Ocean Engineering 325 (2025) 120747

Contents lists available at ScienceDirect

Ocean Engineering

journal homepage: www.elsevier.com/locate/oceaneng

Research paper

Open testbed vessel—Reusable and generic test carrier architecture for maritime testbeds

Janusz Andrzej Piotrowski *, Christian Steger, Axel Hahn

German Aerospace Center (DLR) SE, Escherweg 2, Oldenburg, 26121, Lower Saxony, Germany

ARTICLE INFO

Keywords:
Boats
Marine safety
Marine technology
Marine vehicles
Maritime
Maritime test carrier
System testing
System architecture
Physical testbed
Verification and validation

ABSTRACT

Maritime test carriers are widely used for testing of assistance systems and autonomy systems on vessels. But in general, these test carriers are designed to test one System under Test. Especially with advancing development, a change in necessary functionalities may arise. This could include the introduction of novel sensors, actuators, navigation systems or other System under Test, which requires an extensible test carrier. To ensure extensibility and reusability, a test carrier system architecture is needed, that allows the integration of different Systems under Test, sensors and actuators taking also into account the complexity of these Systems of Systems. This paper presents a test carrier system architecture, which fulfills the requirements of reusability, portability and extensibility based on a zone-oriented Electrical/Electronic architecture from the automotive domain. The presented test carrier system architecture uses a central gateway in combination with zone domain controllers, governing specific functional zones of the test carrier. Furthermore, the integration and placement of Systems under Test inside the test carrier system architecture is identified. Finally, the test carrier system architecture is applied on two test carriers of a physical testbed and evaluated through three different use cases including an additional latency evaluation.

1. Introduction

As digitalization in the maritime industry advances, software is becoming more prominent as the main component of new developments. This is reflected by the development of Maritime Autonomous Surface Ships (MASS), where the number of software components increases with the level of automation and complexity of the overall system (Chae et al., 2020). Like shown by Zhang et al. (2021) using the example of a collision avoidance system the complexity of just this one navigational function includes several methods, which are needed to provide an adequate result. Furthermore, the overall architecture of the collision avoidance system as shown by Zhang et al. (2021) includes several subsystems, which all need to be adapted to test the collision avoidance algorithms, even if they are not directly a part of the collision avoidance functionality, like it is the case for the sensor data acquisition or the execution of control commands. In addition such systems could have high computational performance requirements, especially when they are using machine learning algorithms like for example proposed by Wang et al. (2024) to be used for the collision avoidance using reinforcement learning. To ensure the functionality of such systems, testing must be a priority, especially in the development of new systems in the context of MASS.

Testing is a key factor inside the process of verification and validation, in order to achieve a high level of trustworthiness and ensure the functionality of these novel systems. In addition, this is the only way to create a great level of confidence in the operation of such systems. Unlike in other branches of industry, the principle applies to commercial shipping where once systems have been put into operation, they are difficult to access, and errors can only be rectified with a high level of personnel and cost effort (Kang et al., 2021). To prevent this, it is essential to start early with realistic tests, including system-level and function-level testing of the system (Liu et al., 2024).

To achieve an evaluation of a System under Test (SuT) under realistic environmental conditions, the evaluation cannot just take place in simulated environments, in addition tests in the operative domain must be performed. This step is part of the testing performed during design and construction phase by the system manufacturer as defined by ISO 17894 (ISO, 2005).

The platform needed for these types of tests is in general a test carrier which is, or can be, embedded in a testbed. As defined by the e-NAV Committee of IALA a testbed consists of one or multiple ships, as well as one or multiple shore-stations and communication links that allow ship-to-ship and ship-to-shore communication. Furthermore, it is possible to include non-terrestrial components like drones or satellites in such a

* Corresponding author.
E-mail address: janusz.piotrowski@dlr.de (J.A. Piotrowski).

https://doi.org/10.1016/j.oceaneng.2025.120747
Received 17 September 2024; Received in revised form 19 February 2025; Accepted 19 February 2025
Available online 1 March 2025

testbed. AS the term test carrier is not defined by IALA G1107 (IALA, 2022), it has to be defined in context of this paper.

In context of this paper the word “test carrier” is defined as follows: A test carrier refers to a vessel that has been specifically designed and configured to provide a controlled environment for the integration, testing and evaluation of novel actuators, sensors, and other software and hardware systems. However, a test carrier is not necessarily a standalone entity, but can be integrated in a network of other components such as a remote control center of shore-based sensors.

Currently the system architecture of this test carriers is usually designed for testing one specific SuT or one category of SuT (for example sensors, novel navigation algorithms or autonomous piloting systems) on a specific test carrier, as a result, the test carrier is only built for the one SuT. The term system architecture is defined as the organizational structure of the overall system of the test carrier, consisting out of several subsystems and components, which can be either hardware or software (IEEE, 1990).

Following this, the procedures of testing are costly and time intensive. Especially regarding the diversity of ships, which differ in their dynamic behavior, size, usage and operation domain, current best practices include testing the SuT on different test carriers.

Moreover, due to the heterogeneity of maritime systems, such a test carrier system architecture must meet several requirements. On one hand, there are several data transmission standards in use, such as the IEC 61162 collection (covering NMEA standards and Ethernet), which must be integrated. On the other hand, coming from new automation concepts, not only the data transmission to the sensors and actuators, but also the connection of SuT using different types of middleware and interconnections must be considered. Since different SuT can need a connection with varying wiring and protocols, the test carrier must support them or be able to add the support of these without changing the existing architecture. At least the test carrier should be capable of integrating SuT consisting just out of software components as well as SuT, which consist of hardware and software components.

Summarizing, for the realistic testing of SuT a test carrier system architecture is needed, which is highly extensible by providing different connection possibilities to easily integrate different kinds of SuT. Using such a test carrier, the development time and effort could be decreased by reusing the test carrier without changing the overall system architecture to adapt the requirements from the SuT. Furthermore, such a test carrier could serve as a testbed on its own, provided as service to developers and other stakeholders. To enable this, an system architecture for these test carriers is needed, which provides portability and extensibility to setup ships as test carriers operating in different environments with different purposes. The extensibility comes along with reusability and portability, resulting in no test carriers have to be designed specifically for individual SuT and integration of different SuT do not need any adaptions on the test carrier. At the end the physical test carrier can be replaced, extended or equipped with different sensor, actuator and SuT setups, which enables the testing of SuT in various ways.

2. Related work

A test carrier is composed of different hard- and software components and subsystems. As they are usually built to serve the purpose of testing a single SuT, the selected sensor infrastructure as well as the middleware and interconnection of the systems is chosen for one specific use case.

Choi et al. (2020) presented a test carrier, which is used for the development of an autonomous surface vessel (ASV). A special focus is put on autonomous navigation, i.e., mapping, localization, and autonomous driving. The presented SuT aims to navigate using bathymetry, to be used as an underwater navigation system to complement other systems. For this purpose, the test carrier is equipped with additional sensor technology required for this application. This SuT is connected to a second SuT, which is responsible for controlling and navigating the test carrier and is located centrally in the system architecture. Furthermore, a component for wireless communication is connected to the SuT. The last connected component is a power and emergency module that controls the power supply to the motors installed on the test carrier. This leads to an architecture that is split up into modules allowing reconfiguration or changing of these leading to the possibility of reusing the described setup to test a different SuT. Additionally it’s also possible to extend this system by adding new software based SuT modules. As this is not explicitly mentioned it is only fulfilled with limitations.

Brushane et al. (2021) presented another example of a centrally embedded SuT. The system architecture consist mainly of a microservice concept, where the SuT is a microservice, running inside a docker container, while the communication between the microservices is based on UDP multicast. In addition the architecture does consider the integration of maritime protocols, but depends on the support of the overall microservice architecture and the protocol which is used to exchange the messages and information. in addition to the limited usage of maritime protocols, extensibility regarding the hardware is also limited. At last, also the software extensibility is limited as they need to support the microserivce-based system architecture. The portability to other test carriers is not taken into account.

An example of a test carrier that is not designed for one specific use case is presented by Brekke et al. (2022). Here the authors describe a test carrier and the corresponding system architecture that is used for developing and testing different system components such as algorithms for automated driving or other control algorithms, as well as different sensor configurations. The ship-mounted sensors distribute information to a SuT located on board in form of an algorithm for automation, as well as to a node responsible for monitoring and actual control of the ship. The automation algorithm sends control commands to this node, which is then responsible for controlling the engines. The communication is based on the robotic operating system middleware, where the extensibility is just considered if new components are integrated, which support the given middleware. With the used system architecture, the connected SuT can be either a SuT that is centrally embedded in the system architecture or a SuT that is operated on hardware connected through the network externally. However, this differs from the previously mentioned system architectures not only in the placement of the SuT but also in the reusability of the test carrier system architecture. But still the extensibility is limited to the middleware and does not plan the direct integration and support of various protocols and additional hardware, which do not have native support for the chosen middleware. The portability is considered, as the overall system architecture should be ported to the successor of the current used hardware platform. It was not designed for one specific SuT, instead it can integrate other SuT.

Likewise, Schneider et al. (2020) present a test carrier system architecture that is capable of integrating an SuT at two different locations in the system architecture. The presented test carrier architecture is designed to be used in five different use cases in the field of oceanographic tasks. For this purpose, the authors distinguish between the use of the test carrier as a mother ship, connecting a remotely operated underwater vehicle (ROUV) to it via a wired connection. In this case the ROUV becomes the SuT. On the other hand, the mother ship can be used as a stand-alone ASV itself to test driving functionality. These two use case categories require a test carrier architecture that has the possibility to integrate the SuT at two different predefined locations. Due to the software architecture design that is middleware-agnostic, the proposed architecture is extensible in regards to software. Furthermore, the architectures modular design results in reusability, as demonstrated through the application in five distinct use cases.

An example of SuT that is not directly connected to the test carrier is described by Ziebold and Gewies (2017). Here the SuT is installed on an additional industrial computer alongside the existing communication infrastructure. The test carrier is a passenger ship during its normal operation, which is equipped with additional sensors and then observed in regular operation. All sensors required for the tests are connected to a

Table 1
Fulfillment of the requirements for the different approaches with regard to the identified requirements for test carrier system architectures. Every requirement can be fulfilled (X), fulfilled with limitations (X in brackets), not fulfilled (–) and not mentioned (O).

Approach	Requirement				
	Extensibility (Software)	Extensibility (Hardware)	Supporting maritime standards	Reusability	Portability
Choi et al. (2020)	(X)	O	O	(X)	O
Brushane et al. (2021)	(X)	(X)	(X)	(X)	O
Brekke et al. (2022)	(X)	O	O	(X)	(X)
Schneider et al. (2020)	(X)	O	O	X	O
Ziebold and Gewies (2017)	–	–	(X)	–	O
Leśniewski et al. (2020)	(X)	(X)	O	(X)	–
Ferreira et al. (2017)	(X)	(X)	O	(X)	–

industrial computer via Ethernet or a serial interface. In this case, the SuT is not located directly in the center of the communication infrastructure or even implemented on the main computing unit, but runs parallel to the vessels control systems. Due to this distinction between SuT and test carrier, the SuT is isolated from the vessel it is installed on. As a result of this isolation, the setup can ported to other vessels without modifications. The communication protocols themselves are not mentioned, but the described communication setup supports maritime standards.

A SuT that is not permanently mounted to the ship and could be exchanged with a different SuT is presented in the work of Leśniewski et al. (2020). An existing ship was retrofitted with an electric engine to create a hybrid propulsion system. An electric engine can be used as a substitute for the normally used combustion engine. Due to the special integration of the new electric engine into the existing power train, it would be possible to exchange the electric engine with another engine-type. Because of this, the presented architecture acts as an engine test carrier that is also capable of interfacing software and hardware SuT that interacts with the aforementioned engine. Additionally it is assumed, that it is possible to test other engine setups with the described method, reusing the test carrier.

Last, Aliaj et al. (2018) describe a test carrier setup, where the SuT is integrated through the infrastructure, and the test carriers are small unmanned surface vessels (USVs), presented by Ferreira et al. (2017). As in the approach of Aliaj et al. (2018) and Ferreira et al. (2017) describe that the test carrier have to fulfill specific requirements, which are limited in terms of the system architecture and were not focused on the test carrier. The test carrier setup, along with the system architecture, does not envision the integration of SuT on board of the test carrier. This design is limited in terms of extensibility as there is no provision for integrating additional hard- or software components on the test carrier side. Instead, these additions are restricted to the infrastructure side.

Based on the requirements derived from the introduction, the approaches can be compared. The requirements are defined as:

- Extensibility (Software): The test carrier system architecture should be extensible regarding the integration of additional software components with different middleware interfaces and protocol standards.
- Extensibility (Hardware): The test carrier system architecture should be able to be extended with additional hardware components (like sensors, actuators or SuT).
- Supporting maritime standards: The test carrier system architecture should support industry maritime standards, like the IEC 61162 (including NMEA0183 and NMEA2000) collection.
- Reusability: The test carrier system architecture should be usable for and should be capable of integrating different kinds of SuT.
- Portability: The system architecture concept should be portable on other test carriers.

Table 1 shows the comparison between the different mentioned approaches regarding the requirements fulfillment.

In summary, two insights can be drawn from the related work. First, different use cases require different placements of the SuT. This placement depends in general on the access to different sensors and actuators, as needed. Additionally, the physical placement of the SuT must be considered. There are physical SuT, such as novel sensors, software based SuT that do not require any additional hardware or other configurations. Second, for different reasons most of the mentioned related approaches use SuT specific test carrier system architectures. None of them match all main requirements related to extensibility, reusability and portability. In general, the approaches just consider the integration of one kind of SuT and are very restricted regarding extensibility. This results in the need of an adaptable test carrier architecture possibly lowering development time and effort in the future.

3. Requirements for integration of Systems under Test, sensors and actuators

When creating a reusable test carrier system architecture the identified properties like extensibility, reusability and portability are of high importance. For this the potential sensors and actuators that could be integrated must be considered. Second, the possible placements of the SuT must be determined.

3.1. Sensor and control requirements

To create a test carrier architecture that is capable of integrating new sensors, actuators and SuT into an already existing setup some precautions need to be taken. A baseline for the integration of new sensors in the maritime domain are the standards inside the IEC 61162 collection.

While the older NMEA0183 standard is based on the RS-422 standard, the newer NMEA2000 standard is based on SAE J1939 and therefore extending the CAN standard, which is commonly used in the automotive industry. The CAN standard therefore defines the data link layer, while SAE J1939 and NMEA2000 define higher layer protocols (Zeltwanger, 2012). Both are used for connecting marine instruments like sensors, actuators and display units in a standardized way. On the other hand, some manufacturers use their own proprietary communication protocols like SeaTalk NG or RayNet. The usage of NMEA2000 as well as the proprietary protocols is in general not common in industrial shipping.

Domain agnostic standards like PROFINET, Local Interconnect Network (LIN) or CANopen serve the same purpose while being more prevalent in embedded systems unrelated to the maritime domain. Even though these protocols are not very common in the maritime domain, they must be considered for the prototyping and development phase of new SuT. All these sensor networks can be accessed either by using a serial-based or an IP-based sensor and controller interface. These different layers are illustrated in Fig. 1, which shows the path of data flow from recorded information sensed by a sensor to provisioning of this information via a serial or IP-based interface encoded in an arbitrary data model.

The same principle applies to the actuators of a maritime test carrier. The actuators can be interfaced through electronic control units (ECU). While ECUs pass information between a control algorithm or the bridge of the ship and the engine itself, they can typically be interfaced through two different ways.

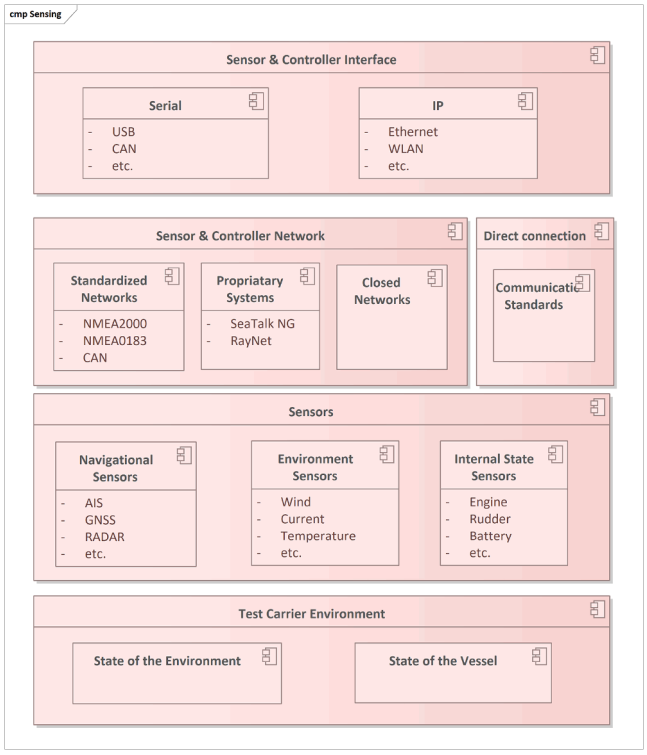

Fig. 1. Common abstraction layers that can be encountered during the integration of sensors on vessels, these layers include the environment itself, the sensors and the networks or connections to interface them, as well as the interface, which is used to receive the measured data.

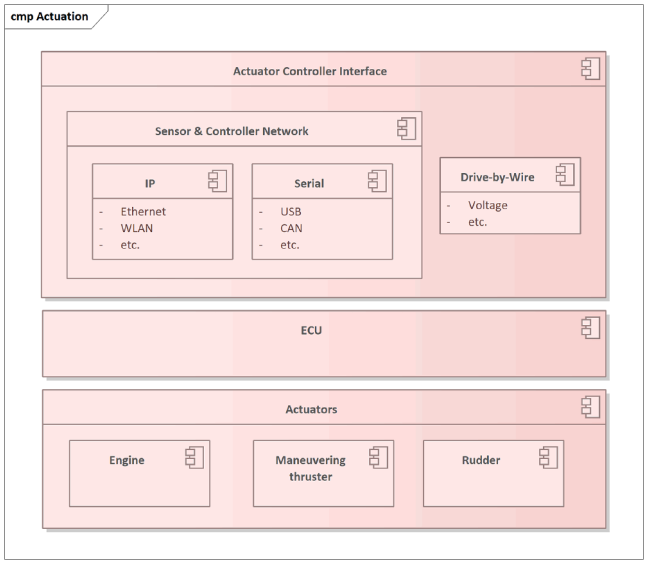

Fig. 2. Common abstraction layers that can be encountered during the integration of actuators on vessels, including the actuators, the ECU, which is used and the actuator controller interface.

First, analog signals and second CAN based communication. Furthermore, other communications are possible but are not very common. In case of CAN based communication each manufacturer in general has an own proprietary CAN implementation, which poses as a further challenge. As in case of the sensors, the actuators must be interfaced through an actuator controller interface, which is able to communicate with the actuator controller through a network or by sending Drive-by-Wire signals. This is illustrated in Fig. 2, the overall structure is similar to the sensors. To address the challenge of multitude of ways of interacting with sensors and actuators when creating a reusable test carrier architecture, the requirement of interacting with these components in a standardized way must be met, where the specific requirements of the different sensors and actuators are abstracted through specific interfaces.

3.2. Placements of Systems under Tests

When it comes to testing a SuT on a test carrier, determining the placement of the SuT is crucial. This placement defines the architectural integration point, which can be classified into several categories: ship-side and shore-side, hosted (hosting on ship-internal computing units), self-hosted (with own hardware on the ship), embedded, or non-embedded. The architectural integration point is defined as the characteristics of the integration into the test setup, based on the mentioned six characteristics. To determine the potential placements of SuT the related work can be analyzed. In general, the main goal is to place the SuT as realistic as possible on the test carrier or inside the global infrastructure and to fulfill the requirements of the SuT, which must be tested.

Based on the analysis of related work, five different placements independent of the concrete connection to the test carrier infrastructure can be determined:

1. **External SuT** is connected through the infrastructure on the shore side. The external SuT can be hosted or self-hosted inside the shore-side infrastructure.
2. **Software SuT** consists only of software, which is hosted on a vessel computing unit.
3. **Software and Hardware SuT** consist of hardware and software and must be placed on the vessel and connected to the test carrier infrastructure.
4. **Sensor SuT** is connected to sensor and controller interface onboard.
5. **Actuator SuT** is connected to a sensor and controller interface or an actuator controller interface.

A summary of these various placements with their corresponding integration points is presented in Table 2.

In addition to the individual placements, the SuT can be a System of Systems itself, consisting of multiple subsystems. In case of such System of Systems, the subsystems can be distributed across different placements inside the test carrier. For example one subsystems could be placed in the infrastructure, while the second consist of software and hardware and is placed on the test carrier itself. Therefore, distributed System of Systems SuT in different placement configurations are possible.

4. System architecture and components of the open testbed vessel

To design a test carrier system architecture, a base architecture must first be selected.

While the base vehicle architectures in the aviation and maritime domain are very similar, the automotive domain has a different base architecture. As can be seen in Oltmann (2014), maritime ship-side system architectures feature a lot of interconnected components. There are specific interfaces, but the communication must take place in a standardized way, which is in general predefined by the used integrated navigation system (INS). The INS includes most components, but does not take the distribution of data on the vessel into account as well as the support for different protocols. Similarly, within the aviation domain, all subsystems are connected through a redundant data bus, as described in Seabridge (2020). Regardless of the type of architecture, such as federated digital architecture or integrated modular architecture, all components are connected to the data bus (Seabridge, 2020). In contrast to these system architectures, the automotive domain uses the Electrical/Electronic (E/E) architecture, which centralizes the distribution of information. Specifically, the zone-oriented E/E architecture

Table 2
Placement and integration point of SuT inside the test carrier system architecture. The X shows if the integration option for the placement is possible.

Placement	Integration					
	Self-hosted	Hosted	Shore side	Ship side	Embedded	Non-embedded
External	X	X	X			X
Software		X		X		X
Software and Hardware	X			X		X
Sensor	X			X	X	
Actuator	X			X	X	

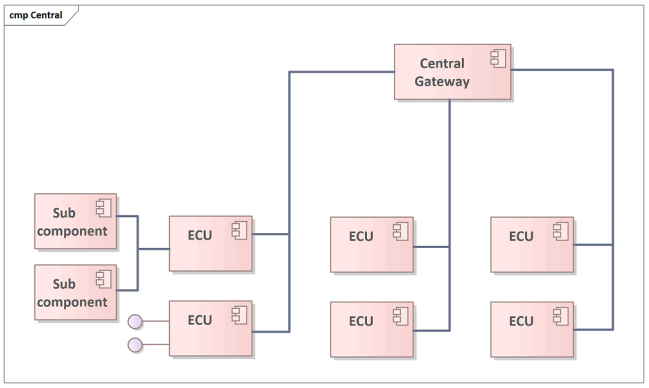

Fig. 3. Central E/E architecture (based on Brunner et al., 2017), where all ECUs are connected through several communication busses to the central gateway, which distributes all data.

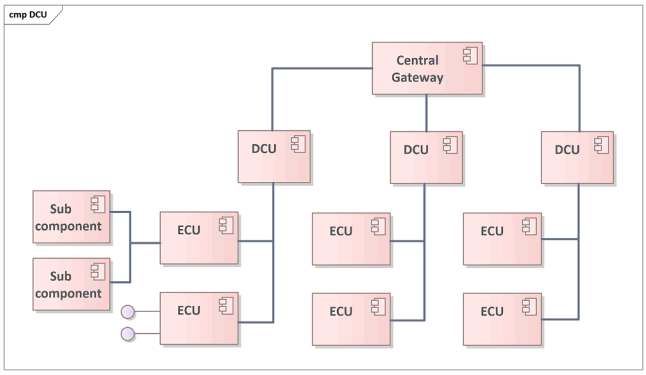

Fig. 4. Domain-oriented E/E architecture (based on Brunner et al., 2017), where domain controller units (DCU) are introduced, reducing workload of the central gateway by filtering inside the domain.

accounts for the interoperability between different protocols and the extensibility and portability of the overall system (Brunner et al., 2017). This makes the E/E architecture a suitable starting point for the development of the test carrier system architecture.

The following section presents the Electrical/Electronic (E/E) architecture, which commonly serves as a basic architecture for vehicles in the automotive domain. Subsequently, this basic architecture is adapted to meet the requirements of a maritime test carrier so that the placement of SuT and the requirements of reusability, portability and extensibility are ensured.

4.1. Centralized zone-oriented E/E architecture

The system architecture to be designed must not only support the different types of SuT with their placements, but also be generic and reusable. To achieve this, the different components of the architecture must be replaceable, while the interfaces stay the same. For example, changing the steering system onboard should be possible, ensuring the functionality and the interfaces staying the same. In addition, the architecture must be applicable to different test carriers, enabling SuT testing on different test carriers without requiring changes to the setup. This advantage can be gained using the base concept of the centralized E/E architecture, which is used in the automotive domain (Bandur et al., 2021; Brunner et al., 2017).

As described by Brunner et al. (2017) in the centralized E/E architecture several ECUs are connected to one centralized gateway (see Fig. 3) through different communication busses. In general the central gateway must handle the whole communication between the different communication busses and needs various interface options as needed for the communication (like Industrial Ethernet, CAN or other technologies as bus/communication system). This results into a very high interoperability and extensibility, but on the other side the central gateway has to provide all possible interfaces and has to handle the whole data flow. As this concept is not optimal, it was extended by introducing domains (see Fig. 4).

Domains group functions and the corresponding ECUs into one domain, which is governed by one domain controller unit (DCU). The DCU serves as gateway between the ECUs of the domain and the central gateway, filtering furthermore the information and just passing through needed information by other domains, which are then distributed by the central gateway. This reduces the workload on the central gateway and simplifies the inter-connectivity to the central gateway. The concept of domains can be further extended by cross-domains (see Fig. 5), where at least two domains are merged together and are governed by a so called cross-domain control unit (CDCU), which fulfills the same functionality as the DCU. Merging domains leads even more to reduce the workload on the central gateway. Last, the E/E architecture concept can be further developed using zones instead of domains, which describes the grouping of components based on their physical location inside the vehicle. These zones are governed by a zone controller unit (ZCU) (see Fig. 6), having a similar functionality as the DCU and CDCU. Furthermore the processing should be more centralized in one server, by shifting processing from the ECUs and ZCUs to the central server as possible. This reduces the distributed workload and centralizes the processing. The communication link between the central gateway and the specific gateway ECUs (DCU, CDCU, ZCU) is in general standardized and relies on one kind of connectivity, like Industrial Ethernet or IP based connection, while the interconnection of the other ECUs to the specific gateway ECU is established using different protocols and wiring.

So, in the zone-oriented centralized E/E architecture, the processing is moved from several different ECUs to one centralized processing unit, which is the central server. However, not all computational tasks can be transferred to the central server, so the ECUs continue to perform some of the processing. The centralization in combination with the specific gateway ECUs (DCU, CDCU, ZCU) brings the advantage of connecting typical field busses (like CAN or LIN) with newer, faster technologies such as Industrial Ethernet. This abstraction meets the requirements for abstracting sensor and actuator interfacing as mentioned in the sensor and control requirements. Moreover, the wiring is simplified and more efficient as all gateway ECUs only need to be connected to the central

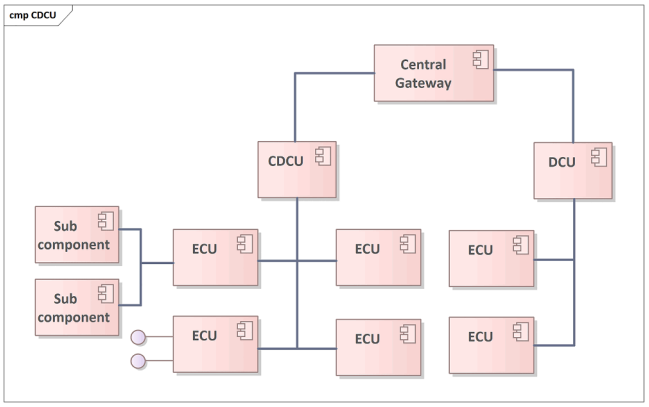

Fig. 5. Cross-domain oriented E/E architecture (based on Brunner et al., 2017), where domains with own domain controller units (DCU) are merged into cross domains which are governed by one cross-domain controller unit (CDCU) replacing the DCUs, reducing the workload from specific domains on the central gateway, while other existing domains with their DCUs are not changed.

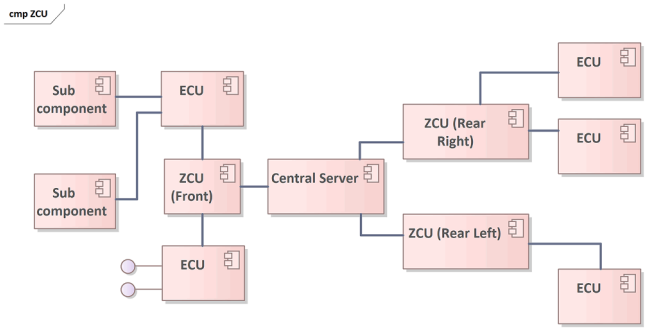

Fig. 6. Zone-oriented E/E architecture (based on Brunner et al., 2017), where the ECUs are grouped by physical zones, which are based on the installation location and independent of the functionality of the ECU, governed by zone controller units (ZCU) replacing the cross-domain controller units (CDCU) and domain controller units (DCU) and the processing is shifted to the central server, which replaces the central gateway.

server and all other ECUs to their gateway ECUs, where the data is distributed centrally as needed.

From the test carrier perspective the centralized zone-oriented E/E architecture has several advantages:

- All relevant data converge in one central server, so the server is capable of filtering and distributing data to the SuT and to the different zones.
- Adding new sensors, actuators or ECUs is simple as they just need to be integrated into an existing zone or by creating a new zone.
- The server forms the bridge between the actuator and the SuT, facilitating integration of safety functions like safety monitors, which observe incoming commands for the actuators and could block them if necessary or verify the test execution using contracts like described by Hake et al. (2024) or Torben et al. (2023).
- The centralization enables manipulation of data streams distributed to specific parts of the architecture, enabling for example the usage of testing procedures like fault injection.
- Similarly, the centralization enables the inclusion of additional data sources, such as the use of a mixed reality testing approaches by integrating simulated data into test runs.

4.2. Open testbed vessel architecture

Based on the placement of SuT and the base zone-oriented centralized E/E architecture concept, the test carrier system architecture was designed (see Fig. 7).

The vessel server is the center of the system architecture and corresponding to the E/E architecture, it is the central gateway for the infrastructure on the test carrier. Every component or subsystem in the architecture is connected directly or indirectly to it. According to the existing test carrier system architectures as presented by Brekke et al. (2022) or Brushane et al. (2021) to ensure extensibility in terms of software and support for different software solutions various middlewares should be considered during the design process. Due to the centralization of processing like described in the zone-oriented E/E architecture, the middlewares are provided and transformation of data is made on the vessel server. This results in the vessel server being capable of hosting software SuT and middleware (like ROS2, RabbitMQ, Kafka, NATS) as well as providing direct socket connections (TCP, UDP) in the manner of a communication system to distribute different data flows on the test carrier. The hosted middleware as well as different communication protocols can be used to interact with the SuT, which consist of software or software and hardware and is connected to the vessel server to receive and send data.

The data produced on the test carrier is provided by one or more vessel zone controller which are directly connected to the vessel server, corresponding to the ZCU in the E/E architecture. Each vessel zone controller manages a zone of the test carrier and is connected to at least one sensor, actuator or SuT through the interconnection as described in the sensor and control requirements and showed in Figs. 1 and 2 but can be connected to more. The vessel zone controller provides data from the sensors in its zone through the vessel server and handles control commands coming from the vessel server for actuators located in its zone. According to this, the vessel zone controller can govern actuator controller interfaces, which are connected to an actuator SuT. Through the use of vessel zone controllers the base system architecture can be extended with other sensors, controllers or functionalities, like monitoring functionalities or data recorders.

Depending on the situation onboard, the vessel server can also serve as vessel zone controller if it is connected to a sensor and controller interface or actuator controller interface. Furthermore, when sensors, actuators or embedded SuT need to be connected to a sensor and controller network, a separate network with its own sensor and controller interface must be added, to prevent violations against other parts of the test carrier infrastructure and architecture. In general, SuT should be separated from the main infrastructure onboard as much as possible, to prevent damage or violations caused by them and to have full control over all data flow from and to the SuT.

In addition to the main vessel server, also additional processing units must be planned as it is the case in integrated navigation system architectures. For example to host software SuT which have higher processing power requirements, like by using machine learning as mentioned in the introduction. This task can be performed by different vessel processing units, which are capable of processing data or providing interfaces to SuT. So, the vessel server as well as each vessel zone controller in general is or can be a vessel processing unit. Vessel processing units are possible in the architecture, based on the needs as part of a zone or directly connected to the vessel server, but they are not additionally showed to the figure.

At last, the vessel server serves as a gateway between a global infrastructure, like the global testbed infrastructure including shore stations and other participants, and the test carrier. In this case the vessel server integrates the global infrastructure like a vessel zone controller, which zone is outside the test carrier. According to this functionality, the vessel server, due to its central position in the system architecture, can serve as Ship Data Server as defined by ISO 19847 (Lee and Lee, 2023). While the Ship Data Server as defined by ISO 19847 also connects the vessel to the shore-side and is capable of storing all data, it does not include the capability of launching different middlewares and hosting SuT as the vessel server does. So the vessel server can be used also as a Ship Data Server according to ISO 19847, but extends this functionality in several ways as described.

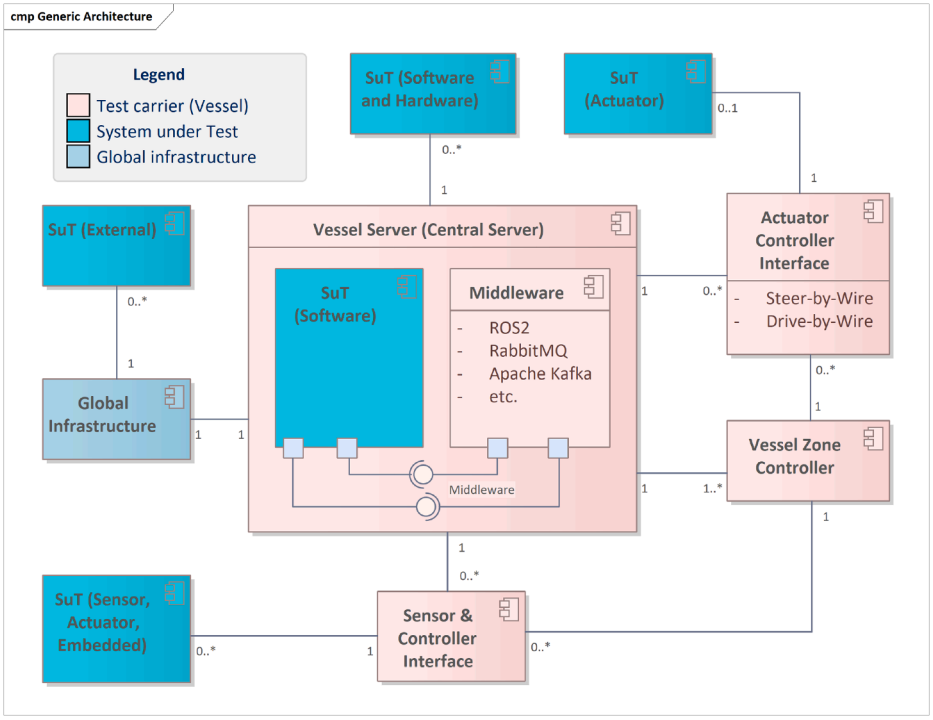

Fig. 7. Centralized zone-oriented E/E test carrier system architecture with the possible SuT placement.

Through the global infrastructure external SuT are connected to the test carrier. In particular, the vessel server, the vessel zone controller as well as the vessel processing units can be designed redundantly in order to provide higher reliability of the test carrier.

In summary the advantages of the proposed architecture are as follows:

Reusability. The proposed architecture enables reusability in cases where a change of SuT may occur, potentially resulting in the change of needed protocol. By isolating the SuT behind a zone controller or by integrating it through a specific middleware, all modifications that need to be made, are confined to the zone controller itself or by reconfiguring to provide data via the needed middleware. In contrast, no changes are necessary to other parts of the architecture, apart from adapting the zone controller or the vessel server configuration to accommodate the new SuT.

Portability. Due to the use of zone controllers, the architecture is portable and adaptable to a different vessel that may feature a distinct sensor and/or actuator setup. In addition, also the general setup of the test carries does not matter, as due to the zone controller unit the specific interface of the actuator is abstracted. Only the zone controllers that directly interact with the changed sensors and actuators require adaption to the new environment or additional zone controllers are added to accommodate the novel setup. The remaining parts of the architecture can be left untouched, such as the interface to the global infrastructure depicted in Fig. 7.

Extensibility. When a novel sensor needs to be integrated into an existing test carrier, the architecture is capable of integrating it by either adding a new zone controller or changing an existing one. Apart from processing components, no other downstream components require adaption, ensuring that the integration process remains efficient. In contrast to the existing system architectures described in the related work section, the proposed architecture does not rely on just one middleware limiting the extensibility in terms of software. Instead, by grouping components in zones where the different protocols and hardware wiring configurations are supported through the zone controller unit, as well as providing various middlewares as needed, the architecture ensures extensibility.

5. Implementation inside the eMaritime integrated reference platform

The testbed used in the evaluation is the E-Maritime Integrated Reference Platform (eMIR) consisting of virtual and physical components (Rüssmeier et al., 2019), located in the virtual testbed HAGGIS and the physical testbed LABSKAUS (Hahn, 2014). The physical testbed of eMIR contains different test carriers (Figs. 8 and 10) used in the evaluation of different SuT. In order to extend the eMIR platform the presented generic system architecture is deployed on the test carriers within the eMIR platform. Inside the physical testbed two test carriers are used. The first test carrier is the Josephine (see Fig. 8), which replaced the previously used test carrier Zuse (see Fig. 9) in 2020. The second test carrier is the Sally, which extends the physical testbed in 2022 (see Fig. 10).

This test carrier setup is designed to serve different applications. While the Josephine serves as test carrier on open sea, the Sally serves as test carrier primarily in harbors and inland waters. With these different operation domains, the equipment, like sensors, varies. Further, the drive systems of both test carriers are completely different, which results in varying interfaces. The detailed setups including the sensors and actuators of the test carriers can be found in Table 3.

To deploy the generic system architecture on both test carriers, it was slightly adjusted. Fig. 11 shows the extended system architecture based on the generic system architecture designed in Section 4. The router was omitted from the figure and connects the vessel to the infrastructure side.

The application of the generic system architecture on the eMIR test carriers has some specific additional components. First, the environment of the test carrier is limited as it has only one connection to the physical testbed through a router. The eMIR platform itself is based entirely on a central exchange bus, which is used to distribute information within the testbed.

Table 3
Technical setup of the two current test carriers within the eMIR phyiscal testbed. The communication of the sensors and actuators is mainly based on NMEA2000, while other communication protocols are mentioned in brackets.

	Josephine	Sally
Engine	Combustion	Electric
Sensors	AIS Transceiver, AIS Receiver, Anemometer, GPS, DGPS (NMEA0183), 2x RADAR (NMEA0183), Log, Sonar, Compass, Compass (NMEA0183), Rudder Indicator, Engine State Sensors	AIS Transceiver, Anemometer, DGPS (NMEA0183), GPS, RADAR (NMEA0183), Log, Sonar, Compass, 8x 1D FMCW RADAR (TCP), Battery State Indicator, 2x Rudder and Engine State Sensor (CANopen), 2x Drive Battery Array State (CANopen)
Actuation	Outboard Engine (Drive-by-Wire), Hydraulic Rudder (Steer-by-Wire)	2x Azimuth Thruster (CANopen)

Fig. 8. The test carrier Josephine.

Fig. 9. The former test carrier Zuse.

Fig. 10. The test carrier Sally.

Furthermore, within the eMIR platform, the conversion between protocols is made by the Polymorphic Interface. This Polymorphic Interface is a System of Systems, including subsystems which are capable of converting various communication protocols used in the maritime domain, such as NMEA2000, NMEA0183 and IHO S-100 to each other. In addition to the conversion it can be used to connect either a single or multiple SuT on the eMIR infrastructure side and provide data in the format required by the external SuT.

The vessel server in the eMIR application is primarily used to extend the exchange bus, which is the backbone inside the eMIR testbed and ensures that all data is available inside the whole testbed. Since the eMIR testbed uses RabbitMQ, the vessel server hosts a RabbitMQ server which is synchronized with the other RabbitMQ servers inside the global infrastructure. All data on the test carrier is transmitted to and over the exchange bus. Through the vessel server software SuT and software and hardware SuT must be connected to the exchange bus. External SuT are connected directly on the side of the infrastructure. Since these potentially cannot be connected directly to the exchange bus, it must be possible to activate various middleware adapters that provide the data via corresponding middleware (in general also hosted on the vessel server), such as ROS2, Kafka, MQTT, or similar.

Since not all messages are sent in every possible protocol, the data must be converted by the Polymorphic Interface, which is hosted on the vessel as well as inside the infrastructure. This conversion is needed to ensure the protocol support, as the protocol of the transmitted messages differs from the wiring. Using a middleware also allows to convert the messages as needed and provide them to the zone controllers in the needed protocol. These processing intensive tasks can be executed on a vessel processing unit, which could be the vessel server. In addition the Polymorphic Interface can provide the converted data to the SuT either via the exchange bus, through the middleware adapters and corresponding middleware or directly via a socket, like TCP or UDP. Using the Polymorphic Interface makes it possible for example to provide NMEA0183 data for communication with a SuT, converted from the test carrier infrastructure which could mainly consists of NMEA2000.

In case of sensor, actuator or embedded SuT, which are connected to a sensor and controller interface, the connectivity and protocol problem does not exist. To achieve independence from connectivity and protocol, the vessel zone controller must ensure that necessary data is converted and forwarded to the sensor and controller interface and data produced by the different devices is converted and forwarded to the exchange bus. To enable this functionality a network interface adapter is needed. Since the vessel server could be a vessel zone controller, it needs to have the same functionality. In addition, the vessel zone controller has the possibility to have an actuator controller interface, where an actuator SuT or a normal actuator of the vessel is connected to. Through this interface the vessel zone controller is able to convert control commands into Drive-by-Wire signals or CAN frames. In general, SuT sensor and controller networks are separated from the existing test carrier infrastructure to prevent violations and to allow an overall monitoring of the SuT in every situation, since every command must be transmitted through the exchange bus where a monitor, located in the testbed, can check the command. Such networks are managed in general by a dedicated

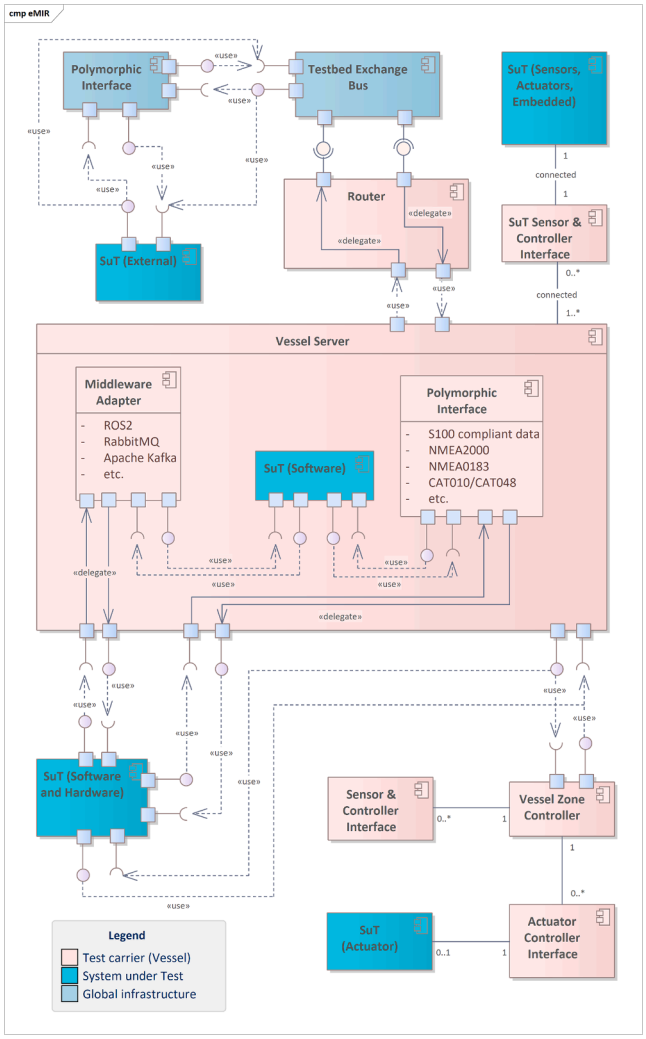

Fig. 11. Centralized zone-oriented E/E test carrier system architecture with the eMIR extension including the possible SuT placement.

vessel zone controller, since other testing functionalities could be used within these networks, like fault injection or similar. These manipulations should not be present in the other networks on the vessel.

Compared to the existing test carrier system architecture approaches from the related work the proposed test carrier system architecture has a high complexity, as it support so many different options and configurations, including wiring, protocols, middlewares, etc. This results in a very high complexity, which comes along with an increasing configuration effort. How this complexity can be handled is not part of the paper, and should be considered in the future. In addition the initial setup requires experts, as it is not trivial to equip a vessel with the presented architecture. There is the need to reorganise the hardware on board, by splitting different sensor networks and adding additional processing devices. While the initial setup can be effortful, the following integration of new devices is much easier.

6. Evaluation

To evaluate the presented system architecture, three use cases with different SuT were tested using the eMIR system architecture application during different research projects, which was implemented on the Josephine and on the Sally. In addition to the three use cases the average delay and throughput is analyzed based on the implementation. The first use case is the Maritime Traffic Alert and Collision Avoidance System (MTCAS) (Steidel and Hahn, 2019). The second use case is the evaluation of the shore-based control center architecture, which was developed within the AVATAR (POM Oost-Vlaanderen, 2024) project. The third use case was the integration of an autonomous piloting system integrated as black box, to test the functionality of the system. These use cases were performed during several projects, including the AMISIA[1] project, the AVATAR[2] project and the FuturePorts[3] project.

The test setup of the Josephine and the Sally in comparison can be found in Table 4. Both test carriers are using cellular network for the connection to the eMIR infrastructure, while the Josephine is using 4G, the Sally is using 5G. As exchange bus RabbitMQ was used. The whole setup was Linux-based using Ubuntu 20.04, 22.04 and 24.04. Since the evaluation of the connection of the 4G and 5G network should not be the focus of the paper, the effect of latency is evaluated mainly within the test carrier setup while delays from the connection to the global infrastructure only be treated briefly. As different setups other communication technologies like satellites could be used, changing the results.

In each use case first the SuT was integrated into the test carrier, based on the test carrier system architecture. After the successful integration, the verification of the functionality takes place. For the verification the SuT was tested during trials in the harbour of Emden (Josephine

[1] https://www.innovativehafentechnologien.de/wp-content/uploads/2022/01/Projektsteckbrief_AMISIA_2021-12-21_rk.pdf.
[2] https://northsearegion.eu/avatar/.
[3] https://www.dlr.de/en/research-and-transfer/projects-and-missions/futureports.

Table 4
Test setup of the test carriers.

	Josephine	Sally
Router	MOXA OnCell G3470A-LTE (4G)	Teltonika RUTX50 (5G)
Vessel Server	Industrial PC with an Intel Core i7-9700TE CPU and 32 GB DDR4 RAM	Industrial PC with an Intel Core i7-9700TE CPU and 32 GB DDR4 RAM
Vessel Zone Controller	Raspberry Pi 4 (8 GB RAM)	ODROID C4 (4 GB RAM), Laptop (Intel Core i5-5200U CPU and 8 GB DDR4 RAM), Laptop (AMD Ryzen 5850U and 32GB RAM), Industrial PC (AMD Ryzen V1605B and 16GB RAM), Industrial PC (Intel Core i9-13900TE and 32GB RAM)

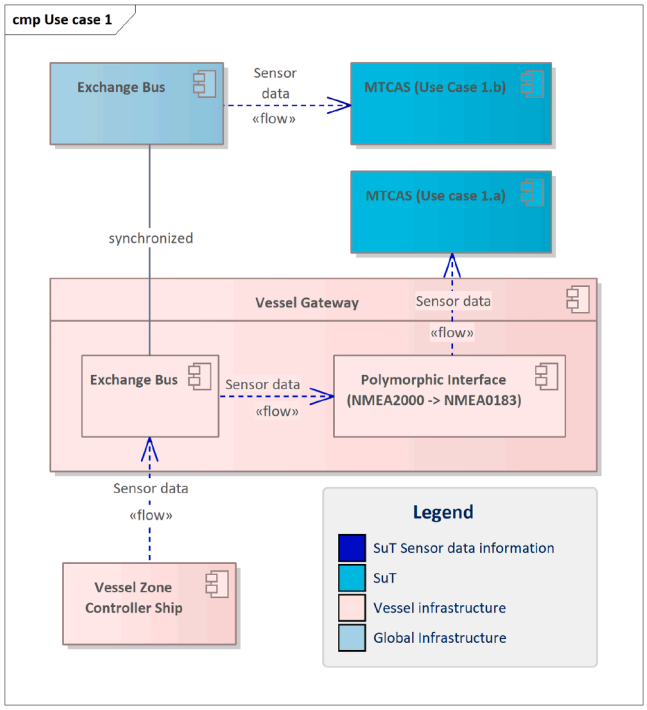

Fig. 12. The integration of the SuT MTCAS into the test carrier.

and Sally) as well as in the Jade Bight (Josephine). The specific test scenarios are described in the following use cases. In the following evaluation first the three use cases are described followed by the evaluation of the latency and throughput of the implementation on both vessels.

6.1. MTCAS

First after a extension, the Maritime Traffic Alert and Collision Avoidance System (MTCAS) (Steidel and Hahn, 2019) was tested with the Josephine and Sally. MTCAS offers functionalities especially for proactive collision avoidance and was designed to support seafarers. However, it does not perform any maneuvers itself and is a decision support assistance system. The SuT was tested in two different test setups. In the first setup (Use case 1.a, Fig. 12) the SuT was hosted on own hardware (processing and display unit) and was connected as software and hardware SuT to the Polymorphic Interface, which transforms NMEA2000 data from the exchange bus to NMEA0183 and provides the transformed data, as decision support assistance system to assist the operator onboard.

In the second setup (Use case 1.b, Fig. 12), the SuT was hosted as an external SuT within the eMIR platform and was connected to the exchange bus, where it serves as decision support assistance system for a vessel traffic service use case. The integration points can be seen in Fig. 12, the router has been omitted from the figure but connects the vessel to the testbed.

In general, the MTCAS SuT requires AIS (Automatic Identification System) data, especially the own position and other ship position reports. The AIS data is produced by an AIS transceiver, which is connected to the NMEA2000 network. This network is governed by a vessel zone controller. In the first use case the SuT expects NMEA0183 over a UDP socket. In both setups the SuT was connected to a Polymorphic Interface transforming NMEA2000 data into NMEA0183 data, which was provided via a UDP connection in the first use case. So in the first use case the SuT received the transformed data directly from the Polymorphic Interface. In the second use case setup the SuT was connected to the exchange bus, while the remaining infrastructure was the same as in the first setup. In the second use case the Polymorphic Interface provides the transformed data using the exchange bus.

Both setups were tested in the Jade Bight as well as in the Harbour of Emden. In this use case, the vessel was not controlled by a software system, but by a human captain on board. The expected result included the correct recognition of the ships, which were forwarded to the SuT using the test carrier system architecture, and to receive an evasion recommendation based on the received data. In all trials, the ships were detected immediately and evasion recommendations were made based on the received AIS targets.

6.2. AVATAR

The second use case was the evaluation of the shore-based control center architecture, which was developed within the AVATAR (POM Oost-Vlaanderen, 2024) project. The shore-based control center architecture enables a remote operator to control one or more ships from the shore-based control center. As already mentioned in Lamm et al. (2022) the Josephine and the eMIR testbed were used in the evaluation. Since the paper of Lamm et al. (2022) was published, additional trials were performed using the Sally. The SuT is a System of System consisting mainly of two subsystems, the shore-based control center, which can be integrated as external SuT and the ship side, which was realized as software SuT and was hosted on the vessel server. The integration is shown in Fig. 13, the router has been omitted from the figure but connects the vessel to the testbed.

This distributed SuT needs at least all information, which are available onboard, including sensors like the log, AIS, RADAR, RPM, engine temperature and similar, which are all available through the NMEA2000 network and NMEA0183 devices on the Josephine or the NMEA2000 network, CANopen network and NMEA0183 devices on the Sally. This information is provided through the vessel zone controller or the vessel server depending on the used test carrier to the exchange bus. For the simplicity of the presentation, the vessel zone controller is shown separately, even if it can also be the vessel server.

The needed sensor readings were then provided through the exchange bus, as it was already described in the MTCAS use case. All the sensor readings were used inside the shore-based control center (the external SuT), which receives the data from the exchange bus inside the global infrastructure. With the transmitted data, the shore-based control

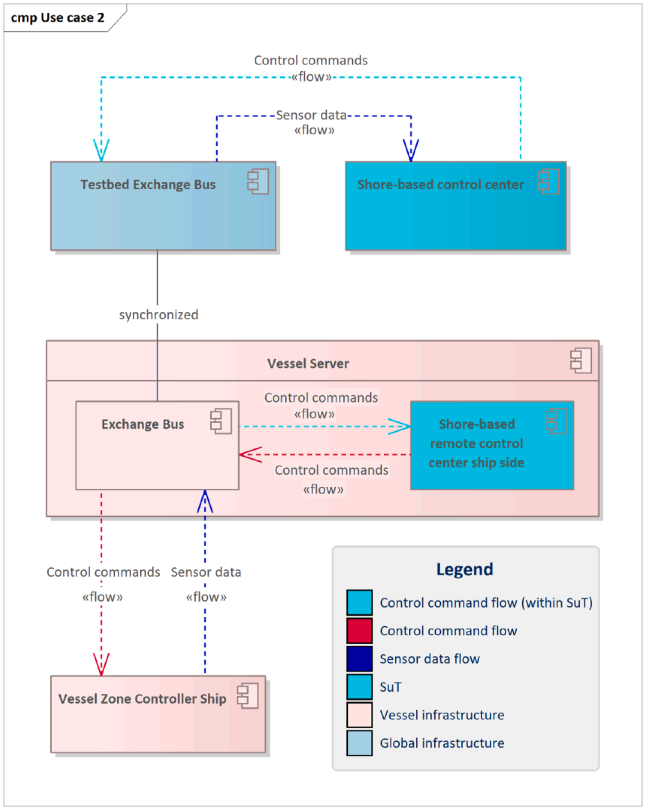

Fig. 13. The integration of the SuT shore-based control center into the test carrier.

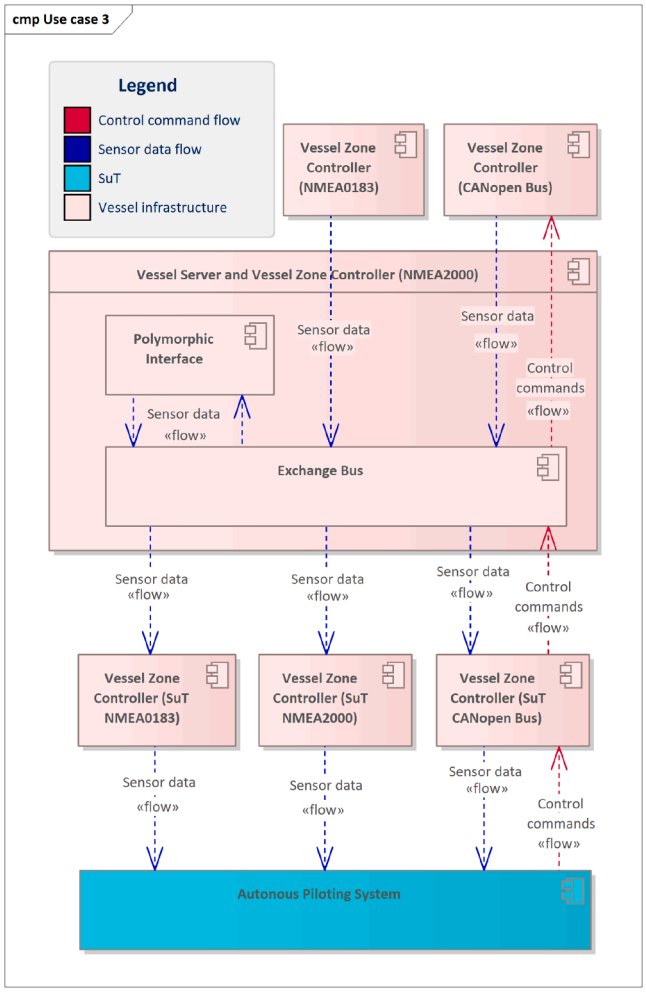

Fig. 14. The integration of the SuT autonomous piloting system into the test carrier.

center was supplied with information, which were then used to get the situational picture from the ship side.

In contrast to the MTCAS use case, the shore-based control center also interacts with vessel actuators. To enable the remote control, control commands were transmitted between the ship side component and the shore-based control center (within the SuT). In this case the control commands are sent using the exchange bus. Since the shore-based control center uses RabbitMQ for communication, like the testbed, no adapters had to be used to enable communication between ship and shore. After control commands have arrived, the ship side component must transform the incoming control commands into the format, which is compatible with the ship vessel zone controller. For this purpose, a IHO S-100 based command model was used. The messages were transmitted through the exchange bus to be transformed by the vessel zone controller into a Drive-by-Wire and Steer-by-Wire signal on the Josephine or a CANopen Frame on the Sally.

Equivalent to the MTCAS use case both setups were tested in the Jade Bight as well as in the Harbour of Emden. In all trials the remote operator should drive through the harbour or inside the Jade Bight. The expected result included the correct data transmission from the ship to the control center on shore and in the opposite the right transmission of control commands from the infrastructure side into the ship. In addition, the expectation was to use the control on both test carriers, so the interface for the engine should be the same. Also in all of these trials, all the data was transmitted correctly and the control center receives all needed information to provide situational awareness to remote control the ship from the shore-side. In addition the control data transmission was successful in both setups and the engines were controlled correctly as expected.

6.3. Autonomous piloting system

The third use case was the evaluation and integration of an autonomous piloting system, consisting of hardware and software as black box system, which was integrated on the Sally. The autonomous piloting system is capable of following preplanned tracks by ensuring the safety of the operation regarding collision avoidance. This SuT uses standardized connections using three different interfaces: CANopen, NMEA2000 and NMEA0183. This results in the need to add three new zones, which provides the SuT NMEA2000 network, the SuT CANopen network and the SuT NMEA0183 communication. In addition three vessel zone controller units are needed which handle the communication from the vessel server to the specific networks and communication links of the SuT. The integration is shown in Fig. 14. In this figure the global infrastructure is omitted from the figure, as the SuT was just tested locally.

The SuT processes RADAR and AIS data from the NMEA0183 input, all other relevant navigational and environmental data is consumed through NMEA2000. At last the SuT processes incoming engine sensor data from the engine CANopen network and produces control commands if it is operating the ship. The data streams are mainly passed through the exchange bus on the vessel server, which is at the same time the vessel zone controller for the ships NMEA2000 network. The AIS data is transformed using the Polymorphic Interface from NMEA2000 into NMEA0183 and provided through the exchange bus to the SuT NMEA0183 vessel zone controller and then to the SuT.

The use case was tested in the Harbour of Emden. In all trials the system was operated autonomously, following a preplanned mission. The expected result included the correct recognition of the needed data, like the ship network data, the AIS and the RADAR, which is received through the new zones. In addition the SuT should not notice it is isolated. Last, the system architecture should enable the test engineer to interrupt the control of the SuT.

In all trials, the data was transmitted correctly, as the system recognizes existing ships and adapts the internal measurements. In addition, simulated ships were injected and the SuT also takes them into account during the builtin collision avoidance. The execution of the SuT was also interrupted by a reconfiguration of the data streams within the vessel server. All in all, also these trials were performed successfully, achieving the expected results.

6.4. Latency and throughput evaluation

Using the described setup (Table 4) an latency and throughput evaluation similar to Hossain et al. (2017) was performed.

As baseline for the evaluation the smallest default update rate from NMEA2000, CANopen, NMEA0183 and the 1D Radar sensors was used. The smallest default update rate for NMEA2000 is 100 milliseconds, the same as for the 1D radar sensors. The NMEA0183 standard has a default update rate of one second. In addition the CANopen protocol, which is used on board of the Sally was evaluated, the smallest update rate of this proprietary protocol is 50 milliseconds and must be applied on the communication on board. Therefore the evaluation is made mainly against the update rate of 100 milliseconds for the external communication, assuming the infrastructure should be capable of processing a message until the next message of this type arrives.

For the evaluation all components were synced with one time server and the time was measured on the test carrier and how fast data is available on the exchange bus on the vessel, so it could be received by a vessel zone controller unit. In addition a second evaluation was made taking also into account the connection to the infrastructure and the availability inside the infrastructure. Different configurations providing NMEA2000, CANopen, 1D Radars and NMEA0183 were recorded over a specified time period. In every configuration, every message is first timestamped at the moment of its occurrence in the vessel zone controller. A second timestamp is added once the message is processed by the RabbitMQ server. The last timestamp is added during the processing on the destination component.

Transmission times were measured in various different configurations, during 15 trials. Each vessel zone controller represents in general one or more sensors in his zone (like NMEA2000 network, CANopen network, NMEA0183 sensors, 1D FMCW RADAR sensors). In order to evaluate the effect of a more extensive configuration, in addition the throughput was evaluated, as the effect of more vessel zone controllers as well as sensors is low, this is not considered in detail. Configuration between 100 up to 1300 messages per second were tested and the results were similar, with outliers reaching up to 4 milliseconds. To analyze the delay times in detail, in Figs. 15 and 16 two representative scenarios are shown. In the first scenario, the delay and transmission time is analyzed just on the test carrier, in the second scenario the delay and processing time is analyzed coming from the SuT through a RabbitMQ in the infrastructure and then received by the destination component inside the infrastructure.

As shown in both scenarios presented in Figs. 15 and 16 the delay and processing times fulfill the requirement of a overall delay of less than 100 milliseconds from data acquisition to the data provision. Further, the delay and processing time on the test carrier as can be seen in Fig. 15 is under 30 milliseconds, so also the requirement from the CANopen protocol can be fulfilled. In general the processing time and the delivery by the RabbitMQ servers seems to be most time consuming. Considering also the mobile network delay the transmission time between the source and the RabbitMQ server as well from the RabbitMQ

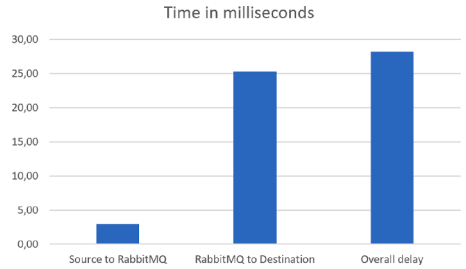

Fig. 15. Average delay and processing time of all messages on the test carrier excluding the connection to the infrastructure, including the time from the data source (ECU/Sensor) through the ZCU to the RabbitMQ, the time from the RabbitMQ to the destination ZCU and the overall time results from the sum of the two other timestamps.

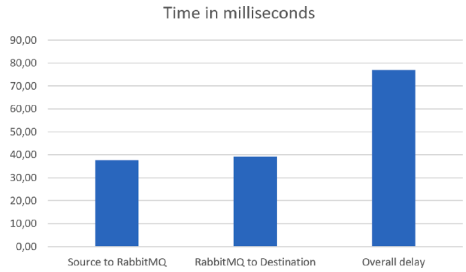

Fig. 16. Average delay and processing time of all messages containing the connection to the infrastructure, including the time from the data source (ECU/Sensor) on the test carrier through the ZCU to the RabbitMQ on the shore side, the time from the RabbitMQ on the shore side to the destination ZCU in the infrastructure and the overall time results from the sum of the two other timestamps.

server to the destination is nearly the same but overall as mentioned less than 100 milliseconds as can be seen in Fig. 16. This results in the fact, that the included delays do not influence the SuT in their processing procedures, as all of them worked as expected with the provided data, as already discussed in the three presented use cases. Nevertheless, the delay introduced by the middleware and hardware should be as low as possible, therefore the performance of different middleware and hardware combinations should be evaluated and adapted to the use case. As shown by the evaluation the chosen middleware RabbitMQ fulfills the requirements as needed by the setup.

Because of the impact of hardware as well as the chosen middleware, the combination of the two plays a significant role in terms of limitations in latency and throughput of the proposed architecture. As this paper presents the architecture and provides a proof of concept, it does not provide an exhaustive list of combinations of hardware and middleware combinations and therefore is not capable to provide a final assessment of the latency and throughput limitations.

7. Conclusion

Summarizing, the presented concept shows a generic reusable test carrier system architecture, which can be used on different test carriers, to make the installation, integration and setup process for testing easier in the maritime domain.

Regarding the integration of SuT, different placements were derived from the related work and how they can be integrated into such a test carrier system architecture without changing the system architecture or the test setup. This shows the extensibility of the concept, since the vessel server works as gateway between the vessel zone controllers, the SuT and the global infrastructure. Based on this, new sensors, actuators or other systems can be integrated without changing the system architecture. As the architecture is adaptable and communication protocol agnostic, it is prepared to deal with technologies that are currently in early development stages but are promising as a basis for future software

or hardware systems. One of these new technologies, that are currently in a testing phase where a test carrier with the proposed architecture might be helpful is Artificial Intelligence based collision avoidance algorithms, as already mentioned in the introduction. In addition, the system architecture enables testing autonomous driving functions, where the different components of the autonomous system can be integrated at the different integration places or connected to different networks and sensors, as shown in the third use case. Due to the portability and extensibility as shown through the usage on the Josephine and Sally, which are equipped with different sensor and actuator setups, the development of assistance systems as well as highly automated or autonomous systems can be simplified and accelerated.

In addition, through the abstraction, which is provided by using the E/E architecture and the vessel zone controller, just the vessel zone controller must be adapted to be compatible with the vessel server, while all other ECUs, components and subsystems are not changed, like shown for example through the integration of the CANopen network on the Sally or the easy integration of the different SuT in the three use cases. In general, the usage of the E/E architecture provides advantages, like abstraction of different protocols and communication layers, which makes integration easier, even on test carriers. In general, no adjustments are necessary and the setup can be used in this form for various tests and trials. The evaluation shows the possibility of integrating different SuT, based on three use cases while the concept itself is independent of the use case. The integration of the SuT was made without any changes in the overall setup and system architecture. This shows the reusability of the concept, in opposite of other existing test carriers and their system architectures, which are designed in general for one specific use case. The requirements extensibility, portability and reusability were fulfilled as described, furthermore through the usage of NMEA2000 and NMEA0183 the usage and support for maritime standards is fulfilled.

All in all, the presented system architecture fulfills all identified requirements in opposite to the existing approaches. Further, the prototypical implementation shows the usability of the concept.

Summarizing the developed generic reusable and extensible system architecture fulfills the requirements for a maritime test carrier system architecture, which can be used to test different SuT by providing the functionality to extend the base setup of the test carrier.

CRediT authorship contribution statement

Janusz Andrzej Piotrowski: Writing – review & editing, Writing – original draft, Software, Conceptualization; **Christian Steger:** Writing – review & editing, Writing – original draft, Software, Conceptualization; **Axel Hahn:** Writing – review & editing, Supervision

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Aliaj, E., Dimaki, G., Getsopoulos, P., Thomas, Y., Fotiou, N., Toumpis, S., Koutsopoulos, I., Siris, V., Polyzos, G.C., 2018. A platform for wireless maritime networking experimentation. In: Presented at the 2018 Global Information Infrastructure and Networking Symposium (GIIS). IEEE, Thessaloniki, Greece, pp. 1–6. https://doi.org/10.1109/GIIS.2018.8635782

Bandur, V., Selim, G., Pantelic, V., Lawford, M., 2021. Making the case for centralized automotive E/E architectures. IEEE Trans. Veh. Technol. 70, 1230–1245. https://doi.org/10.1109/TVT.2021.3054934

Brekke, E.F., Eide, E., Eriksen, B.-O.H., Wilthil, E.F., Breivik, M., Skjellaug, E., Helgesen, Ø.K., Lekkas, A.M., Martinsen, A.B., Thyri, E.H., Torben, T., Veitch, E., Alsos, O.A., Johansen, T.A., 2022. milliAmpere: an autonomous ferry prototype. J. Phys.: Conf. Ser. 2311. https://doi.org/10.1088/1742-6596/2311/1/012029

Brunner, S., Roder, J., Kucera, M., Waas, T., 2017. Automotive E/E-architecture enhancements by usage of ethernet TSN. In: Presented at the 2017 13th Workshop on Intelligent Solutions in Embedded Systems (WISES) IEEE, Hamburg, Germany, pp. 9–13. https://doi.org/10.1109/WISES.2017.7986925

Brushane, F., Jämsä, K., Lafond, S., Lilius, J., 2021. A experimental research platform for maritime automation and autonomous surface ship applications. IFAC-PapersOnLine 54, 390–394. https://doi.org/10.1016/j.ifacol.2021.10.121

Chae, C.-J., Kim, M., Kim, H.-J., 2020. A study on identification of development status of MASS technologies and directions of improvement. Appl. Sci. 10, 4564. https://doi.org/10.3390/app10134564

Choi, J., Park, J., Jung, J., Lee, Y., Choi, H.-T., 2020. Development of an autonomous surface vehicle and performance evaluation of autonomous navigation technologies. Int. J. Control Autom. Syst. 18, 535–545. https://doi.org/10.1007/s12555-019-{\protect\penalty-\@M}0686-0

Ferreira, B., Coelho, A., Lopes, M., Matos, A., Goncalves, C., Kandasamy, S., Campos, R., Barbosa, J., 2017. Flexible unmanned surface vehicles enabling future internet experimentally-driven research. In: OCEANS 2017—Aberdeen. Presented at the OCEANS 2017—Aberdeen. IEEE, Aberdeen, United Kingdom. https://doi.org/10.1109/OCEANSE.2017.8084934

Hahn, A., 2014. Test bed for safety assessment of new e-navigation systems. Int. J. e-Navig. Marit. Econ. 1, 14–28. https://doi.org/10.1016/j.enavi.2014.12.003

Hake, G., Reiher, D., Mentjes, J., Hahn, A., 2024. Integrating scenario- and contract-based verification for automated vessels. J. Mar. Sci. Technol. https://doi.org/10.1007/s00773-024-01008-0

Hossain, M., Noor, S., Karim, Y., Hasan, R., 2017. IoTbed: a generic architecture for testbed as a service for internet of things-based systems. In: Presented at the 2017 IEEE International Congress on Internet of Things (ICIOT). IEEE, Honolulu, HI, USA, pp. 42–49. https://doi.org/10.1109/IEEE.ICIOT.2017.14

IALA, 2022. G1107–Planning and Reporting of Testbeds in the Maritime Domain.

IEEE, 1990. IEEE Standard Glossary of Software Engineering Terminology. https://doi.org/10.1109/IEEESTD.1990.101064.

ISO, 2005. ISO 17894:2005—Ships and marine technology—computer applications—General principles for the development and use of programmable electronic systems in marine applications.

Kang, J., Ryu, D., Baik, J., 2021. Predicting just-in-time software defects to reduce post-release quality costs in the maritime industry. Softw. Pract. Exper. 51, 748–771. https://doi.org/10.1002/spe.2927

Lamm, A., Piotrowski, J., Hahn, A., 2022. Shore based control center architecture for teleoperation of highly automated inland waterway vessels in urban environments. In: Proceedings of the 19th International Conference on Informatics in Control, Automation and Robotics. Presented at the 19th International Conference on Informatics in Control, Automation and Robotics. SCITEPRESS - Science and Technology Publications, Lisbon, Portugal, pp. 17–28. https://doi.org/10.5220/0011266000003271

Lee, C., Lee, S., 2023. Overcoming the DDoS attack vulnerability of an ISO 19847 shipboard data server. JMSE 11, 1000. https://doi.org/10.3390/jmse11051000

Leśniewski, W., Piątek, D., Marszałkowski, K.M., Litwin, W., 2020. Small vessel with inboard engine retrofitting concepts; real boat tests, laboratory hybrid drive tests and theoretical studies. Energies 13, 2586. https://doi.org/10.3390/en13102586

Liu, J., Yang, F., Li, S., Lv, Y., Hu, X., 2024. Testing and evaluation for intelligent navigation of ships: current status, possible solutions, and challenges. Ocean Eng. 295, 116969. https://doi.org/10.1016/j.oceaneng.2024.116969

Oltmann, J.-H., 2014. E-navigation—Ein Überblick. In: Hydrographische Nachrichten 98. Rostock: Deutsche Hydrographische Gesellschaft e.V., pp. 14–19.

POM Oost-Vlaanderen, 2024. AVATAR. https://northsearegion.eu/avatar/ (Accessed August 20, 2024).

Rüssmeier, N., Lamm, A., Hahn, A., 2019. A generic testbed for simulation and physical-based testing of maritime cyber-physical system of systems. J. Phys. Conf. Ser. 1357, 12025 (2019,10). https://doi.org/10.1088/1742-6596/1357/1/012025

Schneider, V.E., Delea, C., Oeffner, J., Sarpong, B., Burmeister, H.-C., Jahn, C., 2020. Robotic service concepts for the port of tomorrow: developed via a small-scale demonstration testbed. In: Presented at the 2020 European Navigation Conference (ENC). IEEE, Dresden, Germany, pp. 1–8. https://doi.org/10.23919/ENC48637.2020

Seabridge, A.G., 2020. Design and Development of Aircraft Systems. 3rd edition. Aerospace Series. Wiley, Hoboken, N.J., USA.

Steidel, M., Hahn, A., 2019. MTCAS—an assistance system for collision avoidance at sea. In: 18th International Conference on Computer and IT Applications in the Maritime Industries, Tullamore, 25-27 March 2019 Hamburg, Technische Universität, HamburgHarburg 978-3-89220-709-2, pp. 261–273.

Torben, T.R., Smogeli, Ø., Glomsrud, J.A., Utne, I.B., Sørensen, A. J.S., 2023. Towards contract-based verification for autonomous vessels. Ocean Eng. 270, 113685. https://doi.org/10.1016/j.oceaneng.2023.113685

Wang, C., Zhang, X., Gao, H., Bashir, M., Li, H., Yang, Z., 2024. COLERGs—constrained safe reinforcement learning for realising MASS's risk-informed collision avoidance decision making. Knowledge-Based Syst. 300, 112205. https://doi.org/10.1016/j.knosys.2024.112205

Zeltwanger, H., 2012. Standardized higher-layer protocols for different purposes. In: Presented at the international CAN Conference, CAN in Automation e.V. Neustadt an der Weinstraße.

Zhang, X., Wang, C., Jiang, L., An, L., Yang, R., 2021. Collision-avoidance navigation systems for maritime autonomous surface ships: a state of the art survey. Ocean Eng. 235, 109380. https://doi.org/10.1016/j.oceaneng.2021.109380

Ziebold, R., Gewies, S., 2017. Long term validation of high precision RTK positioning onboard a ferry vessel using the MGBAS in the research port of rostock. TransNav 11, 433–440. https://doi.org/10.12716/1001.11.03.06