information is called Attributes and contain user-defined metadata, usually used to describe the nature (e.g., unit of measurement), and intended usage of dataset or group [5]. Attributes are defined based on the paradigm *key-value*, with a unique name for the specific object and a value associated. A very simple overview of the main HDF5 component is depicted in Figure 1.

An HDF file can store data consisting of different data types. Ten families or classes of HDF5 datatypes are currently supported: integer, floating-point, string, bitfield, opaque, compound (a kind of tuple type), reference, enum, variable-length sequence and array [7].

HDF5 (released in 1998 and identified as .hdf5 or .h5) has allowed to overcome the size limit of files and the number of objects in a file. HDF5 can have a flexible layout strategy defined on user needs. This can be done by using some important features of HDF5 file format as (i) specialized data storage options based on chunking, data compression, extendable arrays, and split files, (ii) Virtual File Layer (VFL) for different types of storage, such as single, multiple files, local memory, network protocol and files on parallel file system, (iii) Parallel IO through Message Passing Interface (MPI-IO). A good reference for all HDF5 details can be found on [8].

It is noteworthy to acknowledge that in the realm of big data other file formats are available, such as Parquet [9], ORC [10] and Avro [11]. These file formats provide column-wise or row-wise serialization of data, instead HDF5 stores multi-dimensional arrays, then by changing the arrangement of the arrays, the users can effectively store data either column-wise or row-wise. Matching the right storage file format is crucial since it can have impact on various fronts. In general, a good approach could be starting by analyzing the nature of data, then the data semantics and the purposes of the storage in terms of future operations on them.

In our specific application, here presented, the analysis starts from the HDF5 applicability for marine engine data storage but the idea behind this choice retains the possibility to extend the approach to the whole vessel system.

The complex concept of a vessel can be easily broken down in a combination of smaller sub-systems mounted on board of a self-consistent asset capable of operating far from mainland. Each sub-system has its own role and generates data.

In the context of building the smart vessel of the future, a common aggregation of all these data is needed to implement high level applications (e.g., fuel consumption optimization, unmanned asset, system maintenance optimization and remote assistance, safety improvements,

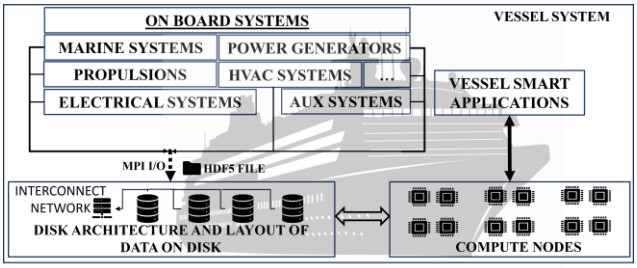

Figure 2. Centralized storage architecture for data of the future vessel sub-systems

strategic functionalities for war scenarios), as shown in Figure 2.

The hierarchical data organization, attributes storage, the multi-dimension capabilities and the intuitive data access based on the path-to-resource approach make the HDF5 a great candidate for supporting all previous concepts, especially for who will work at vessel application development level. Here, the main requirement is developing the smart functionalities with an easy and intuitive interface to data.

Further, the remarkable capabilities of parallel IO exposed by HDF5 can play a crucial role in implementing this kind of centralized system for on-board vessel data storage and processing.

Certainly, with the intention of expanding the study presented here it is not excluded that additional considerations will be done by studying and testing different data file formats, gradually the topic is explored.

B. CAN bus

Controller Area Network (CAN) is a serial communication protocol. It acts as a very common real-time communication protocol for control systems.

CAN communication has some unique characteristics that are distinct: robust real-time capability, reliable data transmission, and strong resistance to interference.

Copyright (c) IARIA, 2023. ISBN: 978-1-68558-111-4

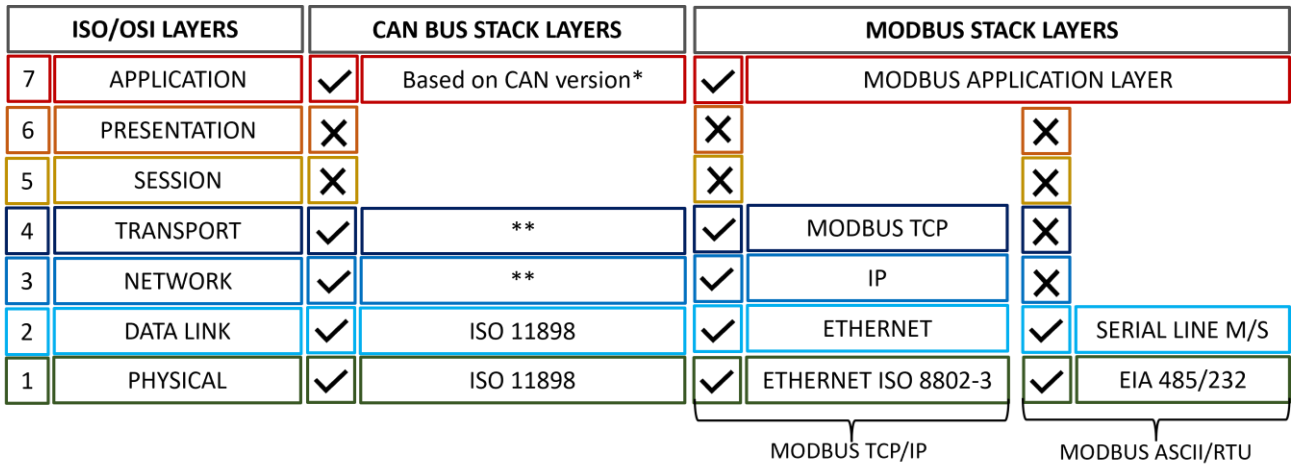

Figure 3. CAN bus and MODBUS ISO/OSI layers

It was initially introduced and used in automotive applications, then industrial automation, marine vessels, medical equipment, and industrial machinery have all utilized CAN due to its high performance and reliability.

Figure 3 is a custom IFM representation where the International Organization for Standardization (ISO) layers distribution for CAN protocol is shown.

Communication over CAN protocol is based on the principle that devices can transmit and receive messages using a shared bus, and a reliable transmission is ensured by a conflict detection mechanism. A good reference for the CAN bus protocol is the standard indicated in [12].

The primary limitation of CAN is the trade-off between transmission rate and distance. The maximum transmission rate allowed by CAN is 1 Mbps, but this rate is applicable only for shorter communication distances (less than 40 meters). If the communication distance exceeds this range, the transmission rate needs to be lowered to ensure reliable data transfer. Consequently, in long-distance communication, the transmission rate of CAN may correspondingly decrease. Hence, while CAN possess advantages such as reliability and real-time capability, it requires a balance between transmission rate and distance in long-distance communication. This trade-off is the main limit of CAN protocol.

The Society of Automotive Engineers (SAE) has developed a family of standards that pertain to the design and use of devices that transmit electronic signal and control information among vehicle components. In the field of engines, SAE has released the standard J1939, developed to exploit CAN protocol physical layer and much of the standard CAN data-link layer [13]. The maximum data rate of CAN J1939 is 250 Kbps and messages include a 29-bit identifier which defines (i) the message priority, (ii) who is the sender and (iii) what kind of data is contained within it.

C. MODBUS

MODBUS is a serial communication protocol originally developed in 1979 by Modicon for its Programmable Logic Controllers (PLC). Over the years, MODBUS has emerged as one of the most prevalent communication protocols in industrial control systems.

It operates as Master-Slave protocol, where there is a designated device acting as the master and other devices as slaves. The master device can both read and write data to the slave devices. The slave devices only transmit data upon receiving request from the master. The master assumes the role of communicator, sending inquiries and making requests to the servant. These requests may involve reading or writing data or performing specific operations. The slave provides appropriate responses accordingly. Details on MODBUS can be found in [14].

MODBUS primarily falls within layer 7 of the OSI Reference Model (the so-called “Application Layer”) and therefore is compatible with any lower-level communication protocols including EIA/TIA-232, EIA/TIA-485, Ethernet (via TCP/IP), as shown in Figure 3. Modbus data formats typically fall into two categories: ASCII and RTU format. In the first data format, ASCII characters are transmitted, where each character is represented by two bytes. With the latter, data format is transmitted in binary form.

D. Marine engine's automation & control systems topology

In Figure 4, a typical IFM automation and control system architecture for an ICE (aimed to genset and propulsion applications) is shown. The engine is a real asset produced by Isotta Fraschini Motori, the *16V170 G*.

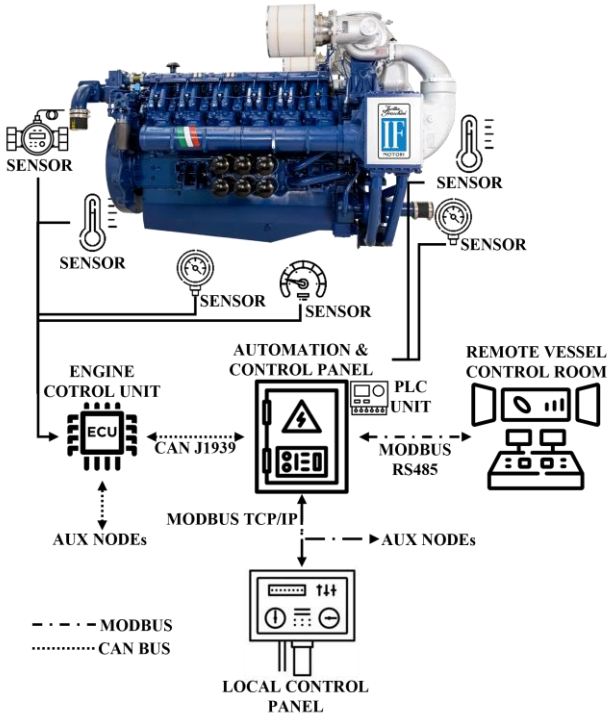

Figure 4. Marine Engine's high level topology

Recognizable are (i) automation & control panel where PLC unit and data-gathering system are located, (ii) the Engine Control Unit (ECU), (iii) local control panel where a Human Machine Interface (HMI) is positioned to facilitate the local asset control and settings, (iv) sensors and (v) communication protocols.

All these components contribute to the realization of a complex system aimed to guarantee a high level of efficiency of the asset in terms of performance, reliability, safety, and integration with vessels.

The ECU item, based on signal acquired from lots of sensors mounted on board the engine, is the heart of the full system and it's in charge of handling the injection cycles in terms of duration, fuel quantity and so on. Here, all the engine control logics and calibration maps are located and continuously executed. Then, the PLC based unit oversees all additional functionalities of the system, such as: communication with ECU, interfacing the engine with vessel central control room, running security logics based on additional sensors, managing the generated output power by interfacing the engine with the electrical machine (in case of genset application), and so on.

Generally, data can be of different nature and ranges: temperatures, pressures, speeds, flows and electrical measurements. Their treatment is not different just in terms of control logics but, as evidenced in the referenced figure, also for the communication protocols where they travel.

A so called data-gathering system is usually located in the control panel or, depending on customer requirements, it could be externally and remotely located. Its main goal is to automatically record, scan and retrieve the data with high speed and greater efficiency during asset running. A data-gathering system is designed, from the hardware and software perspective, i) to be highly and easily configurable from users based on specific application, ii) to be able to communicate with different and, at the same time, several communication protocols, iii) to be able to recover after a malfunction (e.g., node unavailable over a communication network or an anomalous condition that frozen the acquisition system), iv) to store high amount of data and cyclically delete oldest or not useful data based on user configuration.

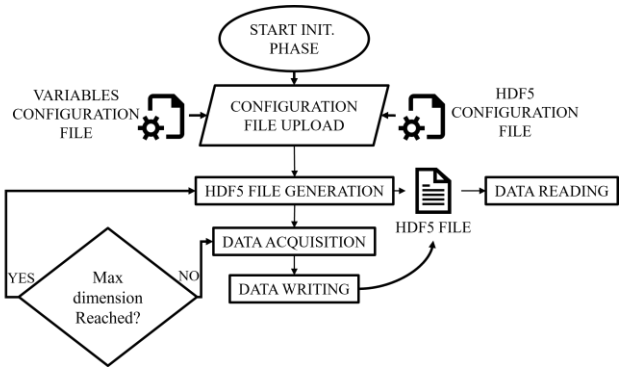

Figure 5. Working principle for data storage within HDF5 on data-gathering system

Depending on customer requirements, data could be stored locally or remotely on a cloud infrastructure. Both solutions must allow us to easily analyze data in case of troubleshooting or during scheduled maintenance operations. Further, independently from the adopted local or remote solution, the software component running on the data-gathering system, after a first check of the right connection with remote nodes (e.g., PLC, ECU, etc.) over different communication protocols, starts acquiring data, pre-processing them and storing data. If for some reason the connection with remote node is lost, then data source is down, the data-gathering system must try to restore communication and handle the storage processes.

III. HDF5 FOR COMMUNICATION PROTOCOLS DATA STORAGE

Based on previous concepts, this proposal aims to present the idea to use the HDF5 file format for storing the big amount of data acquired from the multiple communication protocols used on board marine engines.

The design of the HDF5 files starts by looking at the data organization provided for each communication protocol. Due to their differences at each level of the ISO/OSI stack the HDF5 file associated will be shaped differently.

Figure 5 shows the working principle of the data storage within an HDF5 file format, proposed, and adopted by IFM. The software entity, based on the specific user configuration where the enabled communication protocols and its parameters (Hardware interface, baud-rate, etc.) are

Copyright (c) IARIA, 2023. ISBN: 978-1-68558-111-4

defined, initiates the HDF5 file uploading a so-called *variable configuration file*. The latter is a sort of matrix where variable names, addresses, units, descriptions, and other features are defined. The HDF5 file is configured by considering all parameters handled by the software library wrapping the HDF5 definitions, such as compression, chunk, dimensions and so on.

Data starts to be acquired over communication protocols, processed, and moved into the specific position of the referenced HDF5 file. The addressing of data is acted by means of a simple POSIX-style strategy with "/" separators indicating the hierarchy level.

The design of the software entity responsible for handling the specific HDF5 file structure and transferring the data is based on the design of an appropriate HDF5 architecture, in which the groups and datasets are defined according to the specific communication protocols and system requirements.

Based on this approach, two HDF5 architectures have been designed and here proposed. The adopted strategy, to define the HDF5 architecture, starts from considering how the variables sent over the protocols are statically mapped to be later decoded.

Generally, for the kind of asset under study, marine engines, we can consider the following two variables configuration files and requirements:

- CAN J1939 - Variables are filled into a so called .dbc file [15] that allows to decode the information needed to understand a vehicle's CAN bus traffic. Here, the main information reported are frame name, frame ID, format, length, and description. Each time a new frame is acquired over CAN bus, it is firstly processed by means of .dbc file and then the extracted information is written into HDF5 file within the specific assigned position. Frames (identified in Figure 6 as groups in the HDF5 file under the group ROOT_FOLDER/) can travel with different raster time in the network (*DATASET_Time* in each group stores this information). Internally CAN bus frames are filled with information called *signals* (e.g., group CANbus_PKT_1 contains *DATASET_SGN_1* to *DATASET_SGN_n*). Signals can have different lengths and for each the .dbc file contains information about datatype, length, multiplication factor, offset, minimum and maximum values, unit, and description stored within the HDF5 as attributes.
  The CAN J1939 HDF5 architecture can be also employed for different CAN version, such as standard CAN 2.0 A or B with very low impact.
- MODBUS - Variables are filled into a configuration matrix (usually a .csv file) where name, address, datatype, multiplication factor and unit are defined. Each sampling instant (a single GROUP *Timestamp/* stores this information under the main group ROOT_FOLDER/ as shown in Figure 7) the data-gathering system reads the whole MODBUS memory addresses from the remote node. In the HDF5 file dedicated to MODBUS communication protocol, a GROUP named *MODBUS_VARIABLES/* is initialized with a certain number of DATASET equal to the number of MODBUS variables to be acquired (1 ÷ *g* in Figure 6). Acquired values are firstly

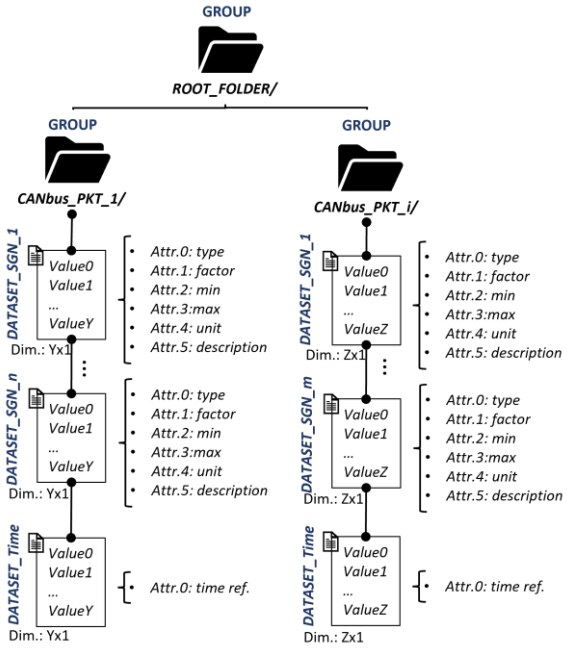

Figure 6. HDF5 proposed architecture for CAN-J1939

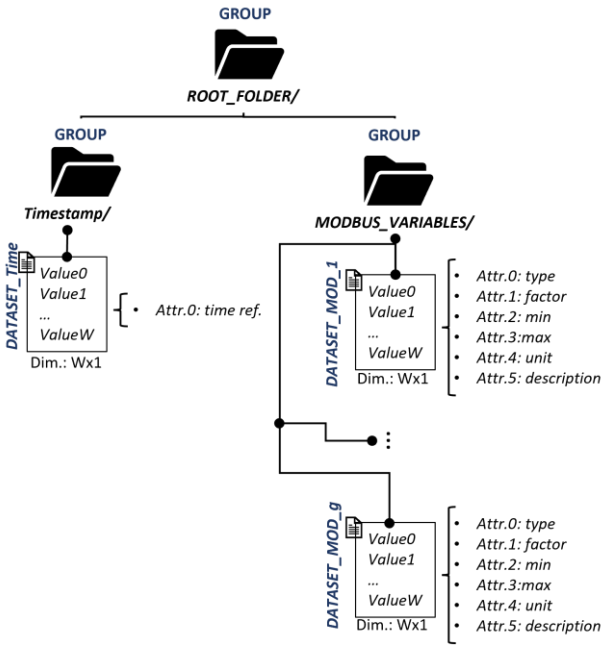

Figure 7. HDF5 proposed architecture for MODBUS

decoded by means of .csv configuration matrix and then stored into the HDF5 file within the specific dataset.

Starting from these considerations, the development has brought to obtain the HDF5 architectures reported in Figure 6 and Figure 7.

It is important to highlight that thanks to the HDF5 capabilities to store metadata under attributes associated to groups or datasets, all the information surrounding the data itself, such as units and descriptions, are not lose but filled into the HDF5 file also. This allows to keep each HDF5 file self-consistent in each moment of its life.

IV. FILE FORMATS BENCHMARKING ANALYSIS

Some preliminary considerations on the data storage and file opening time optimization are reported here. In Figure 8, a comparison with a typical file format, .csv, for storing diagnostic data on board marine engines is reported.

Within same conditions in terms of number of variables (457), sample rate (about 3.5 sec.), quantity of metadata stored (6 per variable), number of samples stored (17043 per each variable) and acquisition duration (about 16 hours) is easy to appreciate the improvement that HDF5 files allow to get. The opening time has been calculated by using the open-source HDFView software tool [8] for HDF5 file and Excel for the .csv file.

The optimization of about 20% in file dimension and more than 2 seconds less in opening time is not negligible considering that a huge amount of HDF5 files could be cyclically generated and stored on the same data-gathering system for the next analysis. Further improvements can be investigated in terms of data compression by exploring some change in the HDF5 architecture without losing the advantages related to having a well-defined data organization which allows us to easily retrieve data.

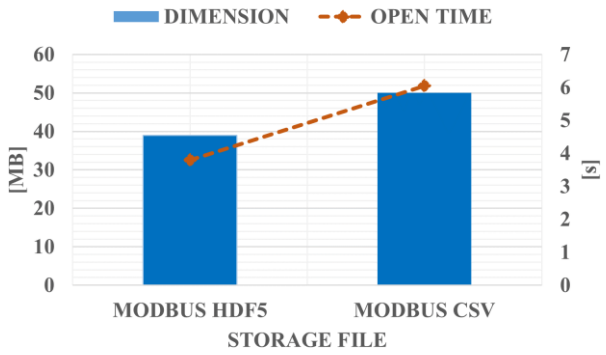

Figure 8. HDF5 vs CSV

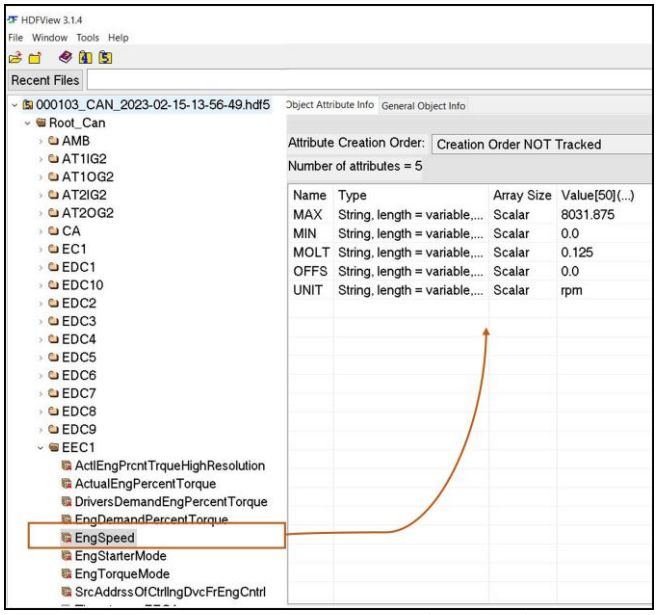

Figure 9. HDFView representation of an HDF5 file

V. EXAMPLE OF HDF5 FILE VISUALIZATION AND ANALYSIS TOOL

After data are stored into an HDF5 file to be able to visualize and analyze them a tool is required. Some open-source SW tools are available on the web. In Figure 9 a screenshot of the popular tool HDFView shows how an HDF5 file, compliant to the architecture proposed in this publication and described in section III, appears. In particular, the figure is referred to the specific CAN bus protocol based on the tree architecture illustrated in Figure 6.

Looking at the left pane, identifiable are (i) group ROOT_FOLDER/ named as *Root_Can*, (ii) group CANbus_PKT_i/ named as *EEC1* in the example and (iii) dataset DATASET_SGN_n/ named as *EngSpeed* (squared) with its specific attributes shown in the right pane of the tool. The dataset EngSpeed is filled with the data collected over CAN bus protocol (visible in HDF5View by clicking on the dataset itself).

In our case, we have developed internally a custom visualization and analysis tool capable of running on Windows and Linux-kernel based machines. The idea is to have the SW tool installed on the assistant operator laptop and/or on the local HMI of the automation and control systems to always give the possibility to study historical data stored into the HDF5 files. A time synchronization process has been implemented so that the data-gathering system is time referenced with PLC unit in turn synchronized with the vessel control systems.

Copyright (c) IARIA, 2023. ISBN: 978-1-68558-111-4

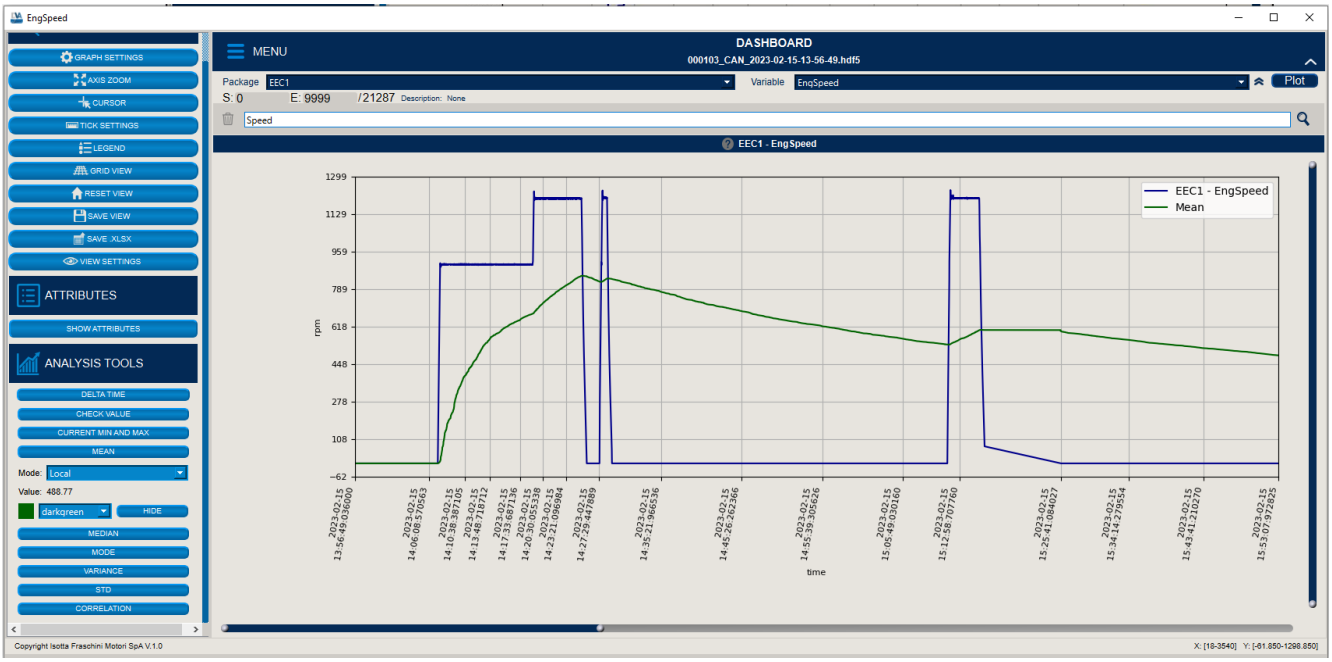

Figure 10. HDF5 - Visualization and Analysis Tool (IFM custom)

Figure 10 shows a sample screenshot of the SW tool developed in IFM. It allows us to view the stored data but also to study some statistical parameters and compare variables by time references for correlation studies. Every time a new HDF5 file is uploaded the software tool first identifies the communication protocols and recognizes the architecture. Then, when the user selects specific variables, the SW tool accesses the specific path and uploads data and metadata.

The specific data plotted in Figure 9 is the EngSpeed dataset extracted from the same HDF5 file analyzed in Figure 9.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we propose an approach to standardize the storage of data acquired over simple or complex systems where single or multiple communication protocols are involved to share functional and diagnostic information. This could speed up the process of democratizing the data usability among different vessel subsystem providers and facilitate the data processing and analysis for different end-users (e.g., maintenance operators, engineers, etc.). The advantages of using HDF5 files are mainly (i) the capability to organize data in user-friendly architecture where data can be written and read using the easy approach of path-to-resource and (ii) optimizing the data storage in terms of required memory.

The usage presented in this paper is primarily focused on marine engines, where on-board data storage and high level applications need to be implemented and connection to remote resources is not always and easily feasible for security or strategic reasons. With a wider-ranging look, it is possible to bring the same approach in different contexts such as automotive and micro-mobility.

Subject to future work further developments have been planned in IFM to optimize and deepen the usage of HDF5 file format within the specific scenario. Extended on-field tests will be executed to evaluate the impact of HDF5 file format in terms of storage optimization and ease and speed of data usage by high level applications.

ACKNOWLEDGMENT

This project is supported (was supported) by founds of *Fondo Europeo di Sviluppo Regionale Puglia* (Italy) – *POR Puglia 2014-2020*, grant number PBJMCM8 (MIR: A0101.168).

REFERENCES

[1] J.A. Pagan, “Marine Diesel Engine Diagnosis System Based on Thermodynamic Model and Artificial Intelligence", *PhD thesis, Polytechnic University of Cartagena*, 2017.

[2] Isotta Fraschini Motori Website [Online]. Available from www.isottafraschini.it (accessed August 24, 2023).

[3] A. Pfeiffer, I. Bausch-Gall, and M. Otter “Proposal for a Standard Time Series File Format in HDF5”, *Proceedings of the 9th International Modelica Conference*, pp. 495-506, September 2012, DOI 10.3384/ecp12076495.

[4] K. Läufer and K. Hinsen “Five Good Reasons to Use the Hierarchical Data Format”, Copublished by the IEEE CS and the AIP, September/October 2010, DOI 10.1109/MCSE.2010.107.

[5] M. Yang, R. E. McGrath, and M. Folk, “HDF5 – A High Performance Data Format for Earth Science”, *21st International Conference on Interactive Information*

Processing Systems (IIPS) for Meteorology, Oceanography, and Hydrology, January 2005.

[6] S. Ambatipudi and S. Byna, “A Comparison of HDF5, Zarr, and netCDF4 in Performing Common I/O Operations”, Publisched on www.arxiv.org, February 2023 (accessed August 24, 2023).

[7] M. Folk, Q. Koziol, and E. Poirmal, “An Overview of the HDF5 technology suite and its application”, Proc. EDBT/ICDT 2011 Workshop on Array Databases, pp. 36-47, March 2011, DOI 10.1145/1966895.1966900.

[8] HDFgroup Website. [Online]. Available from www.hdfgroup.org (accessed August 24, 2023).

[9] Apache Parquet Website. [Online]. Available from https://parquet.apache.org/ (accessed August 24, 2023).

[10] Apache ORC Website. [Online]. Available from https://orc.apache.org/docs/ (accessed August 24, 2023).

[11] Apache AVRO Website. [Online]. Available from https://avro.apache.org/ (accessed August 24, 2023).

[12] ISO Website. [Online]. Available from www.iso.org (accessed August 24, 2023).

[13] SAE Website. [Online]. Available from www.sae.org (accessed August 24, 2023).

[14] MODBUS Website. [Online]. Available from www.modbus.org (accessed August 24, 2023).

[15] Open Vehicles Website. [Online]. Available from https://docs.openvehicles.com/en/latest/components/vehicle_dbc/docs/dbc-primer.html (accessed August 24, 2023).

Copyright (c) IARIA, 2023. ISBN: 978-1-68558-111-4