

**24th Conference on
Computer and IT Applications in the Maritime Industries**

COMPIT'26

Tullamore, 12-14 October 2026

**24th International Conference on
Computer and IT Applications in the Maritime Industries**

COMPIT'26

Tullamore, 12-14 October 2026

Edited by Volker Bertram

24th International Conference on Computer and IT Applications in the Maritime Industries, Tullamore, 12-14 October 2026

Download from: www.compit.info or <https://zenodo.org/communities/compit/>

Sponsored by



tutech.de



<https://simuneer.de>

Media Partners



<https://www.rina.org.uk/>

Volker Bertram <i>25 Years of COMPIT: Visions and Reality of Maritime IT Solutions</i>	7
Tracy Plowman, Volker Bertram <i>AI in Maritime Training: Lessons Learned from Successes and Failures</i>	15
Hazel Mitrik, Tom DeNucci, Volker Bertram <i>A State-Of-The-Art Report on Simulating Ships in Arctic Waters</i>	24
Anna Wagner, Mirko Theiss, Johannes Lützenberger, Carsten Zerbst <i>Enabling Graph-Based Knowledge Links for the Shipbuilding Industry</i>	42
Torben-Henning Woelke, Manuel Ratte, Peter Wittkop, Carsten Zerbst <i>A Capability-Based Blueprint for Shipbuilding PLM</i>	53
Yasuo Ichinose, Kouki Wakita <i>Generative AI Implementations in Ship Design at OCEANS: From Task Automation to Process Transformation</i>	64
Sven Albert, Romain Hottois, Pierre Weyne <i>Advanced CFD Simulations for a Wave-Energy Powered Ship Propulsion System</i>	77
Francesco Oneto <i>Seamless Structural Design Workflow</i>	83
Mladen Radonić, Jan Obić <i>Field Evaluation of the OCX-Based XR Inspection Tool for Ship Structures</i>	93
Diego De Leon, Herbert Koelman, Sam Otjens <i>Automatic Adaptation of Ship Subdivision: An Exploration on Methanol Conversion</i>	104
Helge Bjordal <i>Borderless Virtual Training Simulators in Maritime Industry</i>	114
Thor Albrektsen, Endre Sandvik, Øyvind Rabliås, Sverre Steen <i>Data-Driven Prediction of Residual Resistance of Platform Supply Vessels</i>	125
Chenwei Gui, Kazuhiro Aoyama <i>Toward a Trustworthy Shipyard Digital Twin: Spatio-Temporal Grounding for LLM Surveillance Video Understanding</i>	138
Herbert Koelman, Dennis Korpel <i>Implementing CAD Data Standards in Application Software: Challenges and Solutions</i>	151
Ahmad Ostovar, Shahan Khan, Vivek Ramaswamy <i>From Legacy 2D Drawings to OCX: Automated Structural Knowledge Graph Construction for Model-Based Ship Approval</i>	165
Constantin-Alexander Gresens, Carsten Zerbst <i>AI-Based CAD Model Comparison for Heterogeneous Environments</i>	178

Daniel van den Berg, Piero Favero <i>Algorithmic Optimization of Space Allocation in Initial Design of Ships under Assigned Functional, Performance and Regulatory Requirements</i>	189
Jussi Puurula, Juha Peippo, Michael Pudd, Hans Van den Wijngaert <i>Simulation Driven Approval</i>	206
Heikki Hansen, Nimra Khan, Ole Hympehdahl, Karsten Hochkirch <i>From Undisturbed Wind to Onboard Flow: Estimating WAPS Potential through CFD of Ship-Induced Flow Fields</i>	222
Henrique M. Gaspar <i>A Single Source of Truth Needs Shared Meaning: Ship Data Ontology in the Age of AI</i>	234
List of authors	
Flyer for next conference	

25 Years of COMPIT: Visions and Reality of Maritime IT Solutions

Volker Bertram, VB Conferences, Hamburg/Germany, volker@vb-conferences.com

Abstract

This paper looks at the first COMPIT conference in the year 2000, and the current state of the art in advanced maritime IT solutions. The paper highlights the visionary character of the first COMPIT conference(s), where key developments of the subsequent quarter of a century were already addressed and predicted: Virtual Reality, 3D modeling in early ship design, autonomous ships, robotics, advanced simulations as decision support, and Artificial Intelligence. Many of the visions of the year 2000 have by now become reality, but also some of the thorny issues have remained with us such as interface issues.

1. Introduction

The COMPIT story starts two years before the 1st COMPIT conference. I had had the good fortune to spend a sabbatical in Japan, which had widened my scientific horizon significantly. Among others, I spent hours discussing with Dr Kasai, the director of the Japanese national research program for intelligent ships, learning much about Artificial Intelligence and autonomous ship technology.

In my youthful naivety, I thought that my enthusiasm for these new technologies would be shared by many and I organized the AIOMA Workshop on Artificial Intelligence and Optimization for Marine Applications in Hamburg in 1998. Eight people responded to my call for paper, making the workshop a financial and personal disaster, but it brought great contacts and a learning lesson: The focus of a conference needs to be right; not too narrow, then you don't get critical mass; not too wide, then the participants do not connect and the disciplines don't cross-fertilize.

With this in mind, the scope of themes was widened for the next conference. No flyer of the original COMPIT has survived but the list of topics was likely to be nearly identical to the 2nd COMPIT, Fig.1. The new conference was named "COMPIT", as it signaled computers and IT (information technology), but also could be pronounced as "compete", which felt right in an increasingly competitive industry where IT-savviness was seen as a recipe to survive and thrive. This time, the mix of themes was right. 107 participants from 23 countries were testaments that the COMPIT formula worked.

2nd International EuroConference on Computer Applications and Information Technology in the Maritime Industries

**COMPIT'2002
Rome, Italy, 13.-16.5.2002**

Topics: Computer Aided Design / Computer Aided Manufacturing / Robotics
 Virtual Prototyping / Virtual Reality
 Computer Integrated Manufacturing / Computer Aided Planning
 Information Technology Standards/ Electronic Data Exchange/ Net Technology
 Management Information Systems / Executive Information Systems
 Artificial Intelligence
 Management/Legal/Economical Aspects of Information Technology
In Shipbuilding, Offshore Engineering, and Ship Operation

Fig.1: COMPIT topics in the beginning as taken from the 2nd COMPIT conference flyer

In the following, I will discuss some of these topics and how they evolved over a quarter of a century. I omitted on purpose some other "usual suspect" topics at COMPIT, such as optimization, data formats and taxonomies, and applications to ship production.

2. Key topics – Developments of a quarter of a century

2.1. 3D Ship Modelling

In the year 2000, ship design was still predominantly based on 2D plans and 2D modelling. But we saw already to beginnings of this approach being challenged. *Aarnio (2000)*, Fig.2, argued in the first COMPIT conference in favor of 3D models to be used already in conceptual design, seeing them as the backbone for assorted simulations and Virtual Reality applications, and enabling straightforward 3D modelling in detailed design. Product data would be attached to the 3D model, and updated over time, forming a product model. The fundamental vision was spot-on and has come a lot closer to reality over the years, even if most shipyards have not yet fully implemented it.

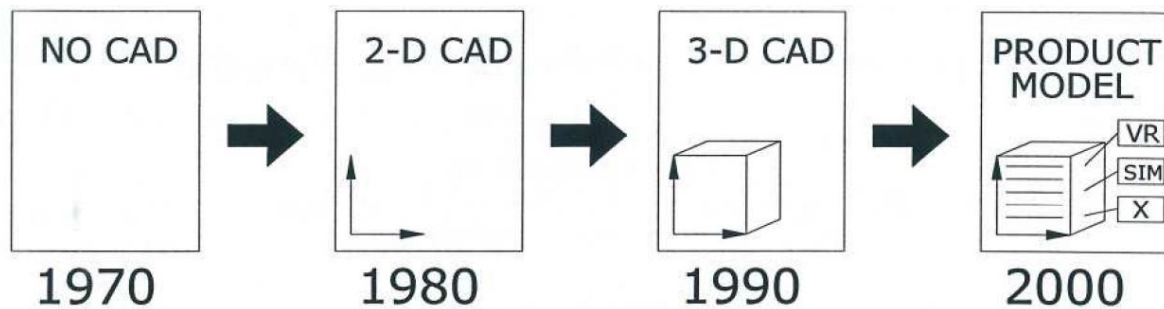


Fig.2: Ship design modelling progressing over time, *Aarnio (2000)*

The discussions about 2D vs. 3D lasted for a long time. 2D modelling, after all, was much faster and cheaper, and thus so tempting, especially for the early conceptual design phase. Proponents of 3D modelling argued, however, that 2D plans could be quickly generated from 3D models, while the switching from 2D to 3D required completely new re-modelling from scratch with the time and cost for the 2D modelling being “wasted”. The rift between the two camps may at least be partially closed by recent developments where 3D models are largely automatically generated from 2D plans, using the power of Artificial Intelligence (AI), *Ha and Gaspar (2025)*.

More recently, the progress in power and affordability of 3D scanning has opened the door to 3D modelling of existing ship structures, both for the outside hull geometry, Fig.3, *Paranhos (2000)*, *Bertram and Harries (2023)*, and the interior, such as engine rooms, Fig.4, *Blom and Czapla (2021)*, *Blom and Stonestreet (2022)*. Alternatively, 3D hull geometries may be recreated for ships in service using parametric modelling, additional information from general arrangement plans and stability booklets, and experience for similar ships, e.g. for trim optimization, *Bertram and Harries (2023)*.



Fig.3: 3D underwater scan of ship hull



Fig.4: 3D scan directly integrated in CAD

2.2. From Simulation Models to Digital Twins

Some of the first maritime computer applications concerned simulations, where the rise of container ships in the 1970s and 1980s were enabled by the rise in Finite Element Analysis (FEA). Increasing computing power and progress in algorithms and processes lead to a proliferation and wide adoption of FEA after the year 2000. Computational Fluid Dynamics (CFD) came up later, maybe two decades after FEA, but had reached wide adoption as an engineering tool in industry practice by 2010, *Peric and Bertram (2011)*. By the early 2000s, a plethora of simulation techniques were used in the maritime industry: structural analysis, hydro- and aerodynamic analyses, vibration & acoustic analyses, traffic flow simulations (for evacuation), dynamics stability, fire and smoke propagation, etc., *Fach (2006)*, *Bertram and Kakalis (2014)*, and growing computing power and progress in algorithms have led to increasingly realistic, ‘high-fidelity’ simulations.

The term ‘Digital Twin’ was coined in 2010 by John Vickers at NASA. Strictly speaking, the Digital Twin is a replica of its physical twin (an asset like a ship) and should evolve and change in time as its physical counterpart and influence the physical counterpart. “In simple terms, the vision was some IT model with the look and feel of the real deal. Not only would it look like its physical twin (this would be mere Computer-Generated Imagery or Virtual Reality), but it would behave like its physical twin and evolve in time like it. The Digital Twin of a ship would lose strength in time as it rusts, slow down as the hull gets fouled, vibrate according to the propeller and engine excitation, etc.”, *Bertram (2021a)*. Lesser forms of modelling are then mere Digital Shadows or simulation models, Fig.5. The definitions are muddy or philosophical quibbling, *Cabos and Rostock (2018)*, and our industry has happily adopted this, that or another term for the modelling of an asset and its physical behavior. The models are often classified as ‘white-box’ (based on first principles) or ‘black-box’ (data-driven) models, but in reality they are all shades of grey, with first principles using empirical coefficients and data-driven model being most successful if they are physics-informed, thus mixing in human insight into the physics (a.k.a. first principles).

More recently, meta-models using AI trained on first-principles simulations have evolved as an accurate and very efficient way to mimic the physics of ships, e.g. *Boogard et al (2022)*, *Ahmed et al. (2023)*, *Wheeler et al (2025)*.

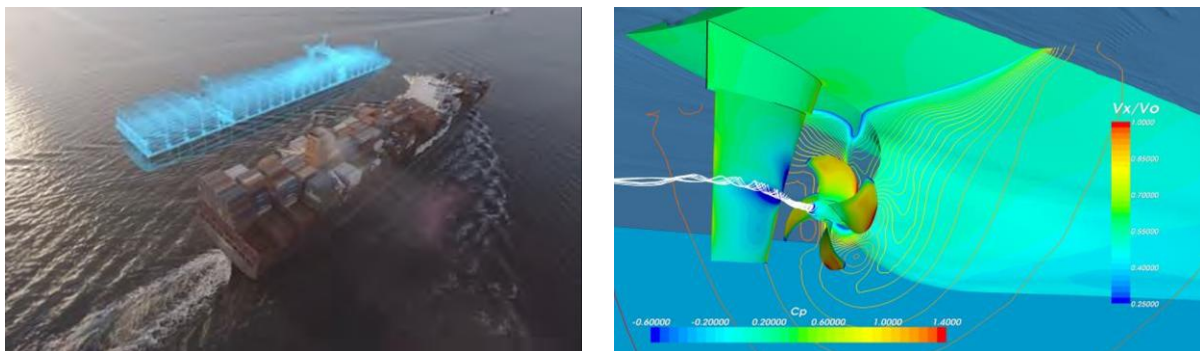


Fig.5: Digital Twin vision (left) and typical hydrodynamic simulation model (right)

2.3. Artificial Intelligence (AI)

In the year 2000, AI was largely terra incognita for the maritime world, except for some very visionary and even in hindsight stunning developments like *Söding's (1977)* Chwarismi compiler, then dubbed machine intelligence. But the first two COMPIT conferences opened the door with several AI papers, on knowledge-based systems, *Bertram (2000)*, machine learning, Fig.6, *Mesbahi and Atlar (2000)*, *Mesbahi and Bertram (2000)*, machine vision, *Vincze et al. (2000)*, natural language processing, *Wauchope et al. (2003)*, and robotics, *Armada et al. (2003)*.

In the two decades to follow, we have seen many more applications in these fields, with only knowledge-base systems fading out from the research activities within a decade, *Bertram (2018)*. The progress was rather evolutionary with new applications rather than fundamentally new approaches.

But then ChatGPT entered the stage and we saw a quantum leap in interest and applications for LLMs (Large Language Models). While my own first experience with LLMs rather revealed the dangers of hallucinations, *Gaspar and Bertram (2023)*, the usefulness of LLMs and also small language models for dedicated maritime applications is undeniable, e.g. *Matsuo (2025)*, *Wheeler et al. (2025)*, Fig.7. And we see the industry moving to the next level, agentic AI, where generative AI for large frameworks of tasks is combined automatically with generative AI for the individual tasks.

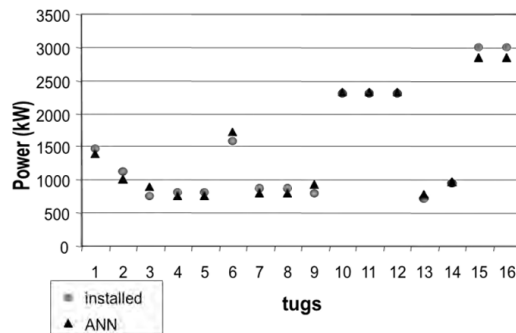


Fig.6: Machine learning application in 2000

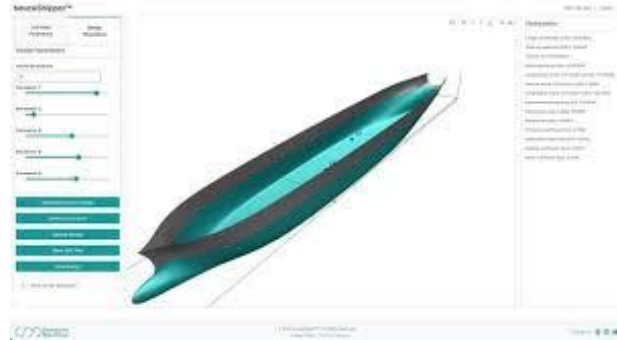


Fig.7: [NeuralShipper](#) uses generative AI

2.4. Autonomous Ship Technology & Robotics

The vision of unmanned ships part already at the 2nd COMPIT, *Bertram (2003)*. Reality lags behind the famous vision of Oskar Levander in his [interview](#) with the Financial Times: “... by 2030 autonomous ships will be a common sight on the oceans.” While timelines and expectations for unmanned ships have sobered down significantly, we have seen a multitude of developments of autonomous systems for ships, e.g. for collision avoidance.

And autonomous shipping is a good case for ‘think big, start small’. Robotics, particularly autonomous robots, have been a constant factor in the COMPT conferences, show-casing autonomous navigation, machine vision, cooperative robotics, human-robotic interfacing, etc., e.g. *Lamiroy et al. (2000)*, *Vincze et al. (2000)*, *Veers and Bertram (2005)*, *Bibuli et al. (2014,2017)*, *Dane et al. (2025)*.

Besides autonomous navigation of surface drones, another field where industry adoption of robotics in shipping has enjoyed exponential growths is hull management, namely in-water inspection and cleaning of hulls, *Bertram (2021b)*.

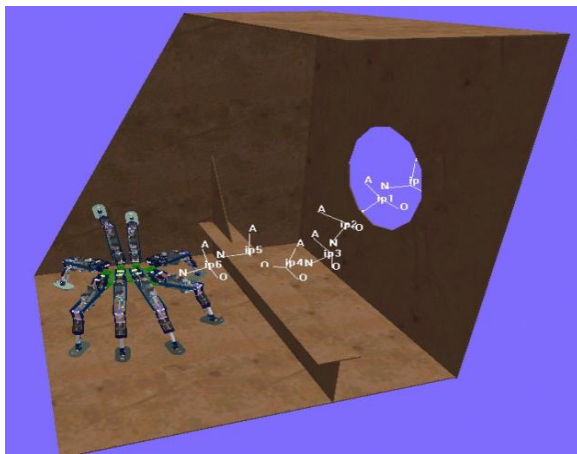


Fig.8: Machine vision robot, *Vincze et al. (2000)*



Fig.9: Autonomous drones, *Dane et al. (2025)*

2.5. Virtual Reality

Beier (2000) described already in the first COMPIT conference how Virtual Reality (VR) is practically used in industry: “The term Virtual Reality initially referred to [...] fully] immersive systems. With time, the meaning of VR broadened and, as of today, VR is also being used for semi-immersive systems, [...] even non-immersive systems, like monitor-based viewing of three-dimensional objects.” While the visuals of early VR systems were crude by today’s standards and expectations, the functionality and usefulness of early systems was impressive, e.g. VR combined with routing and voice operation in *Wauchope et al. (2003)*.

Over time, Virtual Reality applications presented at COMPIT have become increasingly sophisticated both in terms of visuals and underlying functionality, Fig.10, e.g. *Kil et al. (2019)*, *Bertram et al. (2020)*, *Goh (2022)*, *Bjordal (2025)*.

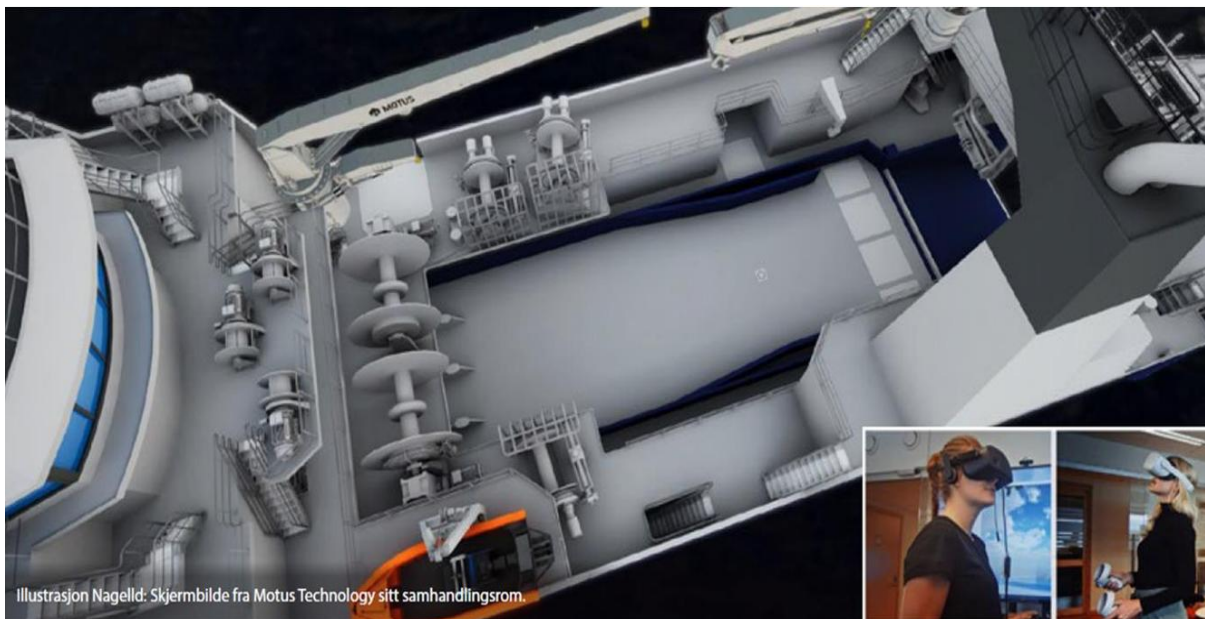


Fig.10: Design evaluation in VR, *Bjordal (2025)*

3. The road goes on forever

To use a line from a country & western song, the road (of advanced IT applications in the maritime industries) goes on forever and the party never ends. Or so one might be tempted to think.

Quite likely, some traditional problems will stay with us, such as interface problems between the many native formats used in heterogeneous software architectures that are so typical for our industry.

Computing power continues to rise, e.g. through 3D chips, dedicated AI accelerators and Graphics Processing Units, and quantum computing. This opens new options, both in simulations and AI applications.

But a new generation of smart and technology-savvy researchers gives hope that the problems will be solved in time and the new opportunities used to make current vision come reality.

Acknowledgements

COMPIT has been an inspiration over the years, often motivating me to write up my thoughts and experience, diving deeper into assorted themes. I am grateful to many of my colleagues who knowingly or unknowingly contributed to this inspiration.

References

- AARNIO, M. (2000), *Early 3-D Ship Model Enables New Design Principles and Simulations*, 1st COMPIT Conf., Potsdam, pp.5-17, http://data.hiper-conf.info/compit2000_potsdam.pdf
- AHMED, O.; HARRIES, S.; LOHSE, J.; SVEN-ERIC SALECKER, S.E. (2023), *Parametric Modeling, CFD Simulations, DoE and Machine Learning for the Design of a Planing Boat*, 22nd COMPIT Conf., Drübeck, pp.176-192, http://data.hiper-conf.info/compit2023_drubeck.pdf
- ARMADA, M.; GONZALEZ DESANTOS, P.; AKINFIEV, T.; FERNANDEZ, R. (2003), *Underwater Climbing Robot for Ship Cleaning*, 2nd COMPIT Conf., Hamburg, pp.516-524, http://data.hiper-conf.info/compit2003_hamburg.pdf
- BEIER, K.P. (2000), *Web-Based Virtual Reality in Design and Manufacturing Applications*, 1st COMPIT Conf., Potsdam, pp.45-55, http://data.hiper-conf.info/compit2000_potsdam.pdf
- BERTRAM, V. (2000), *Knowledge-Based Systems for Ship Design and Ship Operation*, 1st COMPIT Conf., Potsdam, pp.63-71, http://data.hiper-conf.info/compit2000_potsdam.pdf
- BERTRAM, V. (2003), *Cyber-Ships – Science Fiction and Reality*, 2nd COMPIT Conf., Hamburg, pp.336-349, http://data.hiper-conf.info/compit2003_hamburg.pdf
- BERTRAM, V. (2018), *Demystify Artificial Intelligence for Maritime Applications*, 17th COMPIT Conf., Pavone, pp.22-35, http://data.hiper-conf.info/compit2018_pavone.pdf
- BERTRAM, V. (2021a), *Visions Far Ahead of Reality*, 20th COMPIT Conf., Mülheim, pp.7-15, http://data.hiper-conf.info/compit2021_muelheim.pdf
- BERTRAM, V. (2021b), *Robotic Hull Cleaning – Past, Present and Prospects*, 20th COMPIT Conf., Mülheim, pp.16-23, http://data.hiper-conf.info/compit2021_muelheim.pdf
- BERTRAM, V.; HARRIES, S. (2023), *3D Mapping of Underwater Hulls for Robotic Guidance*, 22nd COMPIT Conf., Drübeck, pp.234-240, http://data.hiper-conf.info/compit2023_drubeck.pdf
- BERTRAM, V.; KAKALIS, N. (2014), *Simulation-based design and optimization for navy vessels and megayachts*, 9th HIPER Symp., Athens, pp.17-27, http://data.hiper-conf.info/Hiper2014_Athens.pdf
- BERTRAM, V.; PLOWMAN, T.; FEINER, P. (2020), *Survey Simulator – A Virtual Reality Training Tool for Ship Surveys*, 19th COMPIT Conf., Pontignano, pp.7-17, http://data.hiper-conf.info/compit2020_pontignano.pdf
- BIBULI, M.; BRUZZONE, G.; CACCIA, M.; ZEREIK, E.; BRUZZONE, G.; GIACOPELLI, M.; SPIRANDELLI, E. (2014), *Final Results and Exploitation of the Cooperative Autonomous Robotic Towing System for Emergency Ship Towing Operations Support*, 13th COMPIT Conf., Redworth, pp.181-192, http://data.hiper-conf.info/compit2014_redworth.pdf
- BIBULI, M.; BRUZZONE, G.; CACCIA, M.; CHIARELLA, D.; FERRETTI, R.; ODETTI, A.; RANIERI, A.; ZEREIK, E. (2017), *Cutting-Edge Underwater Robotics - CADDY Project Challenges, Results and Future Steps*, 16th COMPIT Conf., Cardiff, pp.101-114, http://data.hiper-conf.info/compit2017_cardiff.pdf
- BJORDAL, H. (2025), *Digital Twin combined with XR in Design, Training & Operation*, 23rd COMPIT Conf., Pontignano, pp.117-128, http://data.hiper-conf.info/compit2025_pontignano.pdf

- BLOM, M.; CZAPLA, M. (2021), *Direct Integration of 3D Laser Scanning in CAD*, 20th COMPIT Conf., Mülheim, pp.24-29, http://data.hiper-conf.info/compit2021_muelheim.pdf
- BLOM, M.; STONESTREET, G. (2022), *Exploiting Digital Point Clouds in Design E3D*, 21st COMPIT Conf., Pontignano, pp.382-386, http://data.hiper-conf.info/compit2022_pontignano.pdf
- BOOGAARD, M.v.d.; ALESSI, G.; MALLOL, B.; WUNSCH, D.; CLERO, N.; AMADORI, D.; HIRSCH, C.; ZAMPIERI, L.; HILDEBRANDT, T. (2022), *Accelerating Marine Propeller Development in Early Design Stages Using Machine Learning*, 21st COMPIT Conf., Pontignano, pp.7-15, http://data.hiper-conf.info/compit2022_pontignano.pdf
- CABOS, C.; ROSTOCK, C. (2018), *Digital Model or Digital Twin?*, 17th COMPIT Conf., Pavone, pp.403-411, http://data.hiper-conf.info/compit2018_pavone.pdf
- DANE, R.; ROZENUERS, N.; MILLINER, I. (2025), *Autonomy x 2 – Sustainably Powered Smart Unmanned Surface Vessels*, 23rd COMPIT Conf., Pontignano, pp.6-11, http://data.hiper-conf.info/compit2025_pontignano.pdf
- FACH, K. (2006), *Advanced Simulation in the Work of a Modern Classification Society*, 5th COMPIT Conf., Oegstgeest, pp.34-44, http://data.hiper-conf.info/compit2006_oegstgeest.pdf
- GASPAR, H.; BERTRAM, V. (2023), *The Use of ChatGPT and Similar AI in Marine Engineering: Limitations and Opportunities*, 22nd COMPIT Conf., Drübeck, pp.73-83, http://data.hiper-conf.info/compit2023_druebeck.pdf
- GOH, K. (2022), *Application of Virtual Reality Tools in Ship Design*, 14th HIPER Conf., Cortona, pp.81-89, http://data.hiper-conf.info/Hiper2022_Cortona.pdf
- HA, J.; GASPAR, H. (2025), *Two-Stage Conversion of GA Drawings into 3D Model Using Deep Learning*, 23rd COMPIT Conf., Pontignano, pp.228-237, http://data.hiper-conf.info/compit2025_pontignano.pdf
- KIL, W.S.; BYUN, S.; LEE, J.Y.; SON, M.J. (2019), *Development of Real-Time Emergency Response Training Simulator for Collective Ship Crews Based on Virtual and Mixed Reality*, 18th COMPIT Conf., Tullamore, pp.180-190, http://data.hiper-conf.info/compit2019_tullamore.pdf
- LAMIROY, B.; DRUMMOND, T.; HORAUD, R.; KNUDSEN-NECKELMANN, O. (2000), *Visually Guided Robots for Ship Building*, 1st COMPIT Conf., Potsdam, pp.262-276, http://data.hiper-conf.info/compit2000_potsdam.pdf
- MATSUO, K. (2025), *Constructing Design Information Exchange Relations in Shipbuilding Using Large Language Models*, 23rd COMPIT Conf., Pontignano, pp.150-157, http://data.hiper-conf.info/compit2025_pontignano.pdf
- MESBAHI, E.; ATLAR, M. (2000), *Artificial Neural Networks: Applications in Marine Design and Modelling*, 1st COMPIT Conf., Potsdam, pp.276-291, http://data.hiper-conf.info/compit2000_potsdam.pdf
- MESBAHI, E.; BERTRAM, V. (2000), *Empirical Design Formulae using Artificial Neural Nets*, 1st COMPIT Conf., Potsdam, pp.292-301, http://data.hiper-conf.info/compit2000_potsdam.pdf
- PARANHOS, P.M. (2020), *3D Laser Inspection of Ship Hulls*, 1st PortPIC Conf., Hamburg, pp.14-22, http://data.hullpic.info/PortPIC2020_Hamburg.pdf

PERIC, M.; BERTRAM, V. (2011), *Trends in industry applications of CFD for maritime flows*, 10th COMPIT Conf., Berlin, pp.8-18, http://data.hiper-conf.info/compit2011_berlin.pdf

SÖDING, H. (1977), *Ship design and construction programs (2)*, New Ships, 22(8):272

VEERS, J., BERTRAM, V. (2006), *Development of the USV Multi-Mission Surface Vehicle III*, 5th COMPIT Conf., Oegstgeest, pp.345-355, http://data.hiper-conf.info/compit2006_oegstgeest.pdf

VINCZE, M.; AYROMLOU, M.; GALT, S.; GASTERATOS, A.; GRAMKOW, C.; HEWER, N.; HOFFGAARD, S.; MADSEN, O.; MARTINOTTI, R.; NECKELMANN, O.; SANDINI, G.; WATERTMAN, R.; ZILLICH, M. (2000), *RobVision – Visually Guiding a Walking Robot through a Ship Structure*, 1st COMPIT Conf., Potsdam, pp.477-490, http://data.hiper-conf.info/compit2000_potsdam.pdf

WAUCHOPE, K.; EVERETT, S.; TATE, D.; MANEY, T. (2003) *Speech-Interactive Virtual Environments for Ship Familiarization*, 2nd COMPIT Conf., Hamburg, pp.70-83, http://data.hiper-conf.info/compit2003_hamburg.pdf

WHEELER, M.; PONKRATOV, D.; ARENS, E.; HODGES, J.; KHAN, S.; AWAN, J. (2025), *Integrating AI into CFD simulations for Next Generation Ship Design*, 27th NuTTS Symp., Zagreb, <https://cloudtowingtank.com/27th-numerical-towing-tank-symposium-nutts-2025/>

AI in Maritime Training: Lessons Learned from Successes and Failures

Tracy Plowman, DNV, Hamburg/Germany, tracy.plowman@dnv.com
Volker Bertram, VB Conferences, Hamburg/Germany, volker@vb-conferences.com

Abstract

We investigated various AI-based tools advocated for various functions of training – text generating tools, image generating tools, and training material. The focus of our test cases was on maritime applications. The results for specialized maritime training material applications were soberingly poor, but results for general text and image applications sometimes very good. In summary, AI has the potential to accelerate and improve development of training material, but should not be tasked with complete training development for highly specialized areas as frequently found in maritime training.

1. Introduction

Artificial Intelligence (AI) is en vogue. Even if the term is often used with only a sketchy understanding of the technology. “40% of AI start-ups don’t use AI”, *Collé and Morobé (2022)*. AI is widely perceived as revolutionizing our world, and impacting all industry segments, including the maritime industries and the training world.

Machine learning is the most prominent AI application, and frequently “using AI” means “using machine learning” these days. Machine learning is also at the core of machine vision and natural language processing, both often seen as AI branches in their own right, *Bertram (2018)*. Machine learning builds models and produces results based on empirical knowledge. If underlying data bases are sparse or biased, or if the software attempts extrapolation beyond the scope of the data base, errors occur. Resulting dangers from uninformed use of AI in maritime applications were already pointed out by *Bertram (2018)*, and *Collé and Morobé (2022)*.

Gaspar and Bertram (2023), *Gaspar et al. (2023)* tested various AI-based tools for maritime applications, with the conclusion that the tools gave rather disappointing results. The experience since then with AI-based tools supporting assorted tasks in developing maritime training courses has led to a more differentiated view. The conclusion and advice of using AI may then be summarized as: Be aware of limitations, but embrace the possibilities of evolving AI technologies – also in maritime training. AI offers opportunities to develop better training material in shorter time. We will illustrate that in concrete cases in the following.

2. Tools for text applications

AI may help in scientific writing, especially for non-native speakers. For automatic translation, DeepL and Reverso have produced good results in our experience for general texts, and reasonable results for technical translations, where occasionally specific marine, engineering, or mathematical terms were not correctly translated. Fig.1 shows an example of such a translation from a German text on numerical ship hydrodynamics, *Bertram (1994)*, to an English equivalent. The AI-based automatic translation saves significant time in re-using texts from native languages for wider, international publication.

The performance of tools like DeepL is similar to that of human translators who have a good command of languages for common vocabulary, but lack the specific vocabulary of a narrow technical domain. Even if resulting texts need some final editing, the automatic translation saves significant time and eliminates grammatical and spelling mistakes. The highly technical terms are generally the easy parts for non-native speakers with an engineering background.

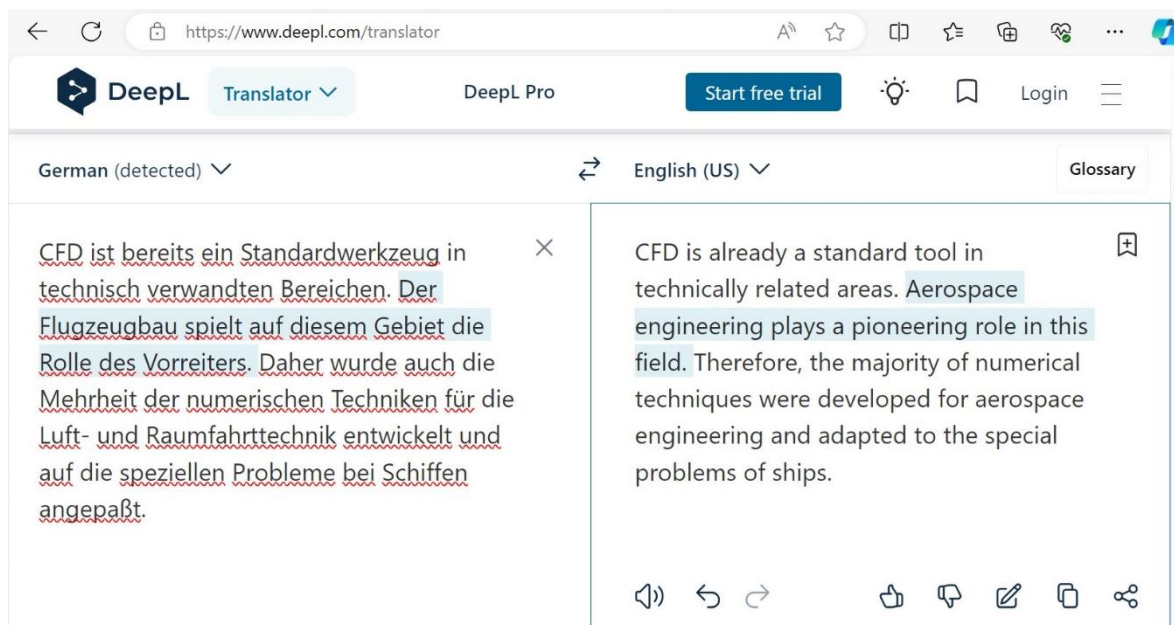


Fig.1: DeepL translation example – German to English

DeepL is limited to translations from/to 26 languages, including British and American English. [ChatGPT](#), now embedded in [Microsoft Copilot](#), may be used for another “special” type of translation, namely that from stilted writing into easy language, from cumbersome jargon to plain English. This is useful not only for foreigners, but also for native speakers, as their wider range of vocabulary and idiomatic expressions may introduce inadvertently linguistic hurdles for a non-native audience. Again, often some human post-processing is needed, e.g., in judging with context information what jargon is useful and what harmful for the intended audience. Microsoft Copilot is also, in our experience, useful for summarizing longer texts or even PowerPoint presentations, e.g. into a concise abstract.

ChatGPT is also good for bringing structure into larger texts. *Gaspar and Bertram (2023)* describe testing it on drafting a chapter structure for a book and a maritime paper, finding the tool very efficient in itemizing and synthesizing information. ChatGPT’s suggestions were largely valid and worth considering. As a positive surprise, ChatGPT suggested some topics that the human authors had not considered, especially when asked for more information, i.e., when asking ChatGPT to “Explain”.

Sometimes, it may be useful to convert images to text, e.g., when scanning older texts that are only available in analogue form, as in books or conference proceedings typically dating from pre-2000. Here, the power of pattern recognition of AI can be harnessed effectively and efficiently. Examples of online tools we have used for such purposes are [JPG to Text](#) and [OCR Online](#).

As ghost-writer for technical texts with maritime topics, ChatGPT did not write usable text and did not create usable lists of references when given just a few prompts or keywords. The texts produced needed so much reworking that any time savings as compared to writing yourself disappeared, *Gaspar and Bertram (2023)*, *Gaspar et al. (2023)*. In addition, Copilot tends towards hallucinations, making up wrong quotes and adding non-existing references. However, Large Language Models (LLMs) like Copilot may be used when tightly supervised. We got good results by prompting for a storyline structure first, editing that, then prompting for a storyline structure for each chapter, editing that, and then prompting for text paragraph by paragraph, editing the text slightly and checking the references provided by Copilot. This approach resulted probably in some time savings and a good text.

3. Tools for image and video creation

While ChatGPT deals with text, Dall-E, now also integrated in Microsoft Copilot and assorted free online text-to-image generators, generates images based on text prompts. The marketing examples

shown are impressive. For example, when it came on the market, Dall-E advertised that you can prompt the tool to paint a specified object in the style of Vincent Van Gogh, Fig.2. However, one should always test software on one's own applications, *Bertram and Couser (2010)*. We tested Dall-E 2 prompting for a ship stability curve (aka GZ curve), *Gaspar et al. (2023)*. The tool responded very quickly with a sine curve, plotted over x rather than heel angle. After 20 minutes of chat [by the ever so patient Henrique Gaspar], trying to prompt Dall-E 2 into plotting a GZ curve, it still did not reach an acceptable result; see Fig.3 for the final attempt.

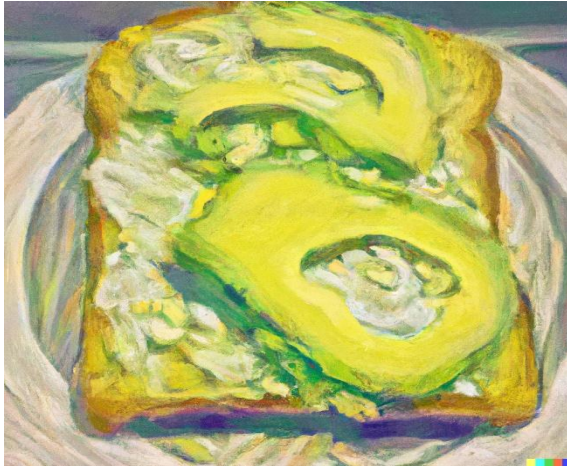


Fig.2: Dall-E 2 response to prompts for Van Gogh style avocado toast

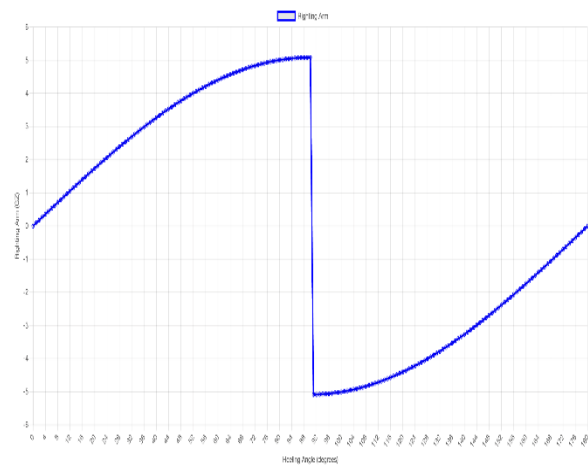


Fig.3: Failed attempt to obtain a GZ curve

Trying to get Dall-E 2 to draw a general arrangement plan for a frigate failed similarly, *Gaspar et al. (2023)*. The results were either a child-like simplistic drawing lacking significantly in detail, or a very artistic perspective drawing that would have been useful for some fantasy novel illustration, but not for practical purposes in our industry, Fig.4. Every once in a while, we have tried again, asking for technical drawings or engineering diagrams, but found at least the free online AI text-to-image tools for naval architectural applications wanting.

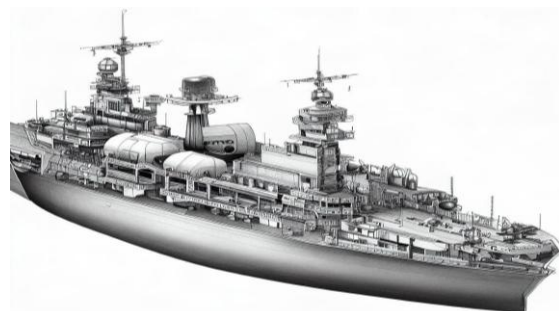
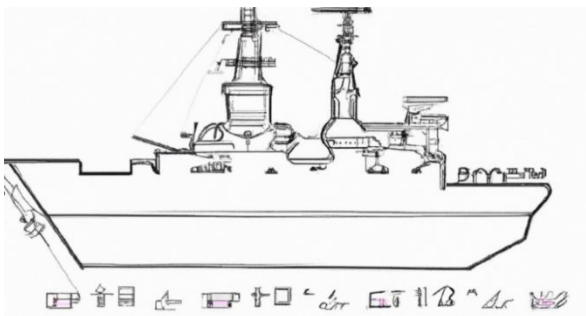


Fig.4: Two attempts of Dall-E to draw a general arrangement plan of a frigate

However, we have found Microsoft Copilot quite useful for more generic maritime images with good, photorealistic results. Fig.5 (left) shows a result for the prompt “worker hand holding a valve in a factory environment”. Fig.5 (right) shows a result for the prompt “small Caterpillar engine inside an engine room of a small fishery vessel. Show also the other equipment of an engine room in the background. Place an engineer who is working on the Caterpillar engine in front of the engine.” The results for these test cases were very directly used in training material, where neither searches in in-house and commercial image databases, nor google image search for open-source images resulted in success. There are some limitations to the tools.

The AI software has been trained on filtered data sets. For privacy reasons, images of celebrities were not used. For example, “show a captain on a bridge who looks like King Charles” will not work.

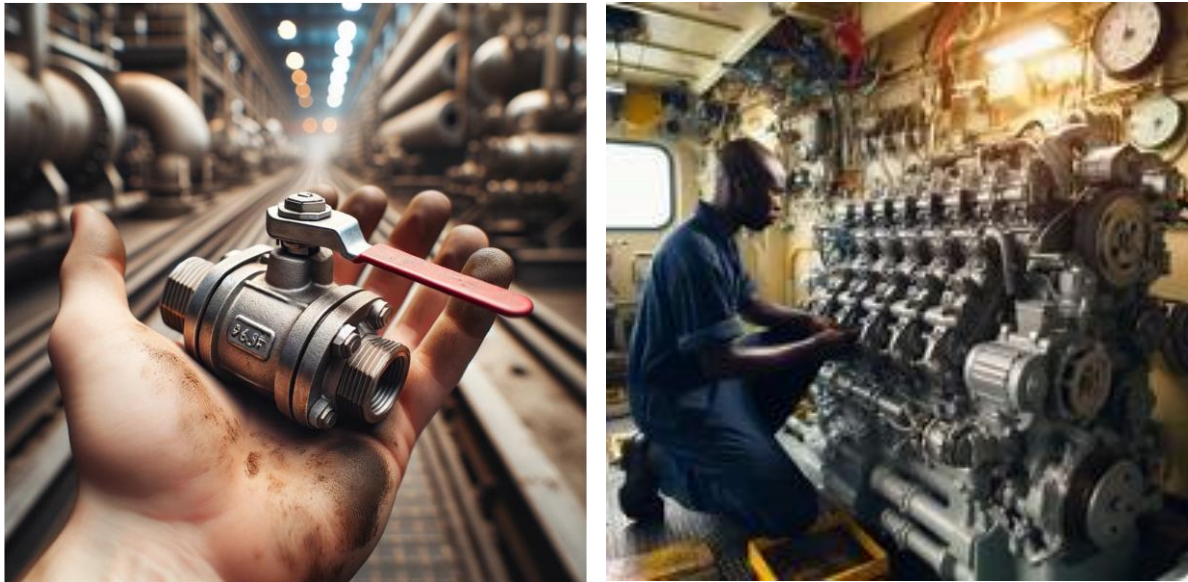


Fig.5: Dall-E 3 results for valve in worker hand (left) and diesel engine on fishery vessel (right)

More annoying for our line of work is however the “dyslexic handicap” of the software. Requests to include specific words in an image often result in garbled text. For example, we requested a photo-realistic image of a truck with ‘ammonia’ written on it in front of a ship in port. Fig.6 shows two results. While truck, ship, and port satisfied our expectations, the word ‘ammonia’ got garbled in different ways (Amnoina and Aminonak), and we gave eventually up to get an image with correct spelling.



Fig.6: Copilot results for truck with ‘ammonia’ in front of a ship; misspelling from software

In summary, Copilot can produce tailored images to illustrate training material on generic maritime topics, without copyright issues. It is in some respect more useful than image databases like iStock or Shutterstock. Finding images to visually support a theme on PowerPoint slides or in e-learning is a frequent task in training development, and harnessing the power of AI for this task offers definite improvements.

There has been similarly impressive progress on “moving pictures”, i.e., video creation using AI. State-of-the-art tools create storylines and dialogues from some prompts, and generate videos with lip-synced speakers. The tools are user-friendly and easy to use for beginners. “I keyed in some [DNV Maritime] Academy related words and it was pretty quick. It automatically made a starting theme with landmark

Dubai buildings in the backdrop of the boat. Then I selected the conversation scene. [...] I don't think it took more than 10 minutes including me experimenting and figuring out the edits,” Michelle Kurian (DNV Maritime Academy in Dubai). Some editing is needed typically for dialogues and details, similar to repeating prompting for images to tailor the product to your needs. 1980s [Vyond](#) cartoon-style videos, Fig.7, are evolving to photorealistic, tailored avatars. The image for the avatar in Fig.8 was created using [MidJourney](#). The avatar and its speech and motion in the video clip was made with [D-ID](#), an AI video generator creating videos from videos and avatars. While our applications for such videos is mainly for the “nice-to-have” introductions, AI in video generation, especially in realistic lip-synced speech, saves a lot of time and makes it accessible for general use, not requiring lengthy training.



Fig.7: Vyond video



Fig.8: D-ID video

4. Tools for more complex training material

4.1. AI-generated presentations

Developing presentations (such as for a conference) and trainings, whether based on presentations or e-learning frameworks, generally takes a lot more time than outsiders suspect, *Bertram and Plowman (2019)*. And time is money, in industry. Against this backdrop, DNV management stumbled across an advertisement of [Docebo Shape](#): “Get AI-based learning content in minutes - Content can make or break your learning programs. Let Shape’s AI do the heavy lifting.”

We tested Docebo Shape on DNV’s “Train the Trainer” course aimed at giving an initial training for ship technology experts with no previous pedagogical experience, *Gaspar et al. (2023)*. Within a minute the uploaded original PowerPoint file was converted to the suggested online content for a training. But the result was largely useless, sometimes amusingly so, as the AI software just picked up keywords and assigned images that matched the keywords, but out of context.

In introducing the course trainer, who holds post-graduate degrees in pedagogy, an image for a fitness trainer was selected, Fig.9 (top left). Or for training workshops, an industrial workshop with a worker welding was chosen as visual aid, Fig.9 (top right). The verb “train” was visualized with an image of a train, Fig.9 (bottom left). Text was chopped into fragments and reassembled, adding apparently random text snippets to form new sentences, which were often nonsensical, Fig.9 (bottom right). For example, “Bernhard Löbermann has a master [degree] in teaching. He has developed trainings in maritime applications for specific purposes.” was converted into “Bernhil Loesbermann has a Masters in Teaching for Specific Purposes”. (Typos were introduced by Docebo Shape.)

In summary, the AI-based software created useless and often nonsensical material. We looked at other AI tools to create complete trainings and presentations “within minutes” and they fell similarly short of the marketing promises and our requirements, *Gaspar and Bertram (2023)*.

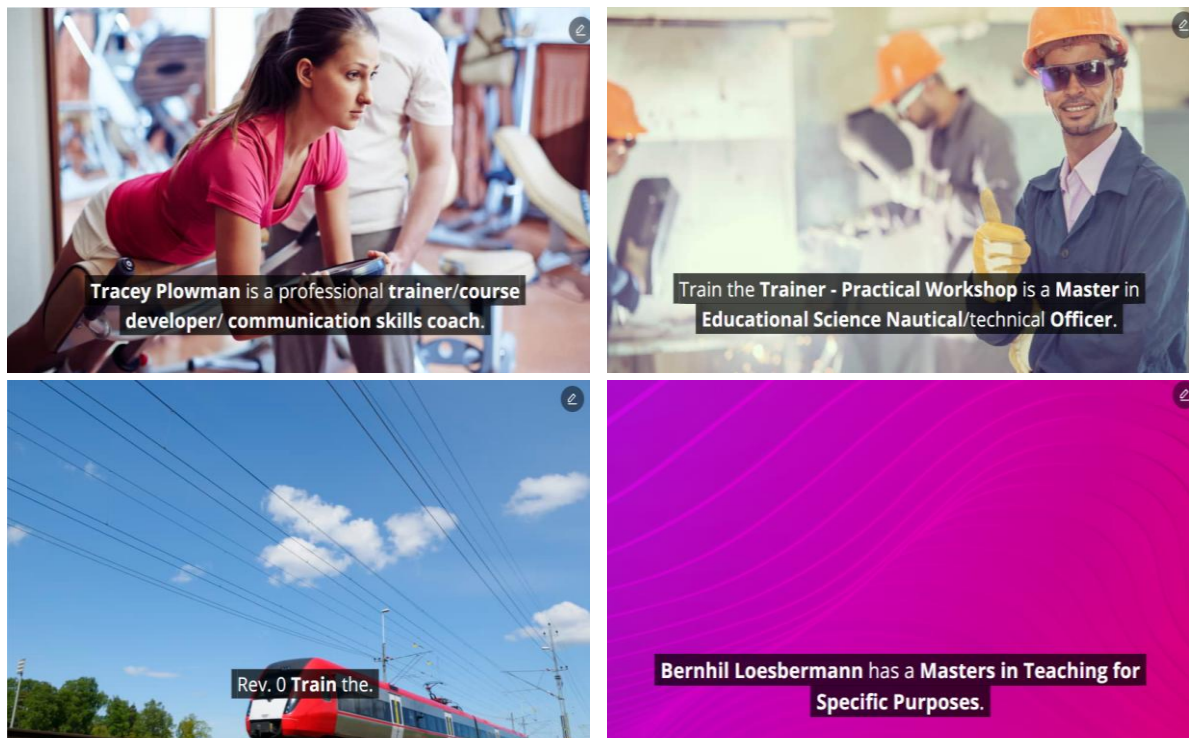


Fig.9: Images are selected based on keywords without context resulting in “nonsense”

4.2. AI-generated assessment quizzes

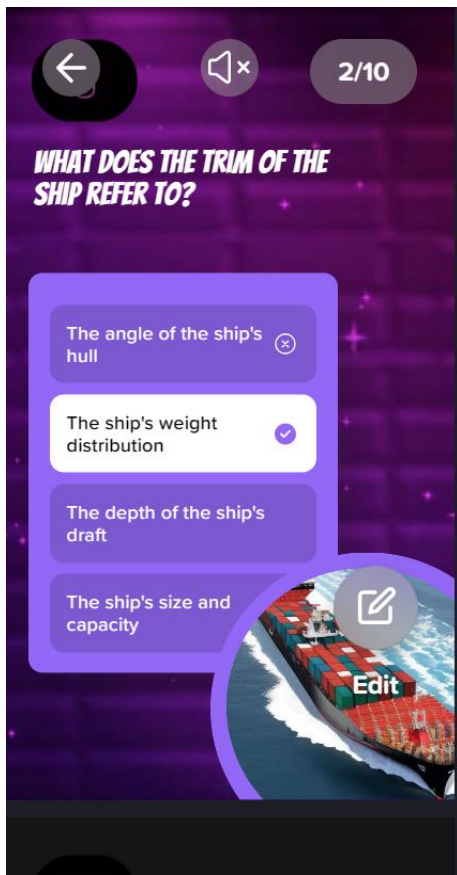
In training development at DNV, the starting point is often a rather factual PowerPoint presentation, for purely frontal teaching. Both for pedagogical reasons and for required assessment of learning success, we then add quizzes that can be automatically assessed in the learning platform. Typical question types are multiple choice, multiple response, matching and filling in blanks. Multiple choice questions are used the most in our applications.

[Piggy](#) is a “visual, creative, and interactive mobile documents creator”. Its website states that it is intended to “create social stories, presentations, and quizzes”, designed for use on mobile phones. We asked Piggy to create a quiz on “ship stability”, *Gaspar et al. (2023)*. Some of the quiz questions were OK, some wrong or nonsense, Fig.10:

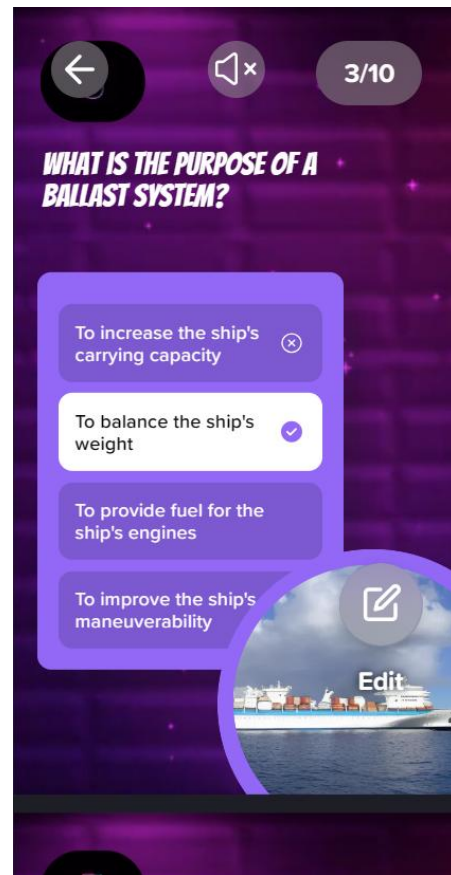
- The trim refers rather to an angle of the hull (first choice) than the ship’s weight distribution (second choice, given as the correct answer).
- The purpose of ballast tanks is certainly not to balance the ship’s weight.
- The effect of low GM is given correctly as ship “will capsize easily”, but “ship gets unstable in rough sea” is arguably also possible, and ship will be difficult to manoeuvre or be slow and inefficient can be easily ruled out in a quiz on stability.
- The metacentric height GM is not the distance between centre of buoyancy and centre of gravity.

With various other maritime topics, results were similarly problematic. The quizzes created by Piggy contained serious errors. With increasing level of specialization, the results deteriorated. Piggy may be a good tool for party games or ice breakers in training, but otherwise failed to convince.

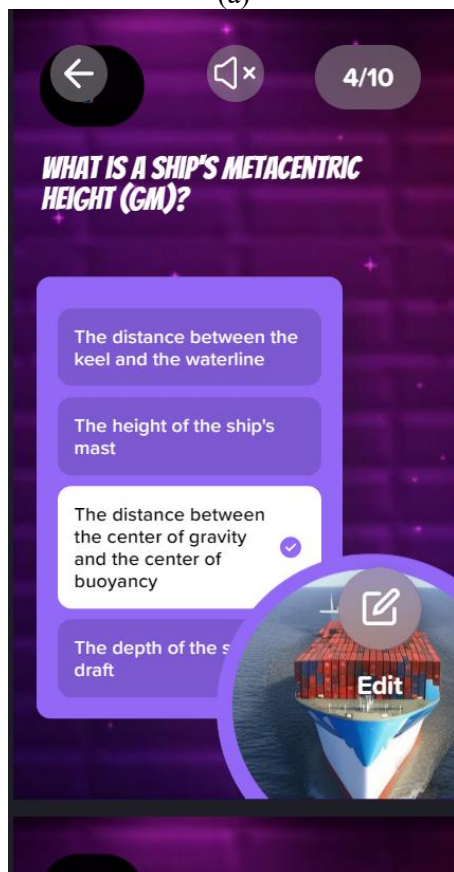
Designing suitable quizzes for a given training requires context information into learning objectives and background of target audience. It is a high-level task in training development. Quiz tasks should assess the learning success, i.e., they should not be so easy to guess the correct answer before the training, but reflecting key training material, such that they are answered correctly after the training.



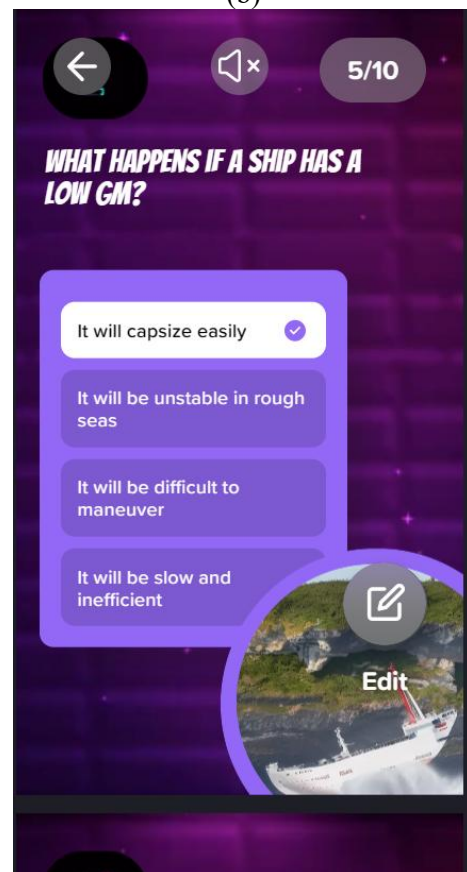
(a)



(b)



(c)



(d)

Fig.10: Sample pages for ship stability quiz from Piggy

5. Conclusions

Our assessment of the state of the art of AI for maritime training development may be summarized as: the glass is half full or half empty, depending on your perspective and your expectations.

- Capabilities have progressed rapidly over the past few years to sometimes impressive levels, and AI-based tools can be used effectively for base tasks, such as translation, image generation, or the generation of lip-synced videos.
- We still don't have the "make training" button which magically designs complete training material. High-level tasks requiring context information and sometimes even creativity are best performed by human experts, e.g., core tasks of pedagogy such as defining what to teach and through what means.

The tools tested performed sometimes well, sometimes poorly, but always worse than in the marketing demos for non-maritime applications. That should not come as a surprise. Always test software for your own test cases, not the vendor's test cases. Many of the failures of AI-tools for training described here were to be expected. The fundamental shortcomings of statistical methods (such as machine learning) are known as such, *Bertram (2018)*, even if the wider public is not so aware yet.

The development of software tools for training design may be seen in analogy to the development of tools for computer aided ship design. For four decades, efforts to use AI to develop "computer generated ship design", i.e., a magic "make ship" button where just the requirements are specified and the computer generates the complete design, Fig.11, have failed, *Bertram (2013)*. However, we have seen continuously increased efficiency in the design process, shifting low-to-medium complexity tasks to the computer. It may well be that we see a similar development for designing trainings, where training developers get increasingly efficient in the product development, as the computer handles more basic tasks with increasing autonomy. Computer-aided training development, rather than computer-generated development.

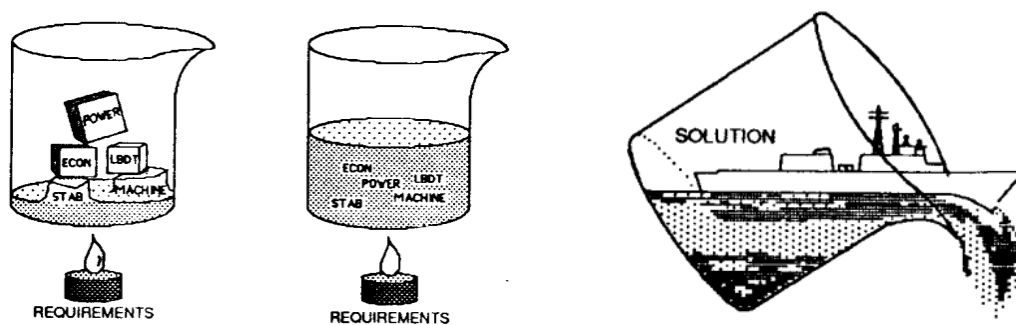


Fig.11: The vision of a magic "make ship" design, *Mistree et al. (1990)*, never materialized

Acknowledgements

We are grateful for many colleagues sharing success and failure stories on using AI-based tools discussed here, namely Bjørnar Krukhaug, Michelle Kurian, Bernhard Löbermann, and Alessio Vicini (all DNV), Henrique Gaspar (NTNU).

References

- BERTRAM, V. (1994), *Numerische Schiffshydrodynamik in der Praxis*, Schriftenreihe Schiffbau 545, <https://d-nb.info/1071620541/34>
- BERTRAM, V. (2013), *A survey on knowledge-based systems for ship design and ship operation*, Int. J. Intelligent Engineering Informatics 2/1, pp.71-90

- BERTRAM, V. (2018), *Demystify Artificial Intelligence for Maritime Applications*, 17th COMPIT Conf., Pavone, pp.22-35, http://data.hiper-conf.info/compit2018_pavone.pdf
- BERTRAM, V., COUSER, P. (2010), *Aspects in selecting the right CAD and CFD software*, 9th COMPIT Conf., Gubbio, pp.20-29 , http://data.hiper-conf.info/compit2010_gubbio.pdf
- BERTRAM, V.; PLOWMAN, T. (2019), *A Hitchhiker's Guide to the Galaxy of Maritime e-Learning*, 18th COMPIT Conf., Tullamore, pp.7-23, http://data.hiper-conf.info/compit2019_tullamore.pdf
- COLLÉ, C.; MOROBÉ, C. (2022), *Saving AI from its Own Hype: Getting Real about the Benefits and Challenges of Machine Learning for Ship Performance Modelling Aimed at Operational Optimizations*, 7th HullPIC Conf., Tullamore, pp.198-206, http://data.hullpic.info/HullPIC2022_Tullamore.pdf
- GASPAR, H.; BERTRAM, V. (2023), *The Use of ChatGPT and Similar AI in Marine Engineering: Limitations and Opportunities*, 24th COMPIT Conf., Drübeck, pp.73-83, http://data.hiper-conf.info/compit2023_drubeck.pdf
- GASPAR, H.; BERTRAM, V.; PLOWMAN, T. (2023), *Hic Rhodus, Hic Salta: ChatGPT and other A.I. Tools for Maritime Applications*, 15th HIPER Symp., Bernried, pp.41-52, http://data.hiper-conf.info/Hiper2023_Bernried.pdf
- MISTREE, F.; SMITH, W.F.; BRAS, B.; ALLEN, J.K.; MUSTER, D. (1990), *Decision-based design: A contemporary paradigm for ship design*, Trans. SNAME 98, pp.565-595, https://www.researchgate.net/publication/236686681_Decision-Based_Design_A_Contemporary_Paradigm_for_Ship_Design

A State-Of-The-Art Report on Simulating Ships in Arctic Waters

Hazel Mitrik, USCG Academy, New London/USA, hazel.a.mitrik@uscga.edu
Thomas DeNucci, USCG Academy, New London/USA, thomas.w.denucci@uscga.edu
Volker Bertram, VB Conferences, Hamburg/Germany, volker@vb-conferences.com

Abstract

This paper surveys numerical approaches to simulate ships and offshore structures in ice-infested waters, excluding the direct simulation of icebreaking. Discrete element techniques model many individual ice floes, while classical continuum mechanics and computation fluid dynamics model the flow of water. Machine learning based simulations also show promise in predicting ship behavior in ice conditions. The state of the art has reached a maturity sufficient for many industrial applications.

1. Introduction

The Arctic is undergoing a rapid transformation due to climate change, with significant reductions in sea ice extent increasingly observed in recent decades. According to the *National Snow and Ice Data Centre (2021)*, Arctic Sea ice extent has declined by approximately 40% since the late 1970s, leading to improved accessibility for maritime activities. As the ice cover decreases, shipping activity in these previously hard-to-navigate waters has surged, driven by a combination of geopolitics, commercial interests, resource exploration, and new shipping routes. The Northern Sea Route and the Northwest Passage are now increasingly viable alternatives for global shipping, offering substantial reductions in transit times and fuel costs compared to traditional routes through the Suez or Panama Canals.

Recent statistics underscore this trend. According to the *Arctic Council (2024)*, the number of unique ships entering the Arctic region has risen by 37% from 2013 to 2024. In 2024 alone, 1,781 unique ships entered the arctic waters—nearly 500 more than when data collection began in 2013. This increase reflects a broader shift, as year-round shipping operations and seasonal voyages are establishing themselves as standard practices in the region. As a result, there is a need for improved understanding of ship-ice interactions in these changing sea conditions.

2. Ice Types

Sea ice in polar regions can be broadly categorized into two main types: level ice, which exists as a continuous surface, and broken ice, which consists of discontinuous pieces. According to *Li and Huang (2022)*, broken ice includes various forms such as brash ice, sliding ice pieces, and ice floe fields that emerge through natural processes. Broken ice features irregularly shaped fragments of varying sizes, ranging from small chunks to larger plates, with thicknesses that can significantly differ. These variations in size and thickness influence the stability and buoyancy of the ice.

Brash ice is defined as an accumulation of floating ice fragments that are no larger than 2 m across. It is characterized by its relatively small size and typically forms an ice layer over broken channels at full concentration. In the context of channels, brash ice is specifically characterized as being thicker than one layer. Brash ice also does not include features such as rafted ice or pressure ridges; it strictly pertains to these floating fragments, a characteristic that plays a crucial role in understanding the ship-ice dynamics in shipping channels, *Griesman (1981)*.

Ice floe fields, on the other hand, are defined by numerous characteristics including ice concentration, diameter, thickness, and shape. An ice floe is a relatively flat piece of sea ice that measures 10 m or more across. Floes are categorized based on their horizontal extent: small (10-100 m), medium (100-500 m), big (500-2000 m), vast (2-10 km), and giant (greater than 10 km). These floes can freeze together to form larger ice fields and are distinct in that they float on the sea surface. Fig.1 shows U.S. Coast Guard ships operating in brash ice and in ice floes.



Fig.1: Vessels operating in brash ice (left) and in ice floes (right)

3. Ice Resistance Prediction Methods

The primary methods employed for predicting ice resistance include full-scale tests, ice tank model tests, empirical formulas, and numerical simulations. Among these methods, full-scale testing is widely regarded as the most reliable approach. However, it is not suitable for concept design and is usually conducted at trials and data gathered helps inform other models. Model tests conducted in ice tanks offer the most direct means of estimating ice resistance. *Mueller and Ettema (1984)* conducted model experiments with a scaled version of the USCGC Polar Star to investigate the interaction between ice forces and the dynamic motion of the hull as it traversed an ice sheet. *Daley (1991)* investigated and proposed the contact process at the edge of the ice layer using real ship tests and model test methods. *Matala and Suominen (2023)* developed a new scaling principle for model tests in old brash ice channels by considering the interparticle cohesion of the ice. The channel similitude number scaled-down flexural strength altered the ice fragment interaction, which is a substantial factor in ship's resistance in an old brash ice channel and results in resistances lower than those determined using current scaling principles.

Table I: Summary of different ice resistance prediction methods and limitations, *Sun et al. (2024)*

Method	Test Conditions	Computational Cost	Suitability	Research Gap
Full-scale test	Ice-covered water	-	Accurate and reliable data	Expensive and non-replicable
Model test	Model ships and ice water pools etc.	-	Applicable to existing or newly designed model ships	Expensive and time-consuming testing
Empirical/semi-empirical formula	Simplify ship-ice interaction through reasonable assumptions, combining tests	Low	Application background, assumed conditions, scope of application	Should improve the applicability and prediction accuracy
Numerical model	Deeply understand the mechanism of ship ice interaction and proficiently apply numerical software	High	Compared with experimental methods, low cost and strong replicability	Often simplifying and assuming some conditions
Deep learning	Dataset collection, data preprocessing, and model selection optimization, etc.	Training: medium Calculation: low	Theoretically applicable to all scenarios of ships	Scale effect, high requirements for dataset quality

While the advancements in model testing have significantly enhanced our understanding of ice resistance, it is important to acknowledge the inherent limitations that accompany these methodologies. Conducting model tests can be time-consuming and costly, particularly for comprehensive testing across different scenarios, which might limit the number of tests that can be performed. Model tests

often focus on certain predefined scenarios, and they may not address all aspects of vessel maneuvering in ice, such as dynamic response to changing conditions or emergency maneuvers. Table I summarizes the different ice prediction methods along with their required test conditions, computational cost and suitability.

4. Semi-empirical and qualitative approaches

Over the years, a variety of ice resistance formulas have been developed. *Lindqvist (1989)* developed a relatively straightforward empirical model that accounts for key ship hull characteristics, ice thickness, ice strength, and friction. This method paved the way for a standardized approach to assessing vessel resistance in level ice, with subsequent research aimed at refining its details, *Kämäräinen (2007)*. *Riska and Leiviskä (1997)* developed an approach for calculating resistance in level ice, utilizing empirical coefficients derived from full-scale data collected from several ships operating in the Baltic Sea. In one of the most comprehensive studies on ship resistance to date, *Keinonen (1991, 1996)* analyzed data from 18 icebreakers through extensive fieldwork and operator interviews. *Spencer and Jones (2001)* explored ice resistance prediction methods, introducing a component-based approach that identifies four distinct resistance forces encountered during icebreaking. However, it is important to note that the accuracy of these empirical formulas can be significantly off, with calculation errors reaching up to 40% when compared to model test results.

Recent improvements in computing power have redirected much of the research effort toward the reformulation of semi-empirical equations used to measure ice resistance. Most approaches differentiate ice resistance into two primary components: breaking (or crushing) and submersion (or bending). A detailed literature review is presented in *Kämäräinen (1993)*, which features several validation cases. More recently, *Erceg and Ehlers (2017)* examined six semi-empirical methods for estimating level ice resistance, testing them on four vessels of differing sizes and bow designs. Their results are supported by full-scale resistance measurements, offering valuable insights into the effectiveness of these methods. Researchers are actively working to enhance existing ice prediction techniques while also developing new ones. Examples of this include *Cho et al. (2014)*, *Jeong et al. (2017)*, and *Myland and Ehlers (2019)*, who have integrated traditional semi-empirical approaches into numerical models.

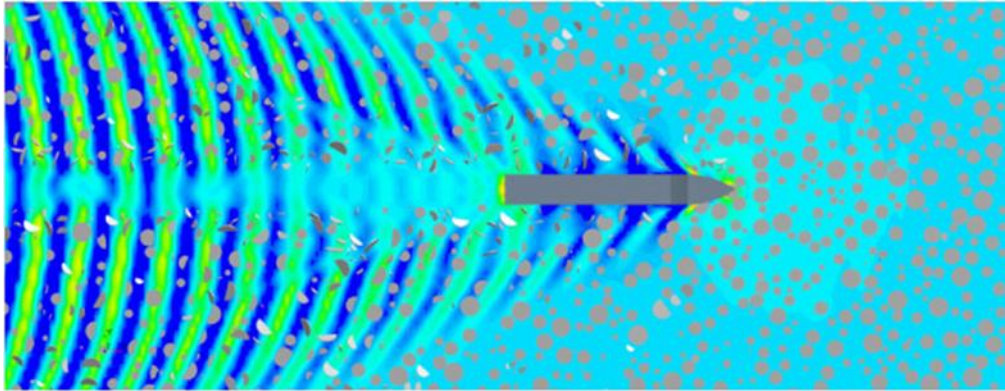
5. Introduction to numerical simulation of ice

Numerical simulations are fundamentally more precise than semi-empirical methods, as they are grounded in first principles; however, they can also be computationally intensive. Several approaches exist for modeling ice in numerical simulations:

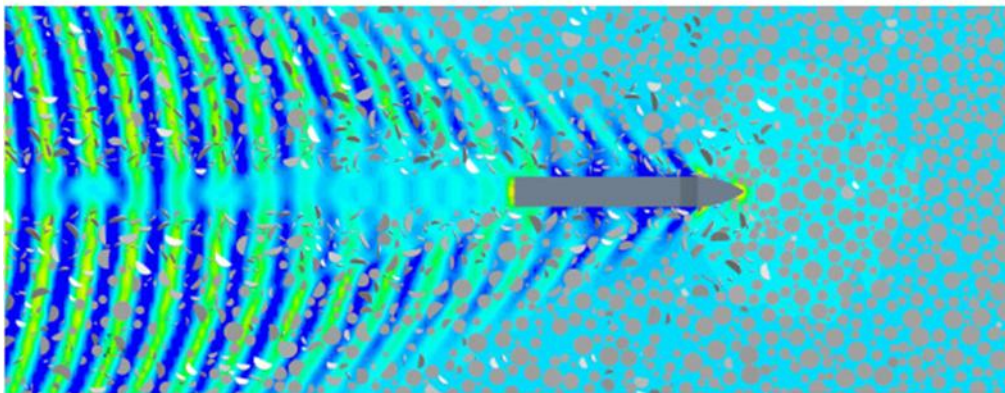
- Discrete Element Modelling (DEM): This technique involves modeling and tracking individual ice floes with discrete element models, widely adopted for simulating ice-structure interactions, *Tuhkuri and Polojärvi (2018)*. Typically, ice floe shapes are predefined, with circular and rectangular forms being the most used. Other studies, including those by *Yamaguchi et al. (1997)* and *Konno and Mizuki (2006)*, explore shapes like paralepidids, while *Wang and Derradji-Aounat (2011)* examine more arbitrary forms.
- Finite Element Method (FEM): FEM simulates ice behavior by subdividing the ice domain into finite elements. Its strengths lie in effectively modeling contact collisions, deformation, and damage, as well as the interactions between the hull and the ice, *Kuutti et al., (2013)*. For instance, *Kim et al. (2019)* employed FEM to analyze the loads exerted on a ship's hull due to ice impacts in fragmented ice fields. They modeled interactions using parameters such as drag coefficients and pressure-penetration curves, derived from comprehensive analyses with the coupled Eulerian-Lagrangian method.
- Computational Fluid Dynamics (CFD): Often used in conjunction with other models, CFD simulates interactions among ice, water, and ships. *Huang et al. (2020)* integrated CFD and DEM to explore the dynamics between ships, waves, and ice, Fig.2. *Zhong et al. (2023)*

developed both experimental and numerical methods to study ice floe-induced hull pressure, facilitating localized monitoring of these pressures within the CFD-DEM framework.

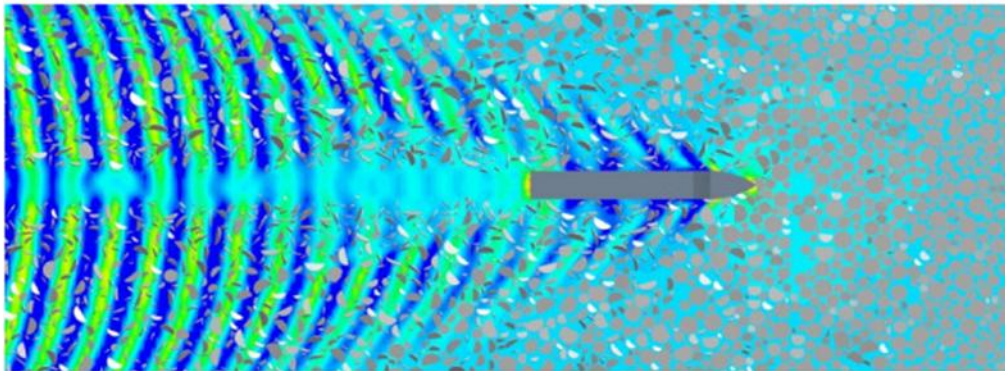
- Machine Learning and Data-Driven Techniques: Recent advancements have introduced machine learning models that can predict ice behavior using historical data, environmental factors, and observed patterns from simulations or real-world situations. *Sun et al. (2024)* proposed a deep learning method for generating data related to ice behavior.



(a) Ice concentration = 30%



(b) Ice concentration = 50%



(c) Ice concentration = 70%

Fig.2: Simulation of a ship advancing in floating ice floes, *Huang et al. (2020)*

Over the past decade, numerical simulations for ships operating in ice have seen considerable progress, yet the field remains relatively underdeveloped. The current landscape consists primarily of specialized research communities focused on innovation rather than commercial product

development, *Veeramani et al. (2020)*. This domain features a wide variety of methodologies, often varying in origin - from internal development projects to established commercial software solutions, *Thomas and Kuhl (2019)*. Many approaches tend to concentrate on specific aspects of ship-ice interactions, highlighting that the field is still evolving. Consequently, the current state of the art appears somewhat immature, emphasizing research efforts aimed at refining and innovating techniques rather than delivering robust commercial applications, *Jiang et al. (2021)*.

6. Numerical simulation of brash ice

Empirical estimation of broken ice resistance is relatively uncommon compared to established formulae for level ice resistance. *Riska et al. (1997)* present formulae specifically for brash ice, channel ice, and ice ridges. However, numerical simulation has emerged as a powerful tool to model ship-ice interactions with enhanced speed, fidelity, and accuracy. Given the discrete nature of broken ice, DEM techniques are particularly useful in modelling broken ice.

6.1. Discrete Element Modeling (DEM)

DEM offers significant advantages for calculating ice loads due to its ability to represent the discrete nature of ice at the micro-scale while accurately modeling the icebreaking phase during ship-ice interactions at a macro level, *Xue et al. (2020)*. Since the foundational work by *Cundall and Strack (1979)* on DEM for granular assemblies, the method has evolved and found numerous applications in ice simulation.

One of the early applications of DEM in ice modeling was by *Løset (1994)*, who developed a two-dimensional discrete element model to simulate dynamics and interaction forces within a broken ice field. This laid the groundwork for further studies. *Karulin and Karulina (2014)* utilized a two-dimensional disc DEM to explore the interaction processes between sea ice and ships operating at various speeds. Their findings were corroborated by comparing the results with existing ship model tests. *Prasanna and Hisette (2018)* focused on simulating ships navigating through brash ice. Their study demonstrated that while the particle behavior near the hull was satisfactory, the predicted ice loads tended to be higher than anticipated. Nevertheless, this tool proved valuable in providing qualitative insights into ship navigation within brash ice channels during the early design stage, particularly in visualizing ice particle flow around the hull and identifying potential concentrations of ice particles around appendages.

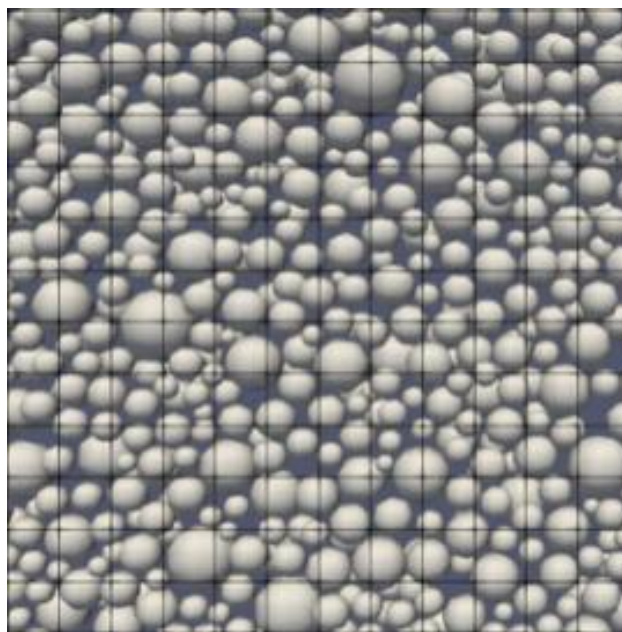


Fig.3: Detail of brash ice in the simulation, *Behen et. al (2020)*

In a more recent development, *Behen et al. (2020)* implemented a DEM model based on the Hertz-Mindlin Deresiewicz contact model, which accounts for elastic viscous behavior. Their simulation results showed significant consistency with HSVA data, though they also revealed a greater level of variation and larger individual peaks in the graphical representations, illustrated in Fig.3.

6.2. CFD-DEM Approaches

DEM is widely employed to study ice-ship interactions; however, it struggles to incorporate the effects of water alone. To address this, a hybrid approach combining CFD with DEM has gained traction. This method employs the Euler method to solve fluid flow while utilizing the Lagrangian method to model ice particle movement.

Vroegrik (2015) utilized the commercial code StarCCM+ with DEM to model brash ice flow around an obstacle. This study showcased the feasibility of CFD in tracking particle flow within environmental contexts, paving the way for further investigations that adopted hybrid CFD-DEM methodologies. *Luo et al. (2020)* implemented a CFD-DEM approach to simulate a bulk carrier navigating through brash ice. Their research involved coupling numerical methods to assess the ship resistance of an ice-strengthened Panama bulk carrier in these challenging conditions. They adopted advanced computational techniques, including finite element analysis, to precisely depict the ice's physical properties and the vessel's structural responses. Their results indicated a close correlation with experimental data from the HSVA ice tank, with an average error of approximately 5.13%.

Zhang et al. (2021) used CFD-DEM simulations to create a ship-ice contact model and compared it with full-scale ship-ice interaction data in a confined brash ice channel. The collision, accumulation, extrusion, rolling, and sliding behaviors observed in the simulations closely matched those seen in ice tank tests, Fig.4. This suggests that the proposed full-scale CFD-DEM method for directly assessing ship-brash ice interactions provides reliable results and holds practical significance. *Xie et al. (2023)* expanded the application of the CFD-DEM methodology to examine ship-ship interactions within brash ice, particularly noting the significant impact of proximity on these dynamics. Their approach simulated the hydrodynamic interactions and ice dynamics of multiple vessels navigating in close formations, utilizing specific rheological models to characterize the brash ice's properties and its interactions with ship hulls. The validity of their models was reinforced through comparisons with prior experimental and field data. *Zhang et al. (2024)* introduced wave effects into the coupled CFD-DEM analysis, employing overlapping grid technology to model ships' motion responses under the combined influences of nonlinear waves, currents, and brash ice. Their work examined both single vessels and formations navigating through these complex conditions, further enhancing the understanding of ice-ship interactions in maritime environments.

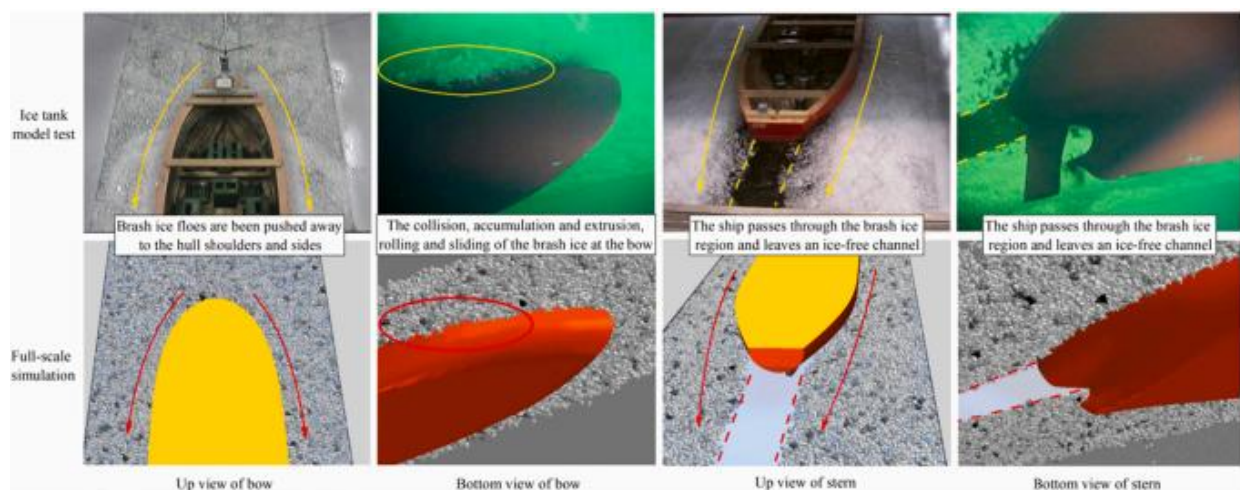


Fig.4: The brash ice distribution comparison between the ice tank model test and the corresponding full-scale simulation, *Zhang et al. (2021)*

Liu et al. (2025) utilized a CFD-DEM model incorporating an overset grid technique to investigate forward ship motion in brash ice channels. This research improved the understanding of ship-ice-water interactions by adopting a quasi-physical framework where the ship moves while the ice remains stationary. They emphasized the need for careful selection and calibration of contact model parameters, as these significantly influence the accuracy of ice resistance predictions.

Zou et al. (2024) utilized a hybrid CFD-DEM approach to simulate the interaction between the ship and the brash ice, allowing for a detailed investigation of the forces exerted on the ship hull, Fig.5. The ship's geometry and operational conditions were defined, and the simulation incorporated varying ice concentrations and channel conditions to evaluate how these factors influence the ice forces acting on the ship. The findings indicate that the distribution of ice forces on the ship hull is significantly affected by the ice concentration and the flow dynamics within the brash ice channel. The study revealed that as the ice concentration increases, the forces exerted on the ship's hull also rise, showcasing a direct correlation between these two variables.

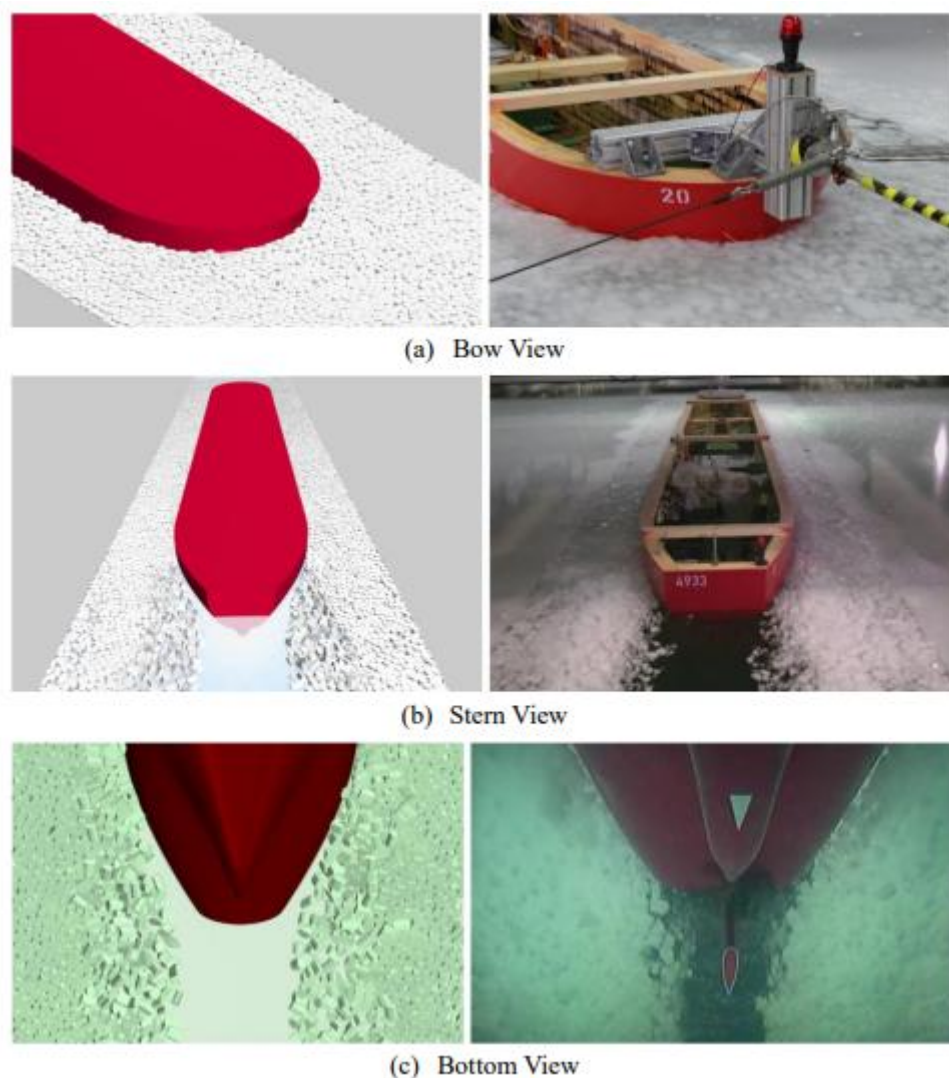


Fig.5: Simulations (left) and experiments (right), *Zou et al. (2024)*

6.3. Other numerical methods for brash ice

Konno et al. (2007) developed a physically based model to simulate a ship in brash ice with an open dynamics engine (ODE). Since the code was multipurpose and designed to deal with different scales of problems, it didn't accurately capture friction and damping forces. Improvements to this method

were demonstrated in the model by *Konno and Yoshimoto (2008)*, which simulated a ship navigating through a channel filled with 1250 pieces of brash ice. *Konno (2009)* and *Watanabe and Konno (2011)* extended the approach to 33000 and 126000 pieces, respectively. They also integrated base water flow using the open-source CFD solver OpenFOAM.

Wang and Derradji-Aouat (2011) described a simulation involving ice floes drifting against an offshore structure, utilizing the commercial code LS-DYNA (www.lstc.com). In this simulation, ice floes were modeled as both rectangular and arbitrary shapes, revealing that shape had little impact on the overall results. The study explored three different ice concentrations, modeling water and air with the Eulerian method while employing the Lagrangian method for the offshore structure and pack ice. Semi-empirical friction coefficients were utilized to characterize the friction between ice floes and the structure. Figs.6 and 7 show rectangular and randomly shaped ice at varying concentrations.



Fig.6: 80%, 50%, and 20% concentrations, *Wang and Derradji-Aouat (2011)*

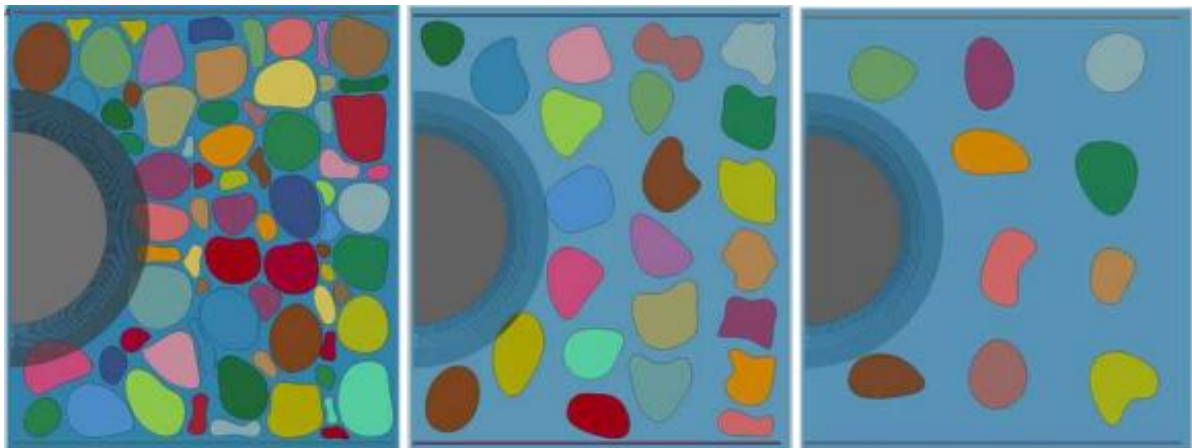


Fig.7: 80%, 50%, and 20% concentrations with random shaped ice, *Wang and Derradji-Aouat (2011)*

Cabrera (2017) employed Smoothed Particle Hydrodynamics (SPH) to create a mesh-free simulation. This innovative model allowed for complex particle interactions through force calculations that included both viscous and pressure forces, yielding realistic representations of ice dynamics. Enhancing this technique, *Chen (2022)* tackled issues related to mesh distortions and disappearance in Finite Element Method (FEM) by integrating it with SPH. However, challenges remain, as simulating a ship using SPH in an enlarged water domain necessitates further development of existing code.

7. Numerical simulation of ice floes

Ice floes are the most prevalent type of fragmented ice found in polar regions. They can naturally form from newly frozen water or result from the breakup of larger ice masses due to heat or ocean

waves, *Li and Huang, (2022)*. The focus of numerical simulations involving ice floes is primarily on resistance, manoeuvrability, and the impacts of ice on the hull, rather than the specific formation methods. *Huang (2020)* addresses the challenges posed by operating ships in environments with floating ice floes by examining the various types of resistance that vessels encounter due to ice interaction and emphasizes the importance of understanding these factors for safe navigation and optimization of ship hulls for Arctic operations.

7.1. Ice floe resistance

Recent advancements in numerical simulation methods have significantly enhanced the understanding and prediction of ship resistance while navigating through pack ice. *Huang et al. (2024)* introduced a novel two-dimensional (2D) numerical simulation method designed specifically to predict the resistance faced by ships operating in pack ice. This approach is characterized by a semi-theoretical and empirical framework. Comparative analyses revealed that the simulations effectively reproduced the interactions between the ships and ice, successfully capturing critical ice failure modes such as sliding, splitting, and bending. Notably, the total discrepancy between the calculated average navigation resistance from the simulations and the resistance measured during the model tests was a mere 9.05%. *Huang et al. (2024)* also explored the modeling of smaller ice floes. They employed a coupled CFD–DEM simulation to assess ice resistance in smaller floes, considering the effects of ship-generated waves. This aspect is crucial for enhancing the accuracy of predictions in varying ice conditions.

The application of advanced computational methods has further refined predictions of ship resistance in ice. *Huang et al. (2022)* highlighted the widespread use of the Finite Element Method (FEM) and Discrete Element Method (DEM) in this domain. These methodologies are often integrated with commercial Computational Fluid Dynamics (CFD) software, enabling the development of combined CFD–FEM and CFD–DEM simulation approaches. This integration has proven effective in simulating ship resistance in ice floes, as suggested by *Zhong (2023)*.

7.2. Manoeuvring

Zhan and Molyneux (2010) utilize the 2D version of DECICE (Discrete Element Code for Ice-related problems) to analyse tug-tanker ship manoeuvring in ice. *Lau and Simoes Ré (2006)* used the code to model a ship operating in pack ice. *Lawrence (2009)* expanded the code's capabilities in his PhD thesis. *Liu et al. (2010)* applied DECICE3D—a 3D version of DECICE—to simulate offshore structures, including a drill ship, navigating through pack ice. In a subsequent study, *Zhan and Molyneux (2012)* also employed DECICE3D for ship manoeuvring in icy conditions. DECICE functions as a time-domain solver, employing discrete elements to tackle ice engineering challenges. These elements experience rigid-body translation and rotation based on classical Newtonian mechanics. Interaction forces, including both normal and tangential forces arising from friction, are modelled during contacts between bodies (whether floe-floe or floe-ship). *Molyneux et al. (2012)* validated DECICE simulations against model tests, and their findings for three different ship designs and varying ice conditions demonstrated a strong correlation with the results from experimental models.

Wang et al. (2020) used a finite element to simulate the navigation of an ice-area bulk carrier in ice floes. The ice is defined as elastic, and the simulations, Fig.8, were accomplished at four model speeds and three ice concentrations. In terms of ship-ice-water interaction phenomena, the ice floes tend to overlap, accumulate, and overturn at the bow and slip along the side of the ship. Most ice floes that accumulate and slip are small and medium-sized, and larger ones are more prone to overturning. *Yang et al. (2024)* coupled a non-smooth discrete element method (NDEM) with the Manoeuvring Modelling Group (MMG) model to simulate the manoeuvring motion of full-scale ships in broken ice. The simulation results demonstrated significant variations in maneuvering performance based on ice concentration and the physical characteristics of ice.

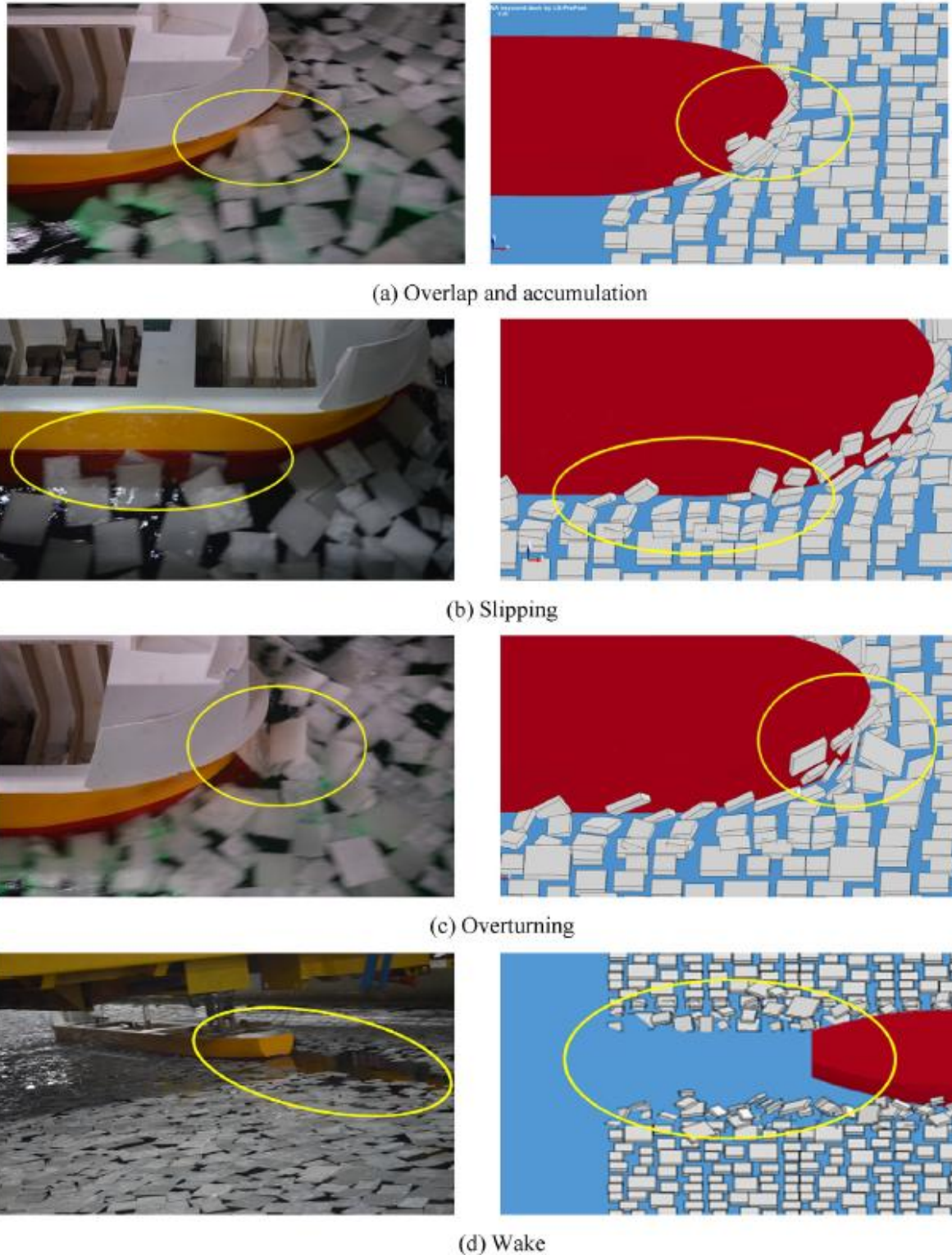


Fig.8. Simulation of various ship-ice interactions, *Wang et al. (2020)*

7.3. Ice impacts

Kim et al. (2019) conducted numerical simulations to analyze the impacts of ice on ship hulls navigating through broken ice fields. They utilized a CFD approach to model the interactions between ice and ship structures, incorporating various parameters such as ice thickness, ship speed, and hull design. The methodology involved creating a detailed simulation environment to replicate real-world conditions, allowing for the examination of stress distributions and potential damage to the hull during

ice impacts. Results of the hull-ice interaction, Fig.9, indicated that the severity of ice impacts varied significantly based on these parameters, highlighting critical vulnerabilities in hull designs and providing insights for improving ship safety in icy waters.

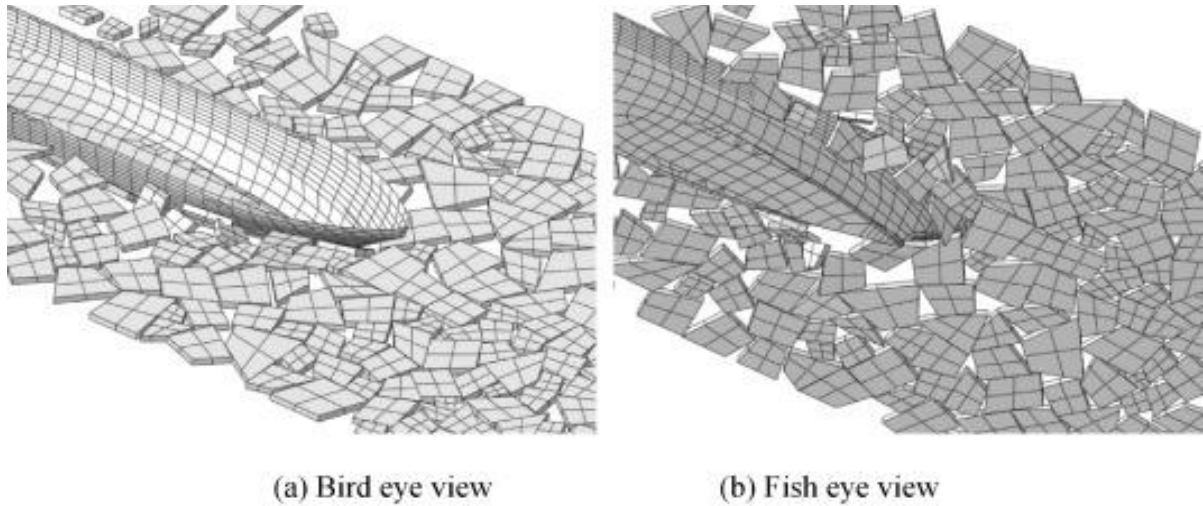


Fig.9: Interaction between ice floes and the ship hull at an 80% concentration, *Kim et al. (2019)*
Xu et al. (2024) conducted a numerical simulation to investigate ice-water-ship interactions using a novel adaptive coupling algorithm combining Finite Element Method (FEM) and Smoothed Particle Hydrodynamics (SPH). The methodology involved creating a detailed computational model that represented the physical interactions between the ship hull, ice, and water, with the adaptive algorithm allowing for effective handling of both solid and fluid mechanics. The results demonstrated that the proposed model accurately captured the complexities of ice breakage and water flow, revealing critical insights into the forces exerted on the ship during ice navigation.

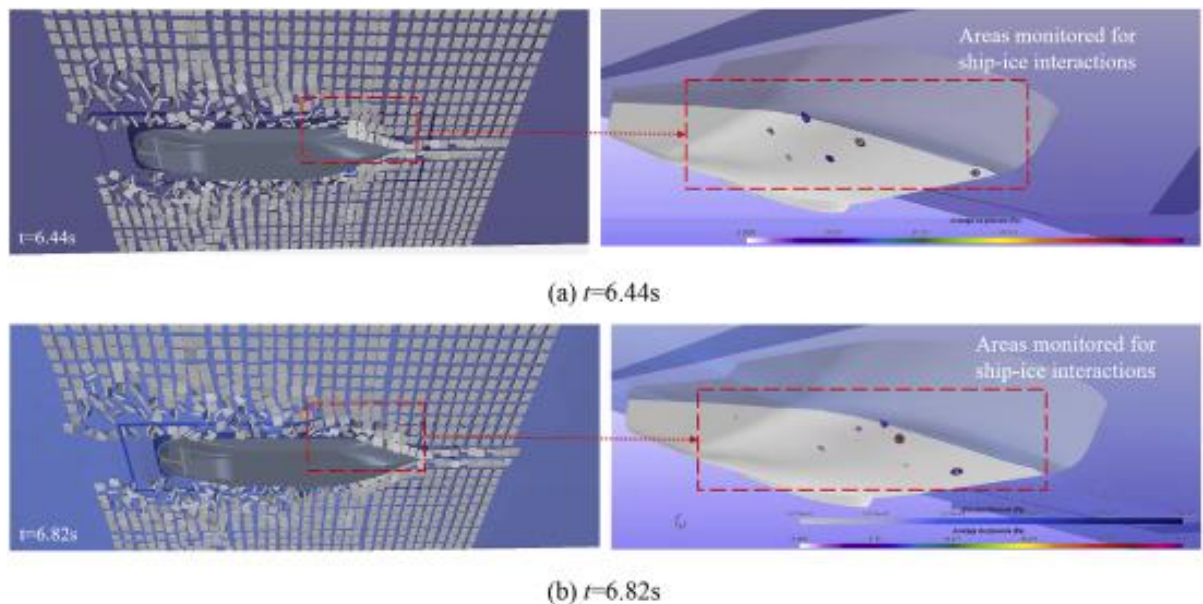


Fig.10: Simulated ship-ice interaction and pressure contour, *Zhong et al. (2023)*

Zhong et al. (2023) employed a CFD-DEM approach to investigate how ice floe shapes, sizes, and ship speeds affect hull pressure characteristics, including water-ice interactions at the hull. The study combined direct measurements with CFD simulations to assess ice-induced hull pressures experienced by ships navigating through floe-ice fields. The researchers conducted field experiments to gather real-time data on hull pressures while a ship operated in ice-laden waters, allowing for precise calibration of their CFD models. The results, Fig.10, demonstrated a strong correlation between measured pressures and simulated values, validating the effectiveness of the CFD approach. *Zhong et*

al. (2024) analyzed ice-induced loads on ship hulls in regions with ice floes using a coupled computational fluid dynamics (CFD) and discrete element modeling (DEM) approach. The methodology involved simulating the interaction between ship hulls and ice floes to quantify the loads experienced by the hulls under various conditions. The results indicated that the ice-induced loads were significantly influenced by factors such as ice thickness, floe size, and ship speed.

Li et al. (2020) presents a methodology for modeling ship-ice interaction using finite element analysis (FEA) to enhance ship performance simulations, specifically focusing on the shoulder and midship areas of the vessel. The authors developed a meta-model to effectively capture the complex interactions between the ship hull and ice, incorporating various factors such as ice properties and ship operational parameters. The methodology involved creating a finite element model to run simulations, which were then used to train the meta-model for quicker predictions. Results indicate that the meta-model significantly improves the accuracy and efficiency of ship performance predictions in icy conditions compared to traditional methods.

8. Ice Simulations using Machine Learning (ML)

Artificial intelligence is a rapidly expanding industry with increasingly widespread applications. Deep learning is a more recent enhancement to the webs of artificial intelligence that has been expanding into areas of traditional knowledge and process-based modelling, *Razavi (2021)*. Deep learning modeling relies on the availability and iteration of existing data as input features for the formation of its neural network for progressive processing, *Sun et al. (2024)*.

Fu et al. (2016) combined data with expert knowledge using Bayesian belief networks to develop probabilistic model for estimating the probability of ship besetting in ice along the Northeast Passage. The researchers employed historical data on ice coverage, vessel characteristics, and environmental conditions to train the model, which was subsequently validated using case studies. The results indicate that the model successfully forecasts the likelihood of vessels becoming beset in ice, providing valuable insights for maritime navigation and safety in challenging Arctic environments.

Li et al. (2020) utilized FEM modeling combined with a neural network to study ship-ice interaction. The approach preserves the accuracy of traditional FEM at the ship-ice interaction scale which significantly reduces computational costs, making it suitable for ship scale simulations. The ice failure process was initially examined quantitatively through full-scale and model-scale observations, along with numerical simulations using the Extended Finite Element Method (XFEM). The model was then simplified and implemented in Abaqus to automatically generate a large database. A neural network was trained on these results to create a fast, simulation-free tool for predicting ship-ice interactions.

Milaković et al. (2020) introduced the first machine learning-based simulator for predicting ship speed profiles in complex ice conditions. The approach involved gathering extensive data on both ice characteristics and ship performance, then applying a range of machine learning algorithms to model the relationship between environmental conditions and ship speed. The simulator was trained and validated using data from full-scale trials of an icebreaking vessel. The model achieved strong accuracy, Fig.11, with average errors between the simulated and actual ship speeds ranging from 2.6% to 9.4%. These results demonstrate the capability of machine learning models to reliably estimate ship speed across varying ice conditions, underscoring their potential to enhance navigational safety and operational efficiency in Arctic shipping.

Sun et al. (2022) developed an Artificial Neural Network (ANN) model for the estimation of total ship resistance in ice-covered waters, with a focus on isolating and predicting the ice resistance component. The model was trained using a hybrid dataset composed of full-scale sea trial measurements and ice tank model test data. Validation results exhibited a high degree of correlation between predicted and empirical resistance values, indicating the ANN's capacity to generalize across varying ice conditions and vessel geometries.

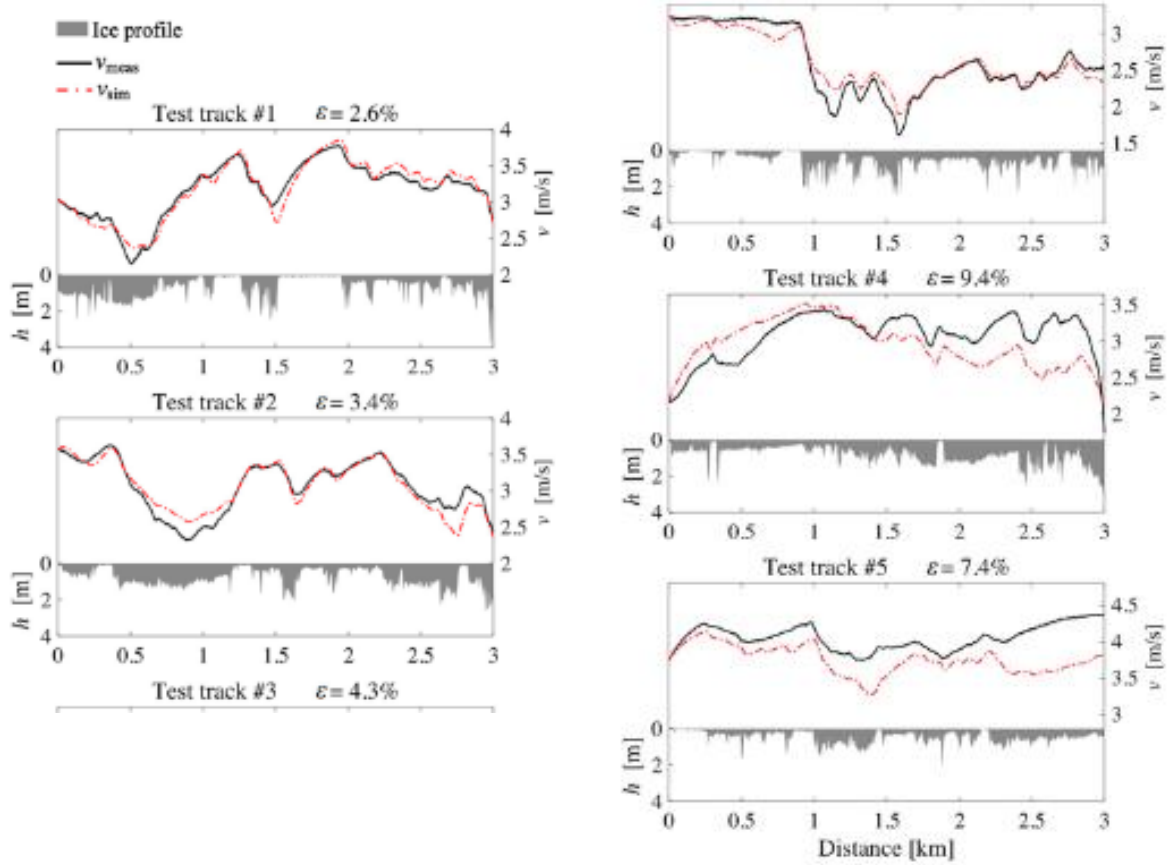


Fig.11: Test results of the ML-based simulator for each of the five test tracks, *Milaković et al. (2020)*

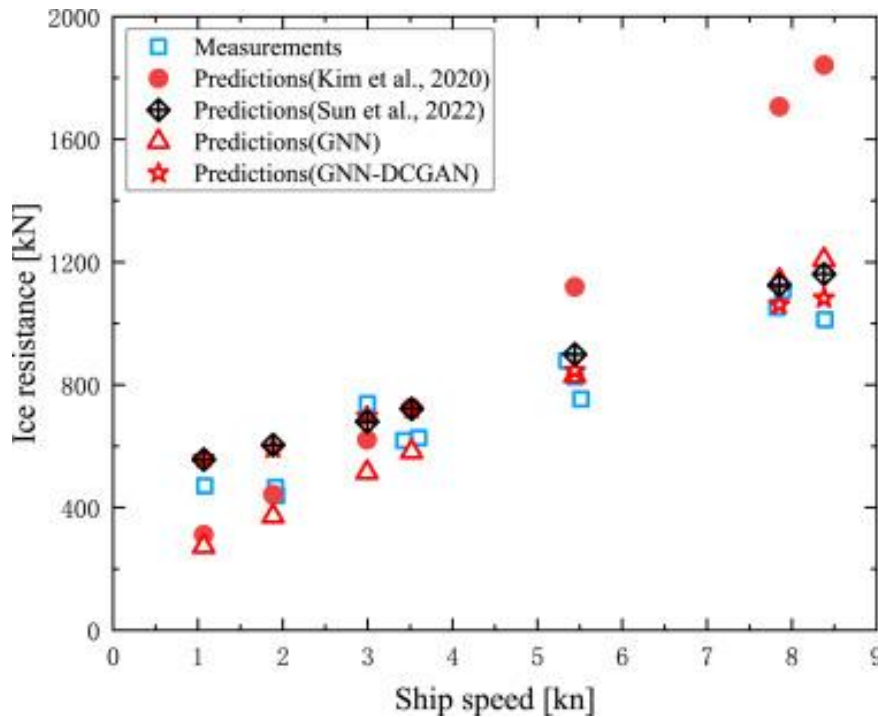


Fig.12: Three ANN methods predicting ice resistance based on Otso ship test data, *Sun et al. (2024)*

Building on this, *Zhou et al. (2023)* presented a supervised machine learning framework for the prediction of propulsion power requirements in ice-infested environments. The model integrates

multiple input features – specifically ice thickness, ambient temperature, vessel speed, and hull form characteristics – and is trained on historical operational datasets. When benchmarked against both model-scale and full-scale trial data, the model demonstrated a mean absolute percentage error (MAPE) of approximately 14%, indicating robust predictive performance suitable for practical deployment in voyage planning and energy management systems under polar operating conditions.

Zhong et al. (2024) addressed the problem of autonomous navigation in ice-covered regions by developing a machine learning-based predictive model for ship-ice interaction dynamics. The approach employs data-driven learning algorithms to infer spatial-temporal patterns of ice impact forces and resistance, facilitating enhanced path planning and real-time navigational decision-making. The model is designed to operate within autonomous control frameworks, contributing to increased situational awareness and operational safety in dynamic and unstructured polar environments.

Sun et al. (2024) advanced the state of the art in ice resistance prediction by developing a deep neural network model tailored for icebreaker hull design applications. The input feature space includes 11 parameters: principal ship dimensions (length, breadth, draft), hull form attributes (stem angle), kinematic variables (speed), and ice properties (thickness, flexural strength, etc.). The model was trained on a dataset of 300 entries derived from ice tank experiments covering 21 distinct hull types. Preprocessing included normalization and dimensionality reduction to address heterogeneity in input scales. A scale extrapolation factor was applied to enable generalization to full-scale operational scenarios. The model achieved high prediction accuracy, affirming its potential utility in early-stage design optimization and performance assessment under ice-loading conditions.

To enhance data generation and improve model robustness, the study also explored the use of Generative Adversarial Networks (GANs), addressing common challenges such as gradient vanishing and unstable outputs. A novel approach was proposed—combining Gaussian fuzzy preprocessing with a Deep Convolutional GAN (DCGAN) architecture, wherein the conventional feedforward neural network is replaced by a Graph Neural Network (GNN) to more effectively capture inter-feature relationships. The GNN-enhanced GAN architecture yielded improved predictive performance and stronger correlation with measured data compared to conventional methods, Fig.12. Nonetheless, the authors noted persistent challenges in GNN-based data optimization, emphasizing the critical role of high-quality data preprocessing and representation in the application of machine learning to complex marine engineering problems.

9. Conclusions

The simulation of vessel performance in ice-infested waters has seen significant advancements through both physics-based numerical methods and data-driven machine learning (ML) approaches. Traditional numerical models, grounded in physical principles of ice mechanics and hydrodynamics, have matured to offer reliable predictions of ice resistance, ship-ice interaction forces, and maneuvering behavior. However, these methods often require extensive computational resources, detailed ice field characterization, and can be limited by model assumptions and simplifications.

In parallel, machine learning has emerged as a powerful complementary tool, offering the ability to learn complex nonlinear relationships from empirical data with reduced computational overhead. Recent developments demonstrate the successful application of artificial neural networks, deep learning architectures, and other ML algorithms in predicting ice resistance, propulsion power, and ship-ice interaction dynamics with promising accuracy. These models have been trained on diverse datasets, including model (ice tank) experiments, full-scale trials, and historical records, enabling scalable and adaptive prediction capabilities.

Moreover, the integration of ML techniques into autonomous navigation systems and digital twins holds potential for real-time decision support, performance optimization, and risk mitigation in operations. However, challenges remain in terms of data availability, model generalization across varying ice conditions and hull forms, and the interpretability of ML models.

Future research should focus on hybrid modeling frameworks that combine the strengths of physics-based simulations with data-driven learning, supported by high-quality, standardized datasets. Such integrated approaches are expected to enhance the fidelity, efficiency, and operational relevance of vessel simulations in ice-covered waters, thereby contributing to safer and more efficient Arctic and Antarctic maritime operations.

References

- ARCTIC COUNCIL (2024), *Arctic Shipping on the Rise: What Trends Can Tell Us*, <https://arctic-council.org/news/arctic-shipping-what-trends-can-tell-us/>
- BEHNEN, J.; EHLERS, S.; HISETTE, Q.; VON BOCK, F.; POLACH, R. (2020), *A Discrete element model for brash ice simulations*, 39th Int. Conf. Ocean Offshore and Arctic Engineering
- CABRERA, I. (2017), *Smoothed particle hydrodynamics modeling of brash ice*, Master Thesis, University of Rostock
- CHEN, C.; SHI, W.; SHEN, Y.; CHEN, J.; ZHANG, A. (2022), *A multi-resolution SPH-FEM method for fluid–structure interactions*, Comput. Methods Appl. Mech. Eng. 401
- CHO, S.; JEONG, S.; LEE, S.; KANG, K. (2014), *Development of a Prediction Formula for Ship Resistance in Level Ice*, 33rd Int. Conf. Ocean, Offshore and Arctic Eng. 10
- CUNDALL, P.; STRACK, O. (1979), *A Discrete Numerical Model for Granular Assemblies*, Geotechnique 29, pp. 47-65
- DALEY, C. (1991), *Ice Edge Contact - A Brittle Failure Process Model*, Acta Polytechnica Scandinavica, Mech. Eng. Series No. 100
- ERCEG, S.; EHLERS, S. (2017), *Semi-empirical level ice resistance prediction methods*, Ship Technology Research 64/1, pp.1–14
- FU, S.; ZHANG, D.; MONTEWKA, J.; YAN, X.; ZIO, E. (2016), *Towards a probabilistic model for predicting ship besetting in ice in Arctic waters*, Reliability Eng. and System Safety 155, pp.124–136
- GREISMAN, P. (1981), *Brash Ice Behavior*, U.S. Coast Guard Research and Development Center: Report No. USCG-D-30-81
- HSVA (2012), *Brash Ice Tests for a Panmax Bulker with Ice Class 1B*, Report, HSVA, Hamburg
- HUANG, L.; TUHKURI, J.; IGREC, B.; LI, M.; STAGONAS, D.; TOFFO, A.; CARDIFF, P.; THOMAS, G. (2020), *Ship resistance when operating in floating ice floes: A combined CFD&DEM approach*, Marine Structures 74
- HUANG, Y.; SUN, C.; SUN, J. (2024), *Two-dimensional numerical method for predicting the resistance of ships in pack ice: development and validation*, J. Marine Science and Eng. 12
- JEONG, S.; KYUNGSIK, C.; KUK-JIN, K.; HA, J. (2017), *Prediction of ship resistance in level ice based on empirical approach*, Int. J. Naval Arch. and Ocean Eng. 9/6, pp. 613-623
- JIANG, Y.; WANG, Y.; ZHANG, H. (2021), *Current challenges and future directions in ice navigation research*, Polar Science 27
- KÄMÄRÄINEN, J. (1993), *Evaluation of ship ice resistance calculation methods*, Licenciate Thesis, Helsinki University of Technology

- KÄMÄRÄINEN, J. (2007), *Theoretical investigation on the effect of fluid flow between the hull of a ship and ice floes on ice resistance in level ice*, PhD thesis, Helsinki University of Technology
- KEINONEN, A.; BROWNE, R.; REVILL, C. (1991), *Icebreaker Design Synthesis – Phase 2 – Analysis of Contemporary Icebreaker Performance*, Report for Transportation Development Centre, Transport Canada, TP 10923E
- KEINONEN, A.; BROWNE, R.; REVILL, C.; REYNOLDS, A. (1991), *Icebreaker Characteristic Synthesis*, Report for Transportation Development Centre, Transport Canada, TP 12812E
- KIM, J.; KIM, Y.; KIM, H.; JEONG, S. (2019), *Numerical simulation of ice impacts on ship hulls in broken ice fields*, Ocean Eng. 182, pp.211-221
- KONNO, A.; MIZUKI, T. (2006), *Numerical Simulation of pre-sawn ice test of model icebreaker using physically based monitoring*, 18th IAHR Int. Symp. on Ice, Vol. 2, Sapporo, pp.17-23
- KONNO, A.; YASUKI, N.; YAMAMOTO, K.; WAKO, D.; TAKIMOTO, T.; IZUMIYAMA, K. (2007), *On the numerical analysis of flow around ice piece moving near icebreaker hull. Third report: comparison of simulation results with experimental results under pre-sawn ice condition*, 22nd Int. Symp. Okhotsk Sea and Sea Ice, Mombetsu, pp.29–32
- KONNO, A. (2009), *Numerical simulation of ship navigation in brash ice channels. Second report: effect of size and layer number of ice pieces*, 24th Int. Symp. Okhotsk Sea and Sea Ice, Mombetsu, pp.104-107
- KONNO, A.; SAITOH, O.; WATANABE, Y. (2011), *Numerical investigation of effect of channel condition against ship resistance in brash ice channels*, 21st Int. Conf. Port and Ocean Eng. under Arctic Conditions (POAC), Montreal
- KRAULIN, E.; KARULINA, M. (2011), *Numerical and physical simulations of moored tanker behaviour*, Ships Offshore Structures 6, pp.179–184
- KUUTTI, J.; KOLARI, K.; MARJAVAARA, P. (2013), *Simulation of ice crushing experiments with cohesive surface methodology*, Cold Regions Science and Technology 92, pp.17-28
- LAU, M.; SIMÕES RÉ, A. (2006), *Performance of survival craft in ice environments*, 7th Int. Conf. Performance of Ships and Structures in Ice (ICETECH), Banff
- LAWRENCE, K. (2009), *Load prediction for a moored conical drillship in level unbroken ice: a discrete element and experimental investigation*, PhD Thesis, University of Waterloo
- LI, F.; KORGESAR, M.; KUJALA, P. GOERLANDT, F. (2020), *Finite element based meta-modeling of ship-ice interaction at shoulder and midship areas for ship performance simulation*, Marine Structures 71
- LI, F.; HUANG, L. (2022), *A Review of Computational Simulation Methods for a Ship Advancing in Broken Ice*, J. Mar. Sci. Eng. 10(2)
- LINDQVIST, G. (1989), *A straightforward method for calculation of ice resistance of ships*, 10th Int. Conf. Port and Ocean Eng. under Arctic Conditions (POAC), pp.722–735
- LIU, Z.; YIN, Y.; ZHANG, X.; SUN, T.; WANG, S.; QIAN, X. (2025), *Numerical simulation of ship straight-line navigation in a pre-generated brash ice channel*, Ocean Eng. 339

- LØSET, S. (1994), *Discrete element modelling of a broken ice field - Part I: model development*, Cold Regions Science and Technology 22, pp.339-347
- LUO, W.; JIANG, D.; WU, T.; GUO, C.; WANG, C.; DENG, R.; DAI, S. (2020), *Numerical simulation of an ice-strengthened bulk carrier in brash ice channel*, Ocean Eng. 196
- MATALA, R.; SUOMINEN, M. (2023), *Scaling principles for model testing in old brash ice channel*, Cold Regions Science and Technology 210
- MILAKOVIĆ, A.; FANG, L.; MOHAMED, M.; SÖREN, E. (2020), *A machine learning-based method for simulation of ship speed profile in a complex ice field*, Ships and Offshore Structures 15, pp. 974-980
- MUELLER, A.; ETTEMA, R. (1984), *Dynamic Response of an Icebreaker Hull to Ice Breaking*. Final Report, Iowa Institute of Hydraulic Research, Iowa City
- MYLAND, D.; EHLERS, S. (2019), *Investigation on semi-empirical coefficients and exponents of a resistance prediction method for ships sailing ahead in level ice*, Ships and Offshore Structures, 14, pp.161–170
- NSIDC (2021), *Arctic Sea ice extent*, National Snow and Ice Data Center, Boulder, <https://nsidc.org/arcticseaicenews>
- PRASANNA, M.; HISETTE, Q. (2018), *Discrete Element Simulation of Ships Navigating Through Brash Ice Channels*, 50th OTC Arctic Technology Conf., Houston
- RISKA, K.; WILHELMSON, M.; ENGLUND, K.; LEIVISKÄ, E. (1997), *Performance of merchant vessels in ice in the Baltic*, Helsinki University of Technology
- SPENCER, J.; JONES, J. (2001), *Model-scale/full-scale correlation in open water and ice for Canadian Coast Guard “R-class” icebreakers*, J. Ship Res. 4, pp.249-261
- SUN, Q.; ZHANG, M.; ZHOU, K.; GARME, K.; BURMAN, M. (2022), *A machine learning-based method for prediction of ship performance in ice: Part I. ice resistance*, Marine Structures 83
- SUN, Q.; CHEN, J.; ZHOU, L.; DING, S.; HAN, S. (2024), *A study on ice resistance prediction based on deep learning data generation method*, Ocean Eng. 301
- THOMAS, G.; KUHL, M., (2019), *Computational approaches to modeling ship interaction with ice*, Cold Regions Science and Technology, 167, pp.102-110
- TUHKURI, J.; POLOJÄRVI, A. (2018), *A review of discrete element simulation of ice–structure interaction*, Phil. Trans. R. Soc. A376
- VEERAMANI, M.; PERERA, P.; LI, T., (2020), *Advances in numerical methods for ship-ice interactions*, J. Marine Science and Eng. 8/2, pp.101
- VROEGRIJK, E. (2015), *Validation of CFD+ DEM against Measured Data*, Int. Conf. Offshore Mechanics and Arctic Eng., New York
- WANG, C.; HU, X.; TIAN, T.; GUO, C.; WANG, C. (2020), *Numerical simulation of ice loads on a ship in broken ice fields using an elastic ice model*, Int. J. Naval Arch. & Ocean Eng. 12, pp. 414-427
- WANG, J.; DERRADJI-AOUAT, A. (2011), *Numerical assessment for stationary structure (Kulluk) in moving broken ice*, 21st Conf. Port and Ocean Eng. under Arctic Conditions (POAC), Montréal

- XIE, C.; ZHOU, L.; LU, M.; DING, S.; ZHOU, X. (2023), *Numerical Simulation Study on Ship–Ship Interference in Formation Navigation in Full-Scale Brash Ice Channels*, J. Mar. Sci. Eng. 11
- YAMAGUCHI, H.; TOYODA, M.; NAKAYAMA, H.; RHEEM, C.K.; MATSUZAWA, T.; KATO, H.; KATO, K.; ADACHI, M. (1997), *Influence of floe shape on behaviour of ice floes around a structure*, 16th Int. Conf. Offshore Mechanics and Arctic Eng. (OMAE), Yokohama
- YANG, B.; ZHANG, G.; RAO, H.; WANG, S.; YANG, B.; SUN, Z. (2024), *Numerical simulation of the manoeuvring performance of ships in broken ice area*, Ocean Eng. 294
- XU, P.; CHEN, B.; GUO, Y.; WANG, H. (2024), *Numerical Simulation Study on Ice–Water–Ship Interaction Based on FEM-SPH Adaptive Coupling Algorithm*, Water 16
- ZHAN, D.; AGAR, D.; HE, M.; SPENCER, D.; MOLYNEUX, D. (2010), *Numerical simulation of ship manoeuvring in pack ice*, 30th Int. Conf. Offshore Mechanics and Arctic Eng. (OMAE), Shanghai
- ZHAN, D.; MOLYNEUX, D. (2012), *3-dimensional numerical simulation of ship motion in pack ice*, 31st Int. Conf. Offshore Mechanics and Arctic Eng. (OMAE), Rio de Janeiro
- ZHANG, B.; ZHAO, J.; WU, C. (2024) *Numerical simulation study on the ice load and ship motion response in formation navigation in brash ice*, 43rd Int. Conf. Ocean, Offshore and Arctic Eng., Singapore
- ZHANG, J.; ZHANG, Y.; SHANG, Y.; JIN, Q.; ZHANG, L. (2021), *CFD-DEM based full-scale ship-ice interaction research under FSICR ice condition in restricted brash ice channel*, Cold Regions Science and Technology 194
- ZHONG, K.; NI, B.; LI, Z.; XU, X.; WEI, H.; LI, H.; XUE, Y. (2023), *Direct measurements and CFD simulations on ice-induced hull pressure of a ship in floe ice fields*, Ocean Eng. 272
- ZHONG, K.; LI, Z.; NI, B.; XUE, Y. (2024), *Numerical Studies on ice-induced loads on ship hulls in ice floe regions*, 43rd Int. Conf. Ocean, Offshore and Arctic Eng., Singapore
- ZHONG, N.; POTENZA, A.; SMITH, S. (2024), *Autonomous Navigation in Ice-Covered Waters with Learned Predictions on Ship-Ice Interactions*, arXiv preprint arXiv:2409.11326
- ZHOU, L.; SUN, Q.; HAN, S.; WANG, A. (2023), *A Machine-Learning-Based Method for Ship Propulsion Power Prediction in Ice*, J. Marine Science and Eng. 11
- ZOU, M.; ZOU, Z.; ZOU, L.; CHEN, C.; ZHANG, X. (2024), *Numerical investigations of the ice force distributions on a ship in brash ice channel using a simplified CFD-DEM solving framework*, Ocean Eng. 312

Enabling Graph-Based Knowledge Links for the Shipbuilding Industry

Anna Wagner, PROSTEP, Darmstadt/Germany, anna.wagner@prostep.com

Mirko Theiss, PROSTEP, Darmstadt/Germany, mirko.theiss@prostep.com

Johannes Lützenberger, PROSTEP, Darmstadt/Germany, johannes.luetzenberger@prostep.com

Carsten Zerbst, PROSTEP, Hamburg/Germany, carsten.zerbst@prostep.com

Abstract

Changes continuously challenge the design and manufacturing of vessels. A seemingly small change can have a huge impact, leading to a cost overrun or delay. Realising the scope, quantity and quality of that impact is no small task, as there is often no formal framework to answer the question. In this article, we propose using a knowledge graph to address this problem, allowing users to create and evaluate dependencies even if the original software vendor never envisaged them. We then present an implementation based on readily available PROSTEP components.

1. Introduction

The design, construction and operation of a vessel generate a large amount of information that does not reside in a single place but is distributed across many departments. Basic and detailed design, class approval, production engineering, and procurement all have an important role in shipbuilding. From the IT perspective, each of those departments uses specialised tools, ranging from requirement documents and 2D P&ID diagrams to 3D CAD models, general PLM or specific manufacturing systems and ERP. Each tool has authority for its own part of the product and impacts others, but the relationships between these parts are rarely captured. Over the last few decades, shipyards have matured in core design, but requirements management remains a different matter: many requirements never exist as structured, machine-readable data. The statutory and class requirements issued by classification societies, as well as the contractual specifications agreed with the customer, are typically written in text form like rule books, regulations and contracts. Translating these textual requirements to design rules and relating them to a system and line in a P&I diagram, an item in the 3D model, or an operation in the manufacturing plan is a manual process. For normal laws, initiatives like SPRIN-D, <https://www.sprind.org/en/actions/strategic-projects/law-as-code>, investigate law-as-code, but we know of no similar movements for statutory requirements.

Understanding the dependencies within a project requires a linked data model between the different artefacts; establishing and maintaining these links weaves threads from requirement through design to production. In case of changing requirements or a problem occurring, e.g. in the procurement process, these threads allow us to quickly judge which parts, systems and downstream activities are affected, and thus evaluate the real scope, cost and schedule impact of the change. Tracing the information without such a thread is a cumbersome process. As a result, a seemingly small change can travel unnoticed all the way down through the delivery chain and cause a cost overrun or a delay.

The idea of a digital thread linking related artefacts across tools and disciplines is not new, and several approaches have been proposed. The SEUS project, for example, set out to bridge the digitalisation gap in European shipbuilding by integrating established CAD, CAE, CAM and PDM tools into a single smart-shipbuilding platform *Gaspar et al., (2023)*. We implemented a dedicated tool which requires no change to the existing IT infrastructure. It links information contained in the IT systems of different domains and supports change management across domain boundaries. All these approaches share the insight that artefacts no longer exist in isolation: mechanical, electrical and software components are developed in separate domains and IT systems, and their dependencies are often reconstructed by hand at project milestones. A broader market shift reinforces the urgency of this problem. Industry analysts and major engineering software vendors increasingly identify the strategic differentiation in the PLM space as moving away from the capabilities of individual systems of record towards the quality of the platform that connects information across many systems and

makes it accessible to AI. This paper presents a way to manage the relationships between data held in the different departments and tools of a yard, without forcing that data out of its source system. We also examine what can currently be achieved when unstructured content, such as requirements written as prose, has to be interpreted outside the design tools or the PLM system.

The specific contribution of this paper is a federated Digital Thread architecture that (1) preserves existing engineering systems of record by connecting them semantically through a knowledge graph rather than migrating data, (2) builds the graph automatically as a by-product of ongoing integrations via event-driven connectors, and (3) employs AI agents both to keep graph relationships current and to assist engineers in navigating information across heterogeneous tool landscapes. In architectural terms, contribution (1) separates engineering data ownership from relationship management: source systems retain sole authority over their own data, while a dedicated store governs the cross-domain semantics that connect them. Such separation was already explored outside shipbuilding, *Kwon et al. (2019)* and, more recently, in model versioning for systems engineering, *Wu et al. (2025)*. Distributed ship design and production processes have themselves been the subject of an information integration platform presented at this conference in earlier work, *Bronsart et al. (2015)*; the present paper extends that line of work with the semantic and AI-oriented layer described in Sections 2 and 4.

The research question addressed here is to what extent a federated knowledge graph can provide scalable cross-domain traceability and change-impact analysis for naval and offshore programmes. It is built automatically as a by-product of integrations a yard already runs rather than through a dedicated modelling effort, and without the need to centralise all engineering data in a single repository. The implementation is demonstrated using PROSTEP TRACE and PROSTEP INTEGRATE, including maritime-specific connectors.

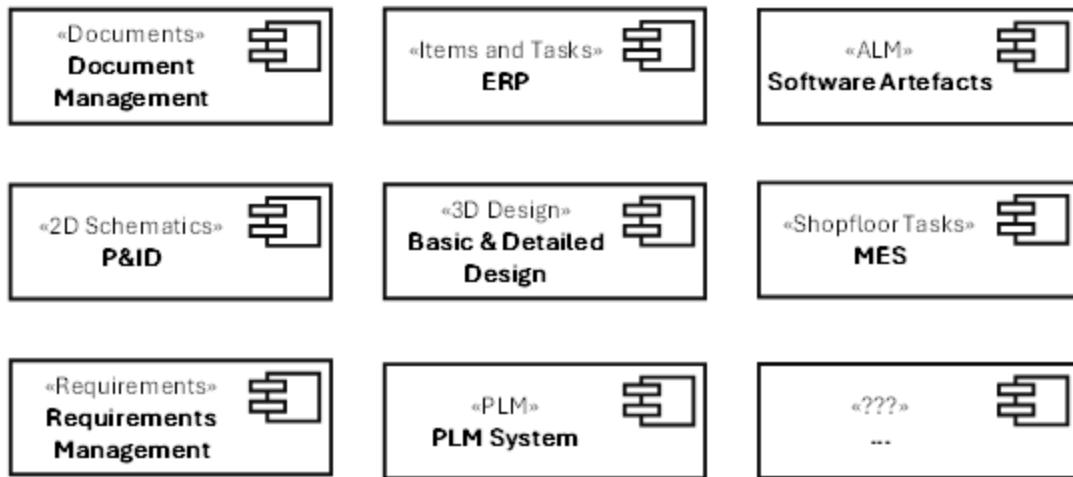


Fig.1: An Example Deployment Diagram on a Shipyards representing the data silos

2. Information as a Graph

A knowledge graph represents information as a network rather than as records in tables or documents. It is a heterogeneous graph whose nodes represent entities, such as a requirement, a pump, a test report or a production order, and whose edges are typed, directed relations between them, for example, is satisfied by, is part of or verifies *Guo et al., (2020)*. All objects are related as subject-predicate-object tuples; many such statements sharing common nodes represent a large body of knowledge. Two storage paradigms are common. The Resource Description Framework (RDF) models knowledge as triples taken from a formally defined vocabulary, while the property graph attaches key-value properties directly to nodes and edges. Both share the main advantage of the graph form: relationships are stored explicitly. A question such as which components are affected by a given requirement then becomes a traversal of edges, and traversals over several relations, so-called multi-hop queries, can be evaluated efficiently. As the path itself is explicit, it can also be shown to

the user as an explanation of the result. Unlike a relational schema, which has to be designed up front, a graph can be extended by simply adding new node and edge types as new kinds of knowledge appear. This suits the heterogeneous and changing data of a shipbuilding project. A note on terminology: knowledge graph denotes the general data structure just described, while Knowledge Base refers to the specific, persisted instance of it maintained by the platform introduced in Section 4 – concept to implementation, not synonyms.

Much of the engineering knowledge that should populate such a graph does not start in a structured form. Requirements, rule books and meeting minutes are written as prose, and the entities and relations they describe are hidden in natural language. Turning this content into nodes and edges is the task of an extraction pipeline. First, named-entity recognition finds the spans of text that refer to entities. These mentions are then disambiguated and linked to a single node, so that ‘the main fire pump’, ‘FP-01’ and ‘the pump in frame 42’ resolve to one entity instead of three. Finally, relation extraction derives the predicates that connect them, *Schuhmacher and Ponzetto (2014)*. By linking the entities found in a text to a reference knowledge base and weighting the semantic relations between them, a free-text document becomes a knowledge-rich graph suitable for comparison and retrieval. As such, extraction is never perfect; the derived relations are usually kept with a confidence value and offered to a domain expert for confirmation rather than treated as fact.

The third challenge is that the relevant knowledge is spread over many source systems, each with its own identifiers and terminology, IT interfaces or APIs. To overcome this technical hurdle, our implementation supports proxy objects, lightweight nodes that stand in for the real artefacts in the PLM, ALM, ERP or CAD tools and carry just enough information to identify the source object and its relations. The authoritative data stays in its source system, and the graph only stores the connections. Linking then reduces to entity resolution, that is, deciding when a proxy in one system denotes the same real-world thing as a proxy in another. The semantic-similarity and graph-matching techniques developed for document modelling can be applied directly to this task. An ontology layer reconciles the different vocabularies, so that a ‘part’ in one tool and a ‘component’ in another are recognised as the same concept.

This graph-centric view can be compared with link-based interoperability standards such as the Open Services for Lifecycle Collaboration (OSLC), which defines how tools expose and cross-reference resources over the web using different vocabularies for individual domains, such as Change Management. OSLC keeps each link inside the pair of tools that created it. The knowledge-graph approach instead collects all relationships in one dedicated store, where they can be queried, traversed and analysed independently of any single tool. This distinction has two practical consequences. First, performance: answering a multi-hop question such as “which production orders are affected by this changed requirement?” through OSLC requires chaining REST calls across multiple tool pairs in sequence, with latency accumulating at each hop. The same query over a property graph is a single traversal in the graph database, which in general is orders of magnitude faster compared to parallel retrievals, e.g. in the PLM and ERP system.

A recent topic is how to provide a knowledge graph for efficient AI usage. Large language model agents and graph-based reasoning systems can consume an explicit relationship store directly and traverse typed edges to build context information. Point-to-point information, as e.g. represented by OSLC links, is often scattered across several tool interfaces, requiring additional orchestration logic before it can be aggregated into the context an AI agent needs. The Knowledge Base therefore serves a dual purpose: it makes the Digital Thread queryable for engineers and consumable for AI agents from a single, coherent data structure. Fig.2 shows a small knowledge graph of this kind, with entities from several domains connected by typed relations. Table I summarises this comparison across the storage paradigms discussed above and the approach used in Section 4. For this reason, we chose a property graph as the primary store, rather than a pure RDF triple store, because engineering impact analysis primarily requires high-performance graph traversals, whereas the semantic interoperability that RDF and OWL are strong at is provided separately through the ontology layer described in Section 4.4; this way neither requirement compromises the other.

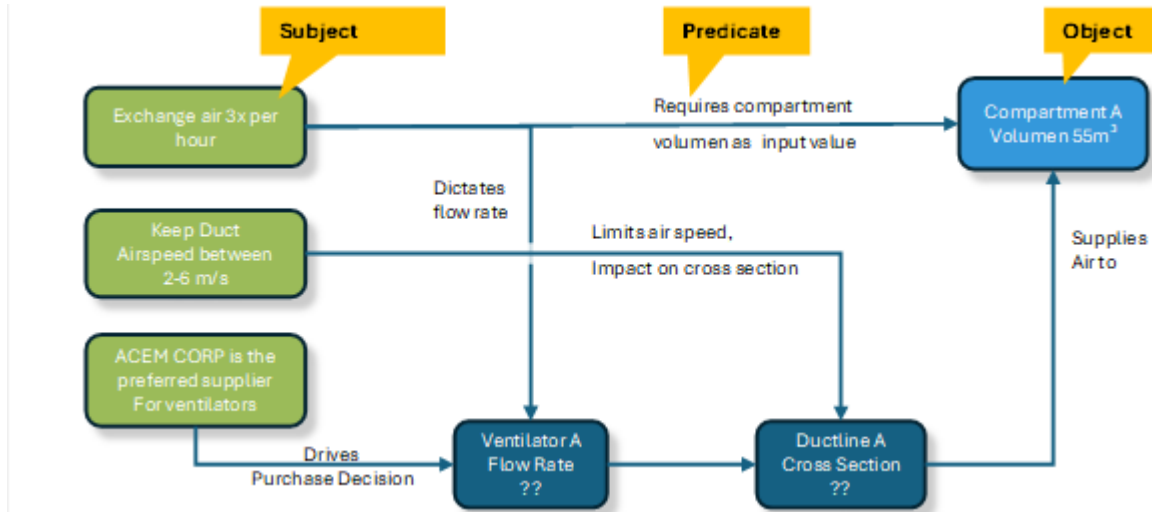


Fig.2: Knowledge Graph Example

Three further architectural alternatives deserve explicit comment. An event store, which records the sequence of changes rather than the current relationship state, was not chosen as the primary structure because change-impact queries need the current graph of dependencies rather than a replay of history; event streams remain useful as the input from which the graph is derived (Section 4.4), not as the query target itself. Federating multiple independent graph databases, rather than maintaining one logical Knowledge Base per tenant, was also considered and set aside for the present scope, since query federation across graph engines adds latency and consistency-management overhead that a single logical store avoids; the source systems themselves remain federated in the sense that their data is never centralised. Finally, a full digital twin, a live, simulatable replica of a vessel's behaviour, addresses a different problem: it is a run-time behavioural model, whereas the Knowledge Base is a structural and relational record of engineering artefacts and their dependencies. The two are complementary rather than competing, and a digital twin could in principle consume the Knowledge Base as one of its data sources. The remainder of the paper applies these ideas to the items, relationships and source systems of the shipbuilding industry.

Table I: Comparison of different API and storage approaches vs. access requirements

Approach	Multi-hop traversal performance	Semantic expressiveness	AI/agent readiness
OSLC (point-to-point links)	Sequential REST calls per hop; latency accumulates and depends on source-system availability.	Domain-specific vocabulary per link; no shared cross-domain semantics	Requires custom orchestration to aggregate context for an agent
RDF / triple store	Single traversal; formally sound, but SPARQL over triples is comparatively slow at scale	High; ontology-based, standardised vocabulary	Queryable by agents, but query complexity limits adoption
Property graph	Single traversal; index-free adjacency, fast at scale	Moderate; schema-flexible node and edge properties	Good; native fit for graph-traversal-based agent reasoning
PLM System	Sequential REST calls per hop; latency accumulates and depends on source-system availability.	Depends on the PLM customising	Depends on the PLM feature set and customising
federated knowledge graph	Single traversal over a property-graph core	Ontology layer (Section 4.4) adds RDF-based semantics on top of the property graph.	High; explicit, typed relationship store consumable directly by LLM agents

3. Application in the Shipbuilding Industry

A knowledge graph in the shipbuilding industry draws on various sources. At the start are the contractual requirements agreed with the customer and the statutory requirements from the classification society and the flag state. These are accompanied by the shipyard's own standards and best practices. They are often driven by production needs rather than by the design itself, for example, a preferred plate thickness, standard pipe diameters or an approved equipment maker. Typical configuration items handled this way span hull sections and production blocks in structural design, compartment and outfitting arrangements, pipe systems and HVAC routing, and cable routing lists, together with the class approval documentation that must reference all of them. The design driven by these inputs is then the next layer. From an IT perspective, all of them are located in different tools: contracts and rules in a document management system, individual requirements in a requirements management tool, the functional systems of the vessel in the P&ID tool, the geometry in the detailed design system, and the parts and manufacturing steps in manufacturing preparation. Only part of the information can be traced directly; this is often only possible between the as-designed and as-planned structures.

Consider a change introduced late by the customer, for example, an upgraded pump that no longer fits the reserved space. To judge whether such a change is cheap or expensive, the yard has to evaluate its impact, and that means collecting everything that depends on the affected requirement: the systems in the P&ID, the components and routing in the 3D model, the affected parts, and the purchase orders already placed for them. A knowledge graph turns this into a single traversal from the changed requirement to its connected parts, instead of a manual search through several disconnected tools. The same query also reveals breaks in the delivery chain, where the purchase department has already ordered a part whose requirement has changed since then.

Concretely, the traversal follows a single chain of typed relationships: the changed customer requirement links to the functional system represented in the P&ID, which links to the affected component and routing in the 3D model, which links to the corresponding part and operation in the production plan, which links to the purchase order already raised against it. Querying this chain once (requirement to P&ID to 3D model to production to purchase order) returns the full impact in a single step, in place of the sequential, manual lookups a planner would otherwise perform in each of the four source systems.

The impact often reaches parts that are already in manufacturing. Here it is not enough to know that a part exists; the yard needs its current status, whether it has only been nested, already cut, assembled into a panel, or purchased from a supplier. A part that is still a line in a nesting plan can be changed at little cost, while a part that has been cut or welded is going to scrap. By linking design parts to their manufacturing parts and production status in the manufacturing preparation system, the graph shows not only what is affected by a change but also how far each affected item has progressed, which is what really determines the cost. The most important part of such a knowledge graph is not the storage of the links but their automatic creation, because only this overcomes the data silos at the scale of a real project.

In some cases, this is straightforward: where a yard follows consistent naming and numbering rules, the relation between the basic design, its detailed-design parts and the production parts can be derived directly from the identifiers. In most cases, it is not as simple as that, in particular for the link between a requirement written in prose and the design that fulfils it. Several research projects are now working on extracting this kind of information automatically, for instance recognising individual requirements in a contract or rule document, <https://www.sprind.org/en/actions/strategic-projects/law-as-code>, Galva et al. (2026), or Hu et al. (2025).

These approaches are based on AI, and with the large language models now available, it is becoming possible to train and apply them on shipbuilding data to create these links, which until recently had to be drawn by hand. The extraction pipeline proceeds in three stages: named-entity recognition

identifies spans of text referring to engineering artefacts (components, requirements, test procedures); entity disambiguation resolves multiple textual references to the same physical object; and relation extraction derives the predicate connecting two entities, for example “satisfies” or “is verified by”. Because extraction is never perfect, each derived relationship is stored with a confidence score and presented to a domain expert for confirmation before it is treated as authoritative. Relationships confirmed by an engineer are marked as exact; those proposed automatically are marked as machine-suggested. This distinction is preserved throughout the Knowledge Base, so that downstream analyses can weight or filter relationships by their origin.

4. Example Implementation

The presented example implementation combines three tools of PROSTEP’s offering into a Digital Thread Platform: (1) a knowledge-graph-based software that allows users to manage traceability in heterogeneous and distributed environments, (2) connectivity components that enable data integration and migration between different systems from various domains and tool vendors, and (3) an LLM-based AI workbench that performs, e.g. text processing to extract requirements from plain documents. The concepts of Sections 2 and 3 are implementation-agnostic; the tools named below are simply where PROSTEP realises them.

4.1. Managing Traceability

The traceability component (PROSTEP TRACE) implements the knowledge graph described above as a separate application built around a component called the Knowledge Base. It does not copy source data; it stores only the relationships between the configuration items living in the connected tools, for example, PLM, ALM, ERP and the manufacturing systems. Each source system stays the system of record for its own data. This is the proxy-object approach of Section 2: a node in the graph stands in for a real object in a source tool. For performance, it can hold additional metadata per object, so that a pre-query runs locally before in-depth queries are delegated to the source systems.

Every action the traceability component enables could in theory be performed in a single domain tool, but only within that tool’s bounded view. In practice, change impact assessment and requirements coverage cannot be judged from one domain’s data alone. A change request in ALM reveals its true cost and schedule impact only when traced through the PLM structure to the affected manufacturing parts. It provides cross-domain views built from the traces stored in the Knowledge Base.

The ALM-to-PLM link is a recurring example: modifying a part without knowing which requirement governs it risks a specification violation that surfaces only at verification. The component provides customisable views on the graph, for example to trace requirements or follow a change across domains. At object level, the graph shows further information aggregated from the source systems, such as maturity state and responsible person; see Figure 3.

It uses the cross-domain links in the Knowledge Base to establish and visualise coverage: which requirements are addressed by which design elements, which remain unresolved, and which design elements lack a governing requirement. This coverage view underpins class approval documentation and milestone completeness checks. It also precedes change impact analysis, which first requires identifying the upstream requirement governing a component.

The data model of the traceability component is split into two tiers. The first is a generic model that allows building a context for the visualised and analysed information, including processes and collections. The second is a dynamic, ontology-based domain model to classify ‘Configuration Items’ and their ‘Trace Links’, which can be customised per instance and carries the domain knowledge and “semantic sugar”. The data is kept in a graph database, better suited to multi-hop queries than a relational store. A property graph (Neo4j) holds the operational data; an RDF-based semantic layer can be added where domain mappings become too complex.

As discussed in Section 3, the potential of using a traceability tool like PROSTEP TRACE in the shipbuilding environment is enormous, but the initial cost of establishing all traces and links between information objects is high. Mappings from source data models to the dynamic domain model must also be established via connectors handled manually or semi-automatically.

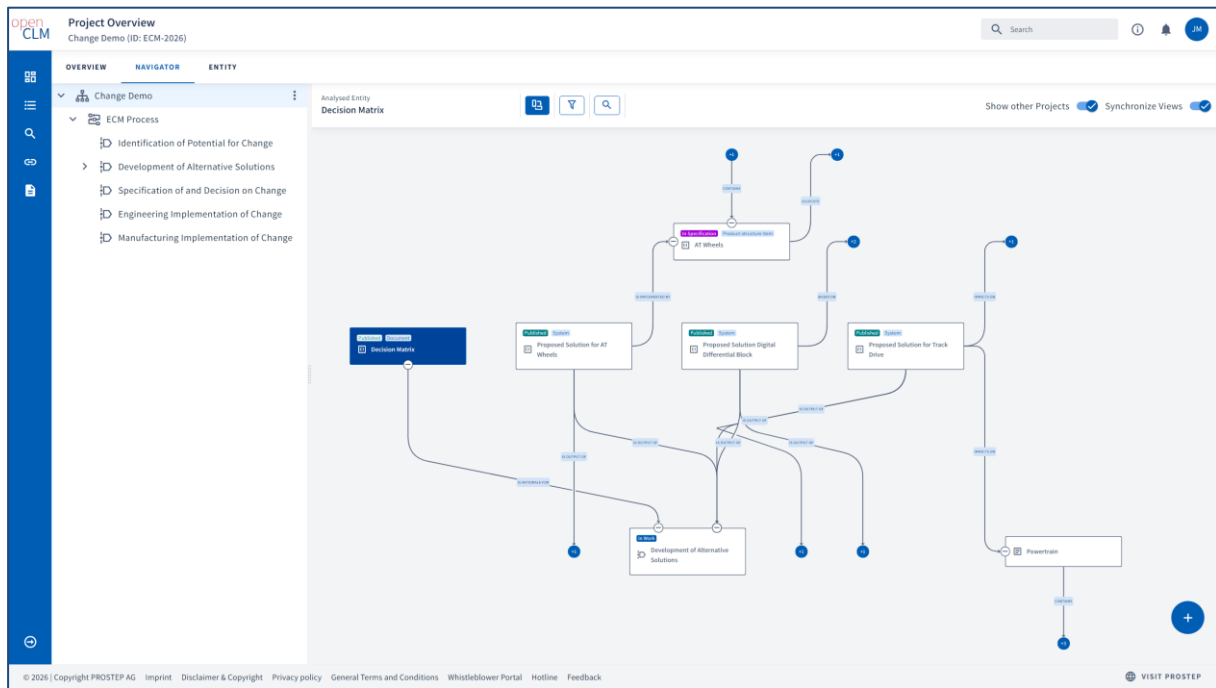


Fig.3: User Interface to Investigate a Change Impact

4.2. Data Integration

When information needs to be shared across different source systems, the integration component (PROSTEP INTEGRATE) is used, which offers a single interface to many engineering tools via more than 30 certified connectors to the major PDM, PLM, ALM, ERP, and requirements management systems used across industries. The INTEGRATE SHIP line adds connectors for maritime design tools, including AVEVA Marine, CADMATIC, FORAN, NAPA, Revit, ShipConstructor, and SM3D. Together they cover the ALM, document and requirement management, PLM and design tools across the lifecycle, so items and relations can be read into the graph without manual export.

4.3. AI-Based Automated Requirement Detection

As established in Section 2, a significant share of the requirements that govern a shipbuilding project, contractual specifications, classification society rules, and flag state regulations, exists only as prose in documents rather than as structured, machine-readable data. Populating the Knowledge Base with them therefore cannot rely on connector-based synchronisation alone. This is the role of the proposed extraction and retrieval component (the PROSTEP AI Workbench), an LLM-based text processing component that connects unstructured requirement documents to the Knowledge Graph. It operates in two complementary modes, generalised for the platform in Section 4.4.

In its Knowledge Update Agent mode, the tool applies the extraction pipeline described in Section 2, named-entity recognition, entity disambiguation, and relation extraction, to contracts, rule books, and specifications as they are ingested. Detected requirements, the components they reference, and the predicates connecting them (for example, governs, limits, requires) are persisted as proxy nodes and edges in the Knowledge Base. Each derived relationship is marked machine-suggested until an engineer confirms it. Confirmed links then feed the coverage analysis of Section 4.1.

In its Engineering Knowledge Assistant Agent mode, the tool provides semantic retrieval, or Retrieval-Augmented Generation (RAG), extended here with graph-guided retrieval over the Knowledge Base as surveyed for RAG systems in general by *Peng et al. (2024)*, directly over the requirement documents. An engineer's natural-language query is answered by combining this retrieval with a query delegation to the Knowledge Graph: the tool resolves which requirements, components, and prior decisions are already represented as nodes, and supplements this with passages retrieved directly from the source document where no structured link yet exists. The value added here is verbatim traceability: a design decision can be traced back to the exact paragraph of a contract or class rule rather than to a paraphrase, which matters wherever a design choice must hold up under class approval or audit.

4.4. Weaving the Digital Thread

Fig.4 shows an example deployment of the complete digital thread platform in a shipyard's tool landscape. Each source system is coupled via a connector to the integration component, whose process server is connected to the traceability Knowledge Base. The latter provides engineers and project managers with an interface to query and analyse the data. AI agents reside on Connectors and the Knowledge Base to automatically provide services to users and keep data consistent. Using the Knowledge Base centrally lets the integration component reuse the ontology-driven vocabulary in alignment with the traceability component. The alignment is based on the prostep ivip's Top Level Ontology, which defines a common set of concepts for product lifecycle data, and is extended by the CASCARA ontology, <https://cascara.gfse.org>, *Bajaj et al. (2016)*.

CASCARA captures the configuration items and dependency types that appear across PLM, ALM, ERP and manufacturing environments. Together, they map tool-specific terms - such as "part" in PLM and "component" in CAD - to the same concept, enabling traversal across domain boundaries without manual schema mapping per tool pair. This graph-based approach to a domain model shared across systems engineering tools is consistent with the broader move toward graph-based digital blueprints for model-based systems engineering *Bajaj et al., (2016)*.

On the other hand, filling the Knowledge Base can be enhanced via the established integrations: whenever the integration component synchronises an object between two systems, it also records the matching nodes and relationships in the Knowledge Base, so the graph is built as a by-product of the integration that the yard runs anyway, without separate modelling projects. Unlike a classical integration bus, the relationships survive the transfer.

However, a synchronisation event captures the state of objects at that moment. Engineering work continues after synchronisation: new versions are created, objects reclassified, and structures reorganised. Keeping the Knowledge Base current as this ongoing work proceeds is the first role of AI agents in the platform. An agent monitors changes in connected systems and updates or proposes updates to the affected nodes and relationships. Links from a synchronisation event are marked as exact; those proposed by an agent are machine-suggested and confirmed by an engineer before becoming authoritative. On this basis, a user can follow a change from a requirement to the affected components, test results and production orders in a single view. As each connector reads and writes, the integration component can also write derived information back to the source systems. Since the traceability component defines authoritative sources and orchestrates processes such as a Global Change Process, the platform can trigger domain-specific release and change processes in the source systems.

The second role of AI agents in the platform addresses a different and equally significant productivity cost: the time engineers spend searching for information across distributed systems. Finding a current document version, the requirement governing a component or the status of a change request usually means navigating several tool interfaces. An AI agent with access to the Knowledge Base can answer such queries directly, because the graph already encodes the relationships between the relevant objects. The Knowledge Base thus grounds the agent in linked engineering data rather than general

training knowledge, which matters where precision and traceability are required for certification. Where the information is not yet in the Knowledge Base, the domain-model mapping tells the agent which connectors to query. Lastly, the AI Workbench adds unstructured data such as requirement documents to the Knowledge Base. Queries can be delegated to it, and results are stored for use in other queries or traceability analyses.

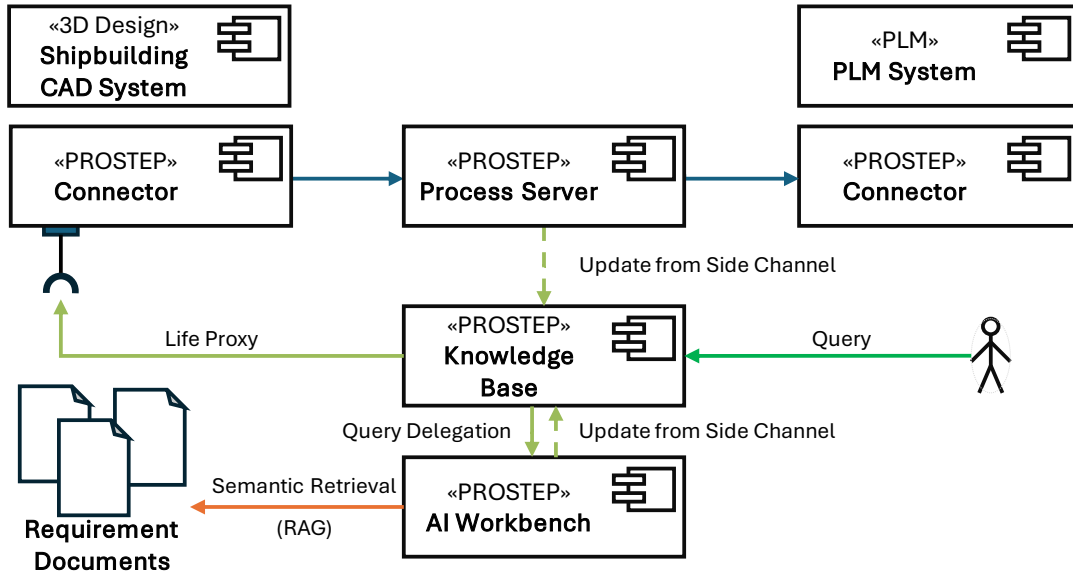


Fig.4: Example Deployment Diagram

4.5. Evaluation Methodology

The evaluation presented so far is qualitative, and this is the most significant limitation of the present paper. This section states the metrics that will support a quantitative study. Two categories of measurement are planned for the validation study announced in Section 5. Traversal and coverage metrics from the Knowledge Base comprise the number of connected source systems, proxy nodes and typed relationships per domain, the requirements-coverage ratio, and multi-hop query latency against the sequential OSLC baseline of Section 2. Extraction-quality metrics from the AI Workbench comprise precision and recall of the named-entity-recognition and relation-extraction stages against an annotated sample of contract and class-rule documents, and the share of machine-suggested relationships later confirmed by an engineer.

The scenario in Section 3 illustrates the qualitative benefit: a single graph traversal replaces several manual lookups across the four systems involved. Quantifying it is part of the planned evaluation announced in Section 5.

5. Summary

The information enabling a shipyard to build vessels is spread over many different departments and tools. In the worst case, each capability needed in a shipyard is implemented by a different system (see *Wölke et al. (2026)* for an overview). As a result, judging the impact of change is hard, as the relevant data is spread over many tools and departments. Furthermore, some information, like requirements, exists only as prose. In this paper, we have argued that a knowledge graph, kept in a dedicated system that stores only the relationships between the objects in the source tools, is a practical way to make these dependencies visible. The source systems keep their role as the system of record, while the graph provides the traceable thread from a requirement through the design to the parts in production. AI agents contribute to the platform in two complementary ways: they keep the Knowledge Base current as engineering work evolves in connected systems, marking automatically derived relationships as machine-suggested until confirmed by an engineer; and they assist engineers

in retrieving relevant information across domain boundaries, using the grounded, linked knowledge in the graph rather than reasoning from general knowledge alone.

We have shown how this can be implemented based on readily available components. More than 30 connectors cover the major PLM, ALM, ERP systems and maritime design environments. This gives access to most of the existing shipbuilding systems and design tools and populates the graph as a by-product of the integrations a yard already runs. The decisive part is not storing the links but creating them automatically. Where naming and numbering rules exist, this is straightforward; for requirements written in prose, it is not, and this is where the AI-based extraction methods discussed above, together with the large language models now available, are beginning to make a real difference.

This results in a holistic Digital Thread Platform that leverages already existing and implemented solutions to provide further benefits to users. At the same time, known shortcomings and high complexity can be reduced. Combined with additional AI agents, issues regarding data integrity and completeness can be mitigated as well.

The next steps are to extend the automatic linking to more domains and to validate the approach on real shipbuilding data, so that the impact of a change can be assessed quickly and reliably across the whole delivery chain. A further extension planned beyond the scope of this paper is to extend the federation across company boundaries, connecting an OEM's Knowledge Base to those of its suppliers without centralising their data, mirroring the broader shift toward supplier collaboration spanning the full delivery chain in naval and offshore programmes. The platform described in this paper is currently under active development. Validation on an industrial shipbuilding programme, measuring reductions in manual linking effort, improvement in traceability coverage, and change impact assessment time, is planned as the next step and will be the subject of a follow-on publication. The architecture presented here is the foundation for that validation: a federated Digital Thread that avoids centralised data migration, builds relationship knowledge incrementally, and makes that knowledge accessible both to engineers through cross-domain views and to AI agents through a grounded, linked data context.

References

- BAJAJ, M.; ZWEMER, D.; YNTEMA, R.; PHUNG, A.; KUMAR, A.; DWIVEDI, A.; WAIKAR, M. (2016), *MBSE++ - Foundations for Extended Model-Based Systems Engineering Across System Lifecycle*, INCOSE Int. Symp., Vol.26, pp.2429-2445
- BRONSART, R.; GAU, S.; LUCKAU, D.; SUCHAROWSKI, W. (2015), *Enabling Distributed Ship Design and Production Processes by an Information Integration Platform*, ICCAS Conf., Bremen
- GALVAN, B.; DRAKOULAS, G.; CALS, B.; KUCUKYILMAZ, T.; SCHAKEL, M.; SEMCHENKO, D.; KUZJATKIN, J.; CABRAL, G.P. (2026), *A Multi-Agent LLM Framework with RAG for Shipbuilding Engineering Documents*, 18th Int. Conf. on Agents and Artificial Intelligence, Marbella
- GASPAR, H.M.; SEPPÄLÄ, L.; KOELMAN, H.; GARCIA AGIS, J.J. (2023), *Can European Shipyards be Smarter? A Proposal from the SEUS Project*, 22nd COMPIT Conf., Drübeck
- GUO, Q.; ZHUANG, F.; QIN, C.; ZHU, H.; XIE, X.; XIONG, H.; HE, Q. (2020), *A Survey on Knowledge Graph-Based Recommender Systems*, arXiv:2003.00911
- HU, J.; PENG, C.; SHI, L.; YANG, H.; ZHENG, J. (2025), *Knowledge Graph for Chinese Shipbuilding Standards: Construction Methodology and Practical Application*, IEEE 6th Int. Conf. on Computer, Big Data, Artificial Intelligence (ICCBD+AI), Xiamen
- KWON, S.; MONNIER, L.V.; BARBAU, R.; BERNSTEIN, W.Z. (2019), *Enriching Standards-*

Based Digital Thread by Fusing As-Designed and As-Inspected Data Using Knowledge Graphs, Procedia CIRP 84, pp.88-93

PENG, B.; ZHU, Y.; LIU, Y.; BO, X.; SHI, H.; HONG, C.; ZHANG, Y.; TANG, S. (2024), *Graph Retrieval-Augmented Generation: A Survey*, arXiv:2408.08921

SCHUHMACHER, M.; PONZETTO, S.P. (2014), *Knowledge-based Graph Document Modelling*, 7th ACM Int. Conf. on Web Search and Data Mining (WSDM), New York, pp.543-552

WÖLKE, T.-H.; RATTE, M.; WITTKOP, P.; ZERBST, C. (2026), *PLM in Shipbuilding: A Capability-Based Blueprint*, ICCAS Conf., Singapore

WU, S.; WANG, G.; LU, J.; HUANG, J.; QIAO, J.; YAN, Y.; KIRITSIS, D. (2025), *Cognitive Digital Thread Tool-Chain for Model Versioning in Model-Based Systems Engineering*, Advanced Engineering Informatics 67, Article 103490

A Capability-Based Blueprint for Shipbuilding PLM

Torben-Henning Woelke, PROSTEP, Hamburg/Germany, Torben-Henning.Woelke@PROSTEP.com

Manuel Ratte, PROSTEP, Köln/Germany, Manuel.Ratte@PROSTEP.com

Peter Wittkop, PROSTEP, Köln/Germany, Peter.Wittkop@PROSTEP.com

Carsten Zerbst, PROSTEP, Hamburg/Germany, carsten.zerbst@prostep.com

Abstract

Shipyards increasingly apply Product Lifecycle Management principles (PLM) to manage design and manufacturing-related data over the vessel lifecycle. This helps to handle engineering changes, ensure data consistency, and preserve knowledge. Yet, there are no established practices on how to implement PLM principles in shipbuilding. To help our customers to choose the correct areas for a PLM application, prioritise based on their organisation's needs, and communicate in a consistent manner with PLM vendors, we created a vendor-neutral, capability-based blueprint that describes and categorises PLM for shipbuilding. In this paper, we present this blueprint and illustrate its usage through real-world shipyard examples.

1. Introduction

Shipbuilding is undergoing a significant digital transformation driven by the adoption of advanced digital technologies and the need to remain competitive in a global market *Calvache et al. (2025)*, *Agis et al. (2024)*. This transformation is further intensified by increasing system complexity, particularly due to the growing use of advanced digital engineering tools across the product lifecycle *Papanikolaou et al. (2024)*, as well as by evolving regulatory requirements, especially in the context of sustainability and environmental compliance, *Agis et al. (2024)*. At the same time, shipbuilding projects are characterised by fragmented processes, distributed stakeholders and data repositories. This causes a significant amount of friction, calling for improved integration of data and collaboration across disciplines and organisational boundaries to solve those problems, *Penciuc et al. (2016)*. In this context, Product Lifecycle Management (PLM) has emerged as a key enabler, acting as the backbone for managing and integrating lifecycle data. By establishing a centralised and consistent data environment, PLM supports the concept of a single source of truth and enables collaboration across engineering domains and organisations throughout the entire product lifecycle, *OECD (2025)*.

But introducing PLM concepts into an organisation requires much more than simply installing yet another tool. Educated decisions on what data is needed for which processes, accessible by what stakeholders, will not only yield better results but also greatly reduce the overall effort put into the implementation of PLM. To avoid repeating this conceptual work for each organisation, we prepared a capability-based PLM blueprint for the shipbuilding industry. It is based on decades of experience in introducing PLM solutions and gives a head start compared to starting from scratch, thus reducing the effort and enhancing the achieved result.

2. Challenges of PLM in Shipbuilding

Shipbuilding organisations face a distinct set of challenges when introducing PLM, arising from the industry's engineer-to-order nature, fragmented IT landscapes, and limited established best practices compared to serial manufacturing sectors. This chapter examines these challenges across four dimensions: increasing complexity and digital transformation, fragmented data and system landscapes, cross-disciplinary change management and traceability, and the growing importance of requirements and lifecycle continuity.

Digital transformation is itself the first major challenge shipyards face. Advanced methods – PLM, Digital Twins, simulation-driven design, AI – improve efficiency but also increase engineering complexity and data dependencies, *Papanikolaou et al. (2024)*, *OECD (2025)*. This is pronounced in

shipbuilding, where engineer-to-order vessels are highly customised, producing large volumes of project-specific information. Digitalisation is therefore pursued department by department rather than as a lifecycle strategy, so its benefits are rarely realised, *Calvache et al. (2025)*, *Agis et al. (2024)*. The same pattern appears across manufacturing, where harmonising mechanical, electrical/electronic, and software development has outpaced legacy PLM landscapes, *PROSTEP AG (2023)*.

Shipbuilding projects involve many disciplines. Design, planning, procurement, manufacturing, and commissioning rely on specialised software with different data models, and product information sits in silos. Manual transfers between them create redundancy and inconsistency, *Agis et al. (2024)*. Systems are optimised locally, but the lifecycle view is lacking. As vessels evolve, keeping design data synchronised with production planning, procurement, and supplier deliverables becomes harder, and the resulting silos slow decision-making *Seppälä (2022)*.

Delivering a vessel requires collaboration across naval architecture, structural design, outfitting, piping, electrical design, production, and supply chain, working in parallel, so engineering changes rarely affect a single discipline, *Agis et al. (2024)*. They must be coordinated across the repositories and tools each discipline maintains, raising the risk of inconsistent implementation and incomplete impact assessment. Unlike serial manufacturing, they cannot be deferred to a later run, as design and production overlap. Traceability therefore becomes critical: organisations must see how requirements, design decisions, changes, and production data connect; without it, impact assessment and regulatory compliance become harder, *Papanikolaou et al. (2024)*, *Penciuc et al. (2016)*.

As complexity increases, structured requirements management grows in importance. Projects must reconcile requirements from owners, class societies, regulators, suppliers, and internal engineering; these evolve and must stay connected to design, verification, and production, *Papanikolaou et al. (2024)*, *OECD (2025)*. Yet requirements, models, and production data sit in separate systems, with links incomplete or manual. The result is no consistent digital thread from requirements to operation, making it hard to govern changes and keep business objectives, engineering, and IT systems aligned, *Agis et al. (2024)*.

Together – fragmentation, disconnected systems, cross-disciplinary change coordination, and the missing digital thread – these challenges share a root cause: no structured, vendor-neutral way to connect processes, requirements, lifecycle data, and systems. The following chapter introduces a capability-based PLM blueprint to address it.

The motivation to apply PLM principles is clear, but introducing a system for daily use is not simple. Most shipyards have no previous experience with such an implementation, and departments, project leads, and management often hold different ideas on purpose, priorities, and wording. PLM vendor sales staff addressing different management levels in yet other terms does not help either.

The capability map we introduced with our customers serves multiple purposes:

- Ensure everybody uses the same terminology, even where vendors use different wording, so later offers can be compared like for like
- Use the map as a 150% solution to identify which PLM aspects can be ignored initially and thus fix the scope, countering the feature creep most PLM projects suffer from
- Use the map to prioritise capabilities and define a useful minimum viable product (MVP); starting small has proved to be the only workable route, as projects attempting everything at once tend to fail

Starting from the same capabilities does not produce the same result everywhere: scope reduced to the customer's requirements, existing CAD and ERP tools, and infrastructure always leads to different implementations. The map keeps every yard on course, not at the same port.

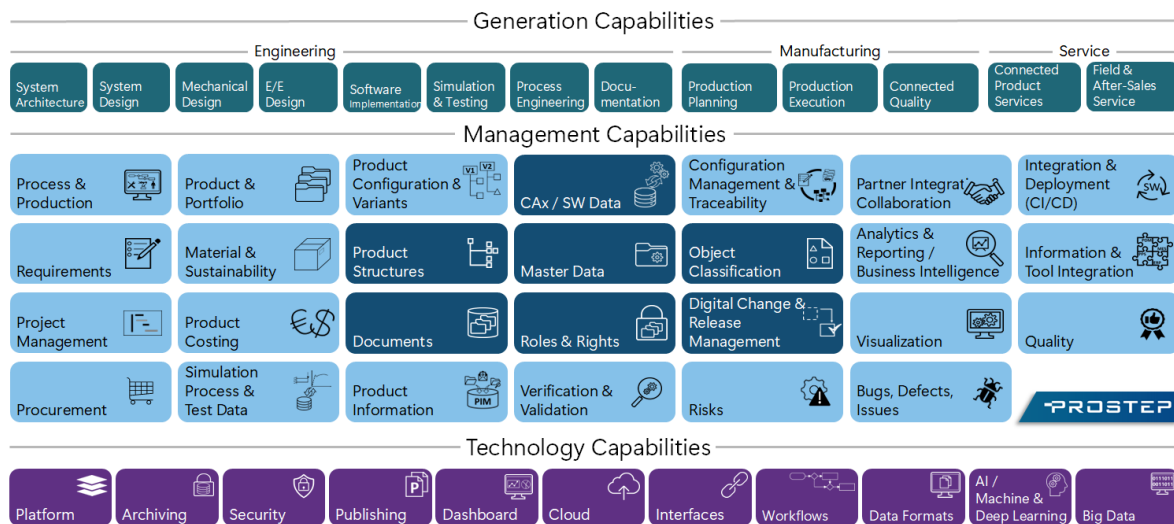


Fig.1: PROSTEP PLM Capability Map

3. Capability-Based PLM Blueprint

To address the challenges of fragmented system landscapes and inconsistent data across the ship lifecycle, this chapter introduces a capability-based PLM blueprint.

3.1. Conceptual Foundation

EAM uses capabilities to structure an organisation's abilities and relate them systematically to the underlying processes, applications, and technologies – an approach that is, for example, well established and widely applied within TOGAF, *The Open Group* (2022). The objective of this approach is to decompose a complex domain [in this case PLM] into manageable, structured elements. A capability is defined as the ability of an organisation to perform a specific, value-adding activity within the product lifecycle. This abstraction creates a clear separation between business processes, functional requirements, and supporting IT systems, and thereby enables a traceable transformation from business needs to IT implementation. By identifying relevant processes and requirements and linking them to capabilities, a structured, vendor-neutral blueprint of the target architecture can be derived – independent of specific applications.

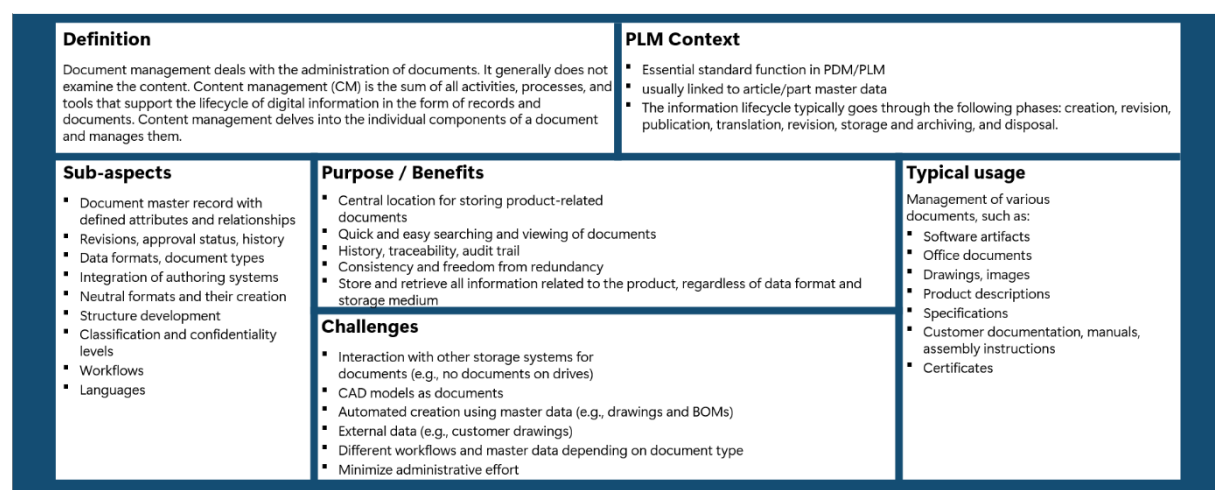


Fig.2: Canvas-based overview slide of the Document Management Capability

This results in a structure that connects business context, processes, data, and IT systems through capabilities, thereby ensuring continuity and transparency across all abstraction levels. This addresses

a key weakness we have observed in current PLM implementations: the lack of continuity between process modelling, capability definition, and system architecture.

3.2. Need for a PLM-Specific Approach

While EAM provides a proven conceptual foundation, generic frameworks such as TOGAF are not tailored to the specific requirements of Product Lifecycle Management. PROSTEP therefore developed an industry-agnostic, PLM-focused method building on the principles outlined in Section 3.1, supporting its established PLM strategy and process consulting practice.

The initial version of the Capability Map, which was developed to support this consulting approach, focused primarily on product development and its associated data management and exchange. In 2018, the Capability Map was substantially extended. This extension was significantly influenced by academic publications from the PLM community, including work from the VDMA (Verband Deutscher Maschinen- und Anlagenbau) as well as renowned universities, *Anderl (2015)*, *Eigner (2013)*, *Stark (2015)*. These sources outlined the vision of an automated, fully networked production environment in the sense of Industrie 4.0. As a result, the method's focus has since expanded to cover the entire scope of PLM – from cradle to grave. As these founding sources originate largely in serial-production industries, applying the method to shipbuilding required additional input to reflect the sector's distinct characteristics set out in Chapter 2. This was drawn from our consulting practice in the marine and offshore sector and from the maritime PLM literature, including *Papanikolaou et al. (2024)*, *Agis et al. (2024)*, *Seppälä (2022)*. The marine literature is explicit that generic PLM has often underdelivered in shipyards when a standard structure was applied without adaptation, *Agis et al. (2024)* – which is precisely why its shipbuilding use rests on tailoring rather than reuse.

The resulting PROSTEP PLM Capability Map is deliberately designed to be industry-agnostic, allowing it to represent cross-cutting topics such as Application Lifecycle Management or Systems Engineering. For use in a specific industry, it must be tailored – for shipbuilding, this tailoring is developed in Section 3.6. This separation is intentional: the structure derives from mature, cross-industry EAM and PLM theory, while its content and level of detail result from a continuous improvement process, informed by practical consulting experience, and close collaboration with the prostep ivip association. This process is reflected in an internal knowledge base, Canvas-based overview slides, and a PLM capability maturity model (see Section 3.5), which together support the application of individual capabilities in practice.

3.3. Structure of the Capability Model

To reflect the diverse business and technological requirements of handling product information throughout the whole lifecycle, PROSTEP's PLM Capability Map is structured into three layers — allowing a continuous perspective from the generation of data, through its management, to the supporting technology involved.

3.3.1. Generation Capabilities

Generation Capabilities describe the creation of product-related data across the ship lifecycle – the engineering, manufacturing, and service activities that produce and refine the digital representation of the vessel, from system and naval architecture through mechanical design and simulation to production planning, execution, and field service.



Fig.3: Generation Capabilities of the PROSTEP Capability Map

These capabilities generate the core data that must remain consistent throughout the lifecycle. Within the Mechanical Design capability, a shipbuilding CAD system is used, for example, for 3D steel structure modelling of the vessel's hull and creates classification drawings as an output.

3.3.2. Management Capabilities

Management Capabilities form the central layer of PLM. They ensure the consistency, governance, and traceability of product related information throughout the lifecycle (including Configuration Management, Variants, Digital Change and Release Management, Product Structures (BOM management), Master Data, and Partner Integration & Collaboration) and lay the groundwork for a company-wide, unified database in line with the PLM guiding principle of a single source of truth. Document Management, for example, is such a capability: it version-controls and releases the drawing created by the shipbuilding CAD system, ensuring the resulting data can be tracked and reused consistently across the shipyard.



Fig.4: Management Capabilities of the PROSTEP Capability Map

3.3.3. Technology Capabilities

Technology Capabilities provide the technical enablers required to implement and scale PLM capabilities, such as integration technologies and interfaces, cloud platforms and data formats, analytics and artificial intelligence, and security mechanisms. Unlike the other two layers, they do not deliver direct business value but provide the foundation for the effective execution and integration of Generation and Management Capabilities.

A workflow engine, for example, is a Technology Capability: it does not generate or govern the classification drawing itself but orchestrates its progression through internal review and release and, beyond the shipyard, the formal approval loop with the classification society, enabling the interaction between the Generation and Management layers, and increasingly across organisational boundaries.



Fig.5: Technology Capabilities of the PROSTEP Capability Map

3.4. Lifecycle Integration and Digital Thread

A key objective of the described concept is the establishment of a “digital thread” that ensures the consistency and traceability of data across all lifecycle phases. In shipbuilding, this requirement is particularly critical due to long project durations, continuous design refinements during construction, and strong interdependencies among the various departments. The “digital thread” connects data and processes throughout a ship’s entire lifecycle, from the early project phases through design and production at the shipyard to sea trials and possibly even operation. It ensures that changes are consistently propagated to all relevant areas.

The capability-based method supports this goal by identifying dependencies between capabilities early on, thereby allowing for an explicit definition of, for example, how traceability between data objects is enabled or what a framework for consistent change management must look like. In this way, the method directly addresses the frequently observed lack of continuity between modelling levels and system implementations. While the digital thread concept itself is industry-agnostic, its practical realisation always depends on how generic capabilities are interpreted within a specific industry context, *PROSTEP (2023)*, a point taken up in Section 3.6 after the assessment methodology has been introduced in the following section.

3.5. Application for Assessment and Roadmap Development

The proposed blueprint is not only descriptive in nature but also prescriptive, as it can be directly applied to support PLM transformation initiatives. Following an established methodological approach, it enables a structured transition from the current state towards a target-oriented PLM architecture. The process begins with the identification of relevant capabilities by systematically mapping business processes to the corresponding PLM capabilities, ensuring that the capability model remains closely aligned with operational requirements and business objectives. The identified capabilities are then prioritised by means of a heatmap that reflects their relevance to the company's business objectives, *PROSTEP (2023)*. Building on this prioritisation, a maturity assessment is conducted to evaluate the current state of each identified capability. This assessment is typically based on predefined maturity levels, allowing for a consistent and comparable evaluation across different domains.

The maturity model associated with each capability provides a structured evaluation framework, typically ranging from initial, ad-hoc processes to managed, defined processes, further to integrated, cross-domain capabilities, and finally to fully PLM-supported cross-company processes that extend collaboration beyond the boundaries of the organisation itself. Subsequently, the results of the maturity assessment form the basis for a detailed gap analysis, in which discrepancies between the current state and the desired target state are identified, thereby highlighting areas that require improvement or transformation. These insights are then used to define a structured implementation roadmap that prioritises capabilities according to their relevance and impact, while outlining the necessary steps for their systematic development and integration. This stepwise approach enables shipyards to systematically evolve towards a fully integrated PLM landscape. The roadmap takes the various aspects from the capabilities and structures them as standalone projects or initiatives at the next level (processes, organisation, data, applications, integrations). Ensuring continuity between the maturity assessment and the resulting architecture and roadmap definition is itself an active area of methodological refinement; academic analysis of this transition has explored model-based tool support to strengthen it, *Bold (2019)*.

3.6. Shipbuilding-Specific Considerations

While the capability-based maturity model is designed to be industry- and vendor-neutral, it must be tailored to the specific characteristics of shipbuilding. These characteristics include engineer-to-order production, complex block structures and assembly sequences, and a strong integration of suppliers throughout the lifecycle. Accordingly, certain capabilities require domain-specific interpretation. For example, System Architecture must be considered in the context of naval architecture and outfitting design. This includes not only specific investigations such as stability or hull-resistance analysis, but also the design of complex onboard systems. Similarly, Production Planning must reflect the specifics of built-to-order processes. Unlike other industries, production planning and work preparation could become a major part of the overall design effort, since it often cannot be amortised across a series of vessels. Furthermore, lifecycle data management must account for ship-specific requirements such as the handling of as-built and as-maintained configurations. As configuration management, product structure management, and change management are closely interdependent, this requirement is best understood as jointly addressed by Configuration Management and Product Structures Management rather than assigned to a single capability alone.

Product Structures (BOM management), already noted above, also raises a distinct and particularly consequential question in shipbuilding: how the product structure itself is organised. Shipyards have a strong document-centric heritage; product information has traditionally been organised around drawings and document registers, frequently indexed by functional classification schemes such as the SFI Group System, the Marine Baugruppenverzeichnis (MBV), or the Enhanced Ship Work Breakdown Structure (ESWBS), each typically maintained in a yard-specific variant. These schemes remain valuable for classification, cost accounting, and maintenance. What they do not provide is an engineering bill of materials (EBOM): they organise the vessel by system or work scope, not as an as-designed product structure. A ship breakdown of this kind is therefore a useful view onto the product, but not the product structure itself. When PLM is introduced, the deeper challenge is less about any single BOM artefact than about challenging the organisation to move beyond a purely document-centric logic towards a managed, coherent product structure. In serial-production industries such as automotive or aerospace, this backbone is provided by the EBOM, to which drawings, configurations, functional classifications, and downstream production structures are linked as views rather than competing masters. Because this shift touches established ways of working and underpins change and configuration management as well as reuse, establishing such a backbone — alongside, not in place of, the classification structures a yard already relies on — is among the most demanding capabilities for a shipyard to mature.

A similar need for domain-specific interpretation applies to the Mechanical Design capability within the Generation layer. At shipyards, this capability is realised through markedly different design disciplines, each shaped by its own tools, working logic, and downstream manufacturing processes. Ship structural design (also referred to as steel or hull structure design) is organised around the shipbuilding-specific logic of ship sections, blocks, and panels, and is typically supported by specialised shipbuilding CAD systems tightly linked to downstream steel production processes such as nesting. Piping and outfitting design, covering piping systems, HVAC, and the placement of inline and smaller devices, relies on dedicated marine outfitting design systems for 3D routing and component placement, and follows a distinct working logic and tool landscape, while remaining closely coordinated with the general arrangement developed under System Design and with the room-level placement coordination carried out by discipline owners. A further sub-discipline, closer to classical mechanical engineering (e.g., the design of hatches, cranes, and other moving mechanical equipment) typically relies on general-purpose mechanical CAD (MCAD) systems originally developed for discrete manufacturing industries, which historically offer strong integration into PLM solutions.

Rather than treating these as separate capabilities, the capability map deliberately groups them under a single Mechanical Design capability, since all of them serve the same purpose within the product lifecycle (generating the mechanical product definition) even though the tools, methods, and underlying design logic differ substantially between disciplines. This reflects a broader principle of the capability-based approach: capabilities are defined by what is achieved for the lifecycle, not by which tool or process a discipline happens to use. Ship structural design, piping and outfitting design, and mechanism design are therefore all equally valid, industry-specific expressions of the same underlying capability and are assessed and matured accordingly, without implying that these disciplines must converge on a shared toolset or adopt a classical mechanical-engineering way of working.

A further, forward-looking consideration concerns data formats and the exchange of engineering information beyond the shipyard's own boundaries. Just as the internal product structure is shifting from a document-centric to a model-centric logic, the external approval process is beginning a parallel shift. From a tool perspective, PROSTEP has developed an internal data format which allows lossless transformation between major shipbuilding CAD tools. External communication, e.g. for ship approval, has traditionally been drawing-based: the shipyard submits classification drawings, which the classification society reviews and approves. A model-based alternative (exchanging the 3D structural model directly between designer/yard and classification society) was set out by *Astrup*

(2017) and has since matured, through a multi-organisation effort, into the open OCX (Open Class 3D Exchange) standard, <https://3docx.org>.

This now enables direct exchange of hull structural information between shipbuilding CAD systems and classification societies, with model-based approval progressively complementing, and in parts replacing, the traditional drawing-based route. For shipyards, this elevates a capability that has historically received little strategic attention: Data Formats. Notably, this single shift does not sit within one capability alone; it spans Data Formats and Interfaces in the Technology layer and Partner Integration & Collaboration in the Management layer, again illustrating how the capability map exposes the full organisational footprint of an apparently technical change. Shipyards would therefore be well advised to treat data-format strategy, and OCX support in particular, as an explicit part of their PLM roadmap rather than a downstream integration detail.

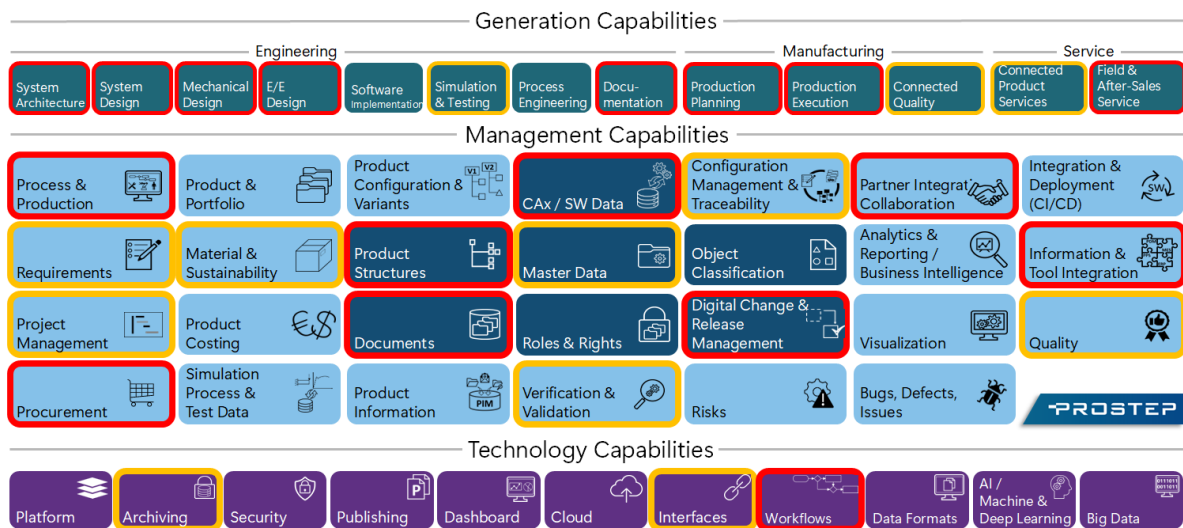


Fig.6: Aggregated heatmap reflecting common capability priorities across shipbuilding customers

By incorporating these considerations, the proposed blueprint ensures that it is not only conceptually sound but also directly applicable within shipbuilding environments. In shipbuilding projects, capabilities such as Configuration Management and block-based Production Planning typically score high on the heatmap, reflecting their direct link to regulatory compliance and engineer-to-order project structures. Given the diversity of design disciplines described above, System Design and Mechanical Design likewise emerge as critical capabilities in most shipyards, warranting particular attention during capability prioritisation and maturity assessment, Fig.6. Product Structures Management is similarly prominent, and often the hardest shift to achieve (the obstacle being organisational rather than technical) which makes it arguably the highest-value opportunity a shipyard can pursue towards a resilient, future-ready PLM foundation.

4. Application and Validation

The capability map was primarily developed as a tool to help customers to decide on the focus area for a PLM introduction or as a benchmark for the maturity of existing implementations. The latter is of great importance when one is forced to replace obsolete technologies. The capability map itself is not a purely theoretical construct but a product of this practical experience. Its current structure emerged from earlier information flow analyses conducted with several shipyards, including two shipyards in northern Germany and one in Australia, in which industry-specific capabilities served as the starting point for identifying how information actually flows between disciplines, e.g. establishing what information is generated, consumed, and exchanged across shipyards' domains. The industry-specific capabilities identified during these engagements directly shaped the structure of the generic capability map presented here, refined and generalised for broader application. Over the past three years, we have used this capability map as a basis for our consulting work in shipyards.

Most shipyards have no recent experience with introducing a PLM solution or replacing an outdated system with a new one. Starting with a blank sheet often leads to the case where the business places all their wishes as must-have requirements for the introduction of PLM. The capability map could be used as Occam's razor to filter out free riders who want to use PLM initiatives for other purposes such as introducing another CAD tool. Another reason is to check whether all relevant information and business processes are investigated when talking about product lifecycle management. Quite often, shipyards are steered by a PLM vendor towards what their own product supports best. This is rarely for the benefit of the shipyard, especially if this is an aspect where the cost-benefit ratio is not the best. The huge landscape offered by the capability map provides a good starting point to get an overview of what topics could become part of PLM processes and check on potential benefits.

From then on, one could derive a customer-specific heatmap from the shipbuilding-representative baseline shown in Fig.6, following the same prioritisation logic introduced in Section 3.5. This heatmap acts as a guideline for the focus topics in a PLM introduction or upgrade. If certain aspects are already successfully covered or little benefit is potentially gained from adding more management capabilities, they are deprioritised, allowing consulting effort and roadmap steps to concentrate on the capabilities with the greatest transformation potential for the specific shipyard. Beyond capability identification and prioritisation, the approach has proven applicable across several structurally different types of engagements, illustrating that its value is not limited to a single use case. In an engagement with one of the northern German shipyards mentioned above, the capability model was applied across the full course of a PLM introduction. It first served as the basis for an information flow analysis used to derive precise requirements for selecting a PLM solution, comparable to a formal requirements specification. This phase had to explicitly account for the shipyard's use of both shipbuilding-specific CAD systems for structural design and piping, and MCAD for discrete mechanical equipment (see Section 3.6), each requiring a different integration approach. Building on these requirements, a benchmark of candidate PLM solutions was carried out, in which capabilities were used to obtain a measurable indication of how well each solution matched the shipyard's actual needs, informing the final system selection. Once the implementation project began, the capability model was used again — this time to help shape the solution design and implementation process. The PLM implementation itself was performed by a separate system integration partner, illustrating that the concept is not restricted to in-house usage. This continuity across requirements definition, system selection, and implementation illustrates, in practice, the same principle of cross-level traceability described conceptually in Section 3.1.

For a new shipyard under development in the United States, the capability map was applied in a greenfield context, where no legacy PLM landscape yet existed. Here, capabilities provided a vendor-neutral way to describe, at a high level, what a digital PLM landscape for the shipyard would actually need to support, before any specific software decision had been made. The three-layer structure of Generation, Management, and Technology Capabilities introduced in Section 3.3 proved particularly useful here, as it allowed the future landscape to be organised independently of any specific tool or vendor from the outset. A Finnish shipyard was independently pursuing a similar capability-based approach to evaluate PDM software against its own requirements. PROSTEP supported this effort by outlining its methodology and sharing illustrative examples of how capabilities can be used to describe and compare vendor solutions, which the shipyard then applied on its own to support its tool selection. This gave the shipyard a consistent, tool-independent basis for comparing candidate solutions, turning the selection into a measurable comparison rather than a subjective preference for a specific vendor. This illustrates that the underlying capability-based principle not only supports PROSTEP-led consulting engagements but can also be adopted independently by shipyards as an internal decision-making tool — even without transferring the specific capability map introduced in this paper.

The capability model does not replace the underlying technical evaluation itself. Compiling an industry-relevant shortlist of candidate PLM solutions remains a core part of PROSTEP's consulting expertise, since not every system is a realistic contender for shipbuilding-specific requirements. Once a shortlist has been established, the capability model's contribution is to make the subsequent

evaluation comparable: without a shared, capability-driven basis, different vendors would present their solutions according to their own strengths, making an objective, like-for-like comparison difficult. The capability model therefore serves as a qualification framework that structures and grounds the technical evaluation on comparable terms, rather than substituting for it. Taken together, these engagements (spanning full-lifecycle support from requirements definition through vendor selection to implementation shaping, greenfield strategy definition, vendor-neutral requirement specification, and the map's own empirical origin in industry-specific information flow analysis) demonstrate that the capability-based blueprint provides a consistent, reusable foundation across the different situations in which shipyards approach PLM transformation, rather than being tailored to a single project type or shipyard profile.

5. Conclusions

PLM is no longer optional for shipbuilding organisations, but a prerequisite for managing product information across complex and distributed environments. Regardless of whether a formal PLM strategy exists, shipyards must manage requirements, product structures, engineering changes, configurations, and lifecycle information throughout the delivery process. The challenges identified – fragmented system landscapes, data silos, product complexity, and demands for traceability and lifecycle continuity – cannot be addressed through isolated tools or departmental initiatives alone. Without a structured approach, lifecycle information evolves in disconnected processes, limiting transparency and the ability to manage change.

The capability-based PLM blueprint proposed in this paper provides a vendor-neutral framework for systematically linking business processes, organisational requirements, and supporting systems. By establishing continuity between these layers, it offers a foundation for assessing existing PLM landscapes and guiding future transformation initiatives within shipbuilding organisations.

As programmes grow in complexity and duration, the industry faces what automotive and aerospace passed decades ago: the need for a common, vendor-neutral way to describe what PLM means. The blueprint presented here, validated through the engagements in Chapter 4, is a first step towards such a shared reference model. Ultimately, shipbuilding PLM will be defined not by the right software platform, but by a shared understanding of the capabilities needed to design, build, and sustain complex vessels.

References

- AGIS, J.J.; BRETT, P.O. (2024), *Digital Shipbuilding – Needs, Challenges, and Opportunities*, 15th Int. Marine Design Conf.
- ANDERL, R.; FLEISCHER, J. (2015), *Leitfaden Industrie 4.0 – Orientierungshilfe zur Einführung in den Mittelstand*, VDMA, 2015.
- ASTRUP, O.C.; CABOS, C. (2017), *A Model Based Approval Process for Basic Hull Design*, ICCAS Conf., Singapore
- BOLD, L. (2019), *PLM-Fähigkeitsbasiertes Architekturkonzept zur Entwicklung cyber-physischer Systeme*, Karlsruher Institut für Technologie (KIT)
- CALVACHE, M.; PRUYN, J.; NAPOLEONE, A. (2025), *Navigating shipbuilding 4.0: analysis and classification of technologies for the digital transformation of the sector*, Ship Technology Research
- EIGNER, M. (2013), *Modellbasierte Virtuelle Produktentwicklung auf einer Plattform für System Lifecycle Management*, in *Industrie 4.0 – Beherrschung der industriellen Komplexität mit SysLM*, Springer Vieweg

OECD (2025), *Industrial policy for the future: Insights from the shipbuilding industry*, OECD

THE OPEN GROUP (2022), *The TOGAF® Standard, 10th Edition – Business Architecture*, The Open Group

PAPANIKOLAOU, A.; BOULOUGARIS, E.; ERIKSTAD, S.O.; HARRIES, S. (2024), *Ship Design in the Era of Digital Transition – A State-of-the-Art Report*, 15th Int. Marine Design Conf.

PENCIUC, D.; LE DUIGOU, J.; DAABOUL, J.; VALLET, F. (2016), *Product Lifecycle Management Approach for Integration of Engineering Design and Lifecycle Engineering*, AI EDAM

PROSTEP (2023), Strategic PLM Consulting, Whitepaper, www.prostep.com/whitepapers

SEPPÄLÄ, L. (2022), Information Management in Shipbuilding Projects – Information Flow from 3D Design to Production Data, ICCAS Conf.

STARK, J. (2015), *Product Lifecycle Management (Volume 1): 21st Century Paradigm for Product Realisation*, Springer

Generative AI Implementations in Ship Design at OCEANS: From Task Automation to Process Transformation

Yasuo Ichinose, University of Osaka, Osaka/Japan, ichinose.yasuo.eng@osaka-u.ac.jp

Kouki Wakita, University of Osaka, Osaka/Japan, wakita.kouki.eng@osaka-u.ac.jp

Abstract

One task made faster and a change in how design is done are both called the effect of generative AI, and no measure tells them apart. This paper offers one: read the impact in three layers — task automation, process automation, business transformation — each judged by its own measure. Drawing on OCEANS implementations, we fill the layers from the bottom by kaizen. At the task layer every case rested on written knowledge. At the process layer design intent crossed tool boundaries; agents carrying the exchange between organizations come next. The largest impact, we expect, lies in the business layer.

1. Introduction

The Japanese shipbuilding industry faces growing demand for next-generation vessels while it struggles with labour shortages and a long delay in digital transformation (DX), *Ichinose and Wakita (2026)*. Generative AI is expected to change that situation. What is hard, on the design floor and in management alike, is knowing where to place that expectation. A report that one task became faster and a prospect that the whole way of designing will change are both spoken of as the effect of generative AI. To size an investment, or to decide what to automate towards, one needs a measure that tells these apart.

Reports on generative AI at COMPIT trace a short but steep history. *Gaspar and Bertram (2023)* tested ChatGPT on marine-engineering tasks and found more smoke than fire in specialised domains, yet suggested that bots trained on a company's intellectual property might become useful. *Bronson et al. (2025)* showed that a large language model (LLM) with retrieval-augmented generation can validate design parameters across fragmented documents. *Ando and Seppälä (2025)* identified data fragmentation as a structural reason why lifecycle information in shipbuilding and shipping is not integrated, and pointed to the Model Context Protocol (MCP), an open standard for connecting AI agents to tools and data, *Anthropic (2024a)*, as one way out. Each is a solid step as an individual application. What none of them offers is the measure itself: a way of saying how far such results reach, and in what order.

Why we think such a measure is now within reach has to do with a second history, one that runs alongside the first. The object of engineering effort on the user's side of AI has itself been moving, and moving outward. It began with the prompt, the wording of a single instruction, which grew into a field of its own, *Sahoo et al. (2024)*. It moved to context: not one instruction but the whole set of information an agent is given and keeps while it works, which *Anthropic (2025a)* distinguishes from prompt engineering as a continuing process rather than a discrete task. It moved again to the harness, the tools and the environment an agent acts in, where progress is judged by the ground truth the environment returns at each step, *Anthropic (2024b)*. What is forming next is the loop: the agent generates, an evaluator scores, and the score feeds the next attempt.

This progression is not incidental to the argument; it is why the impact spreads outward instead of deepening in place. A prompt reaches one task. A harness reaches as far as the tools reach, which is a process. A loop reaches as far as its evaluation function encodes, which is whatever a company holds to be valuable. The three layers we propose are, read from the other side, the places where these four stages come to ground.

This paper proposes such a measure: reading the impact of generative AI in three layers: task automation, process automation and business transformation. How far the impact reaches in each layer cannot, we think, be deduced from above. So we take the approach that Japanese manufacturing has built up, kaizen: turn one complaint at a time (this work is heavy) into a requirement for automation, and from what follows, look at what happens in the layer above. What this paper shows is how far that accumulation has come at OCEANS (Open Collaboration Laboratory for Enabling Advanced Marine Systems), the industry–academia laboratory at The University of Osaka, *Ichinose et al. (2025)*.

2. The Roles of AI in Ship Design

Before asking where generative AI changes the design process, it helps to be clear about what AI actually does inside design activity. Fig.1 gives the whole picture. Design activity is drawn as designers operating design resources under the requirements, policies and rules set by the design board. Those resources are structured databases, unstructured data such as knowledge bases, CAD models, simulators, and the repository of PLM and BOM. What generative AI newly supplies is the four components shown in black: the AI interpreter, AI-based automatic model generation, the AI-based surrogate model, and the AI agent that orchestrates them. Of the four, only the surrogate model continues from before; the other three appeared after the LLM.

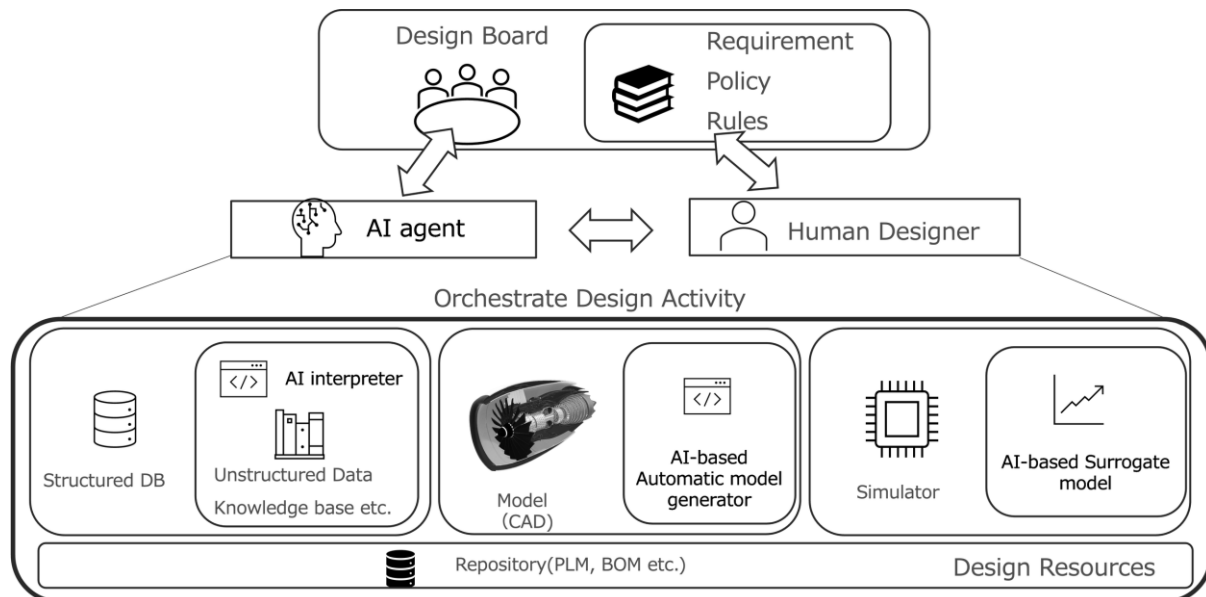


Fig.1: Roles of AI in ship design. The four components in black are supplied by AI; those in grey are the design resources that already existed

2.1. Surrogate Models: What Continues

Of the four, the surrogate model is the one generative AI did not create. Machine learning has been substituted for simulation in naval architecture for two decades, from early neural networks to deep learning, with hydrodynamic performance as the recurring target. *Ichinose and Taniguchi (2022)*, for one, predicted the wake field with a convolutional neural network built on a representation that treats grids on the hull surface as an image, reaching a coefficient of determination of 0.986 over 2,730 hull forms, with inference under 0.01 s, about 100,000 times faster than the physics-based computation. More complex targets, structural analysis among them, are now within reach; at OCEANS the target is CFD in waves, the most expensive region to compute (Section 4.2). What generative AI changed is not the surrogate itself, but where it can sit. It can now be placed inside an automation loop, as the evaluator an agent calls.

2.2. Automating Model Generation

Model generation is where the discontinuity in kind is clearest. Parametric and rule-based design in CAD have been practised for a long time, but the automation CAD brought after the 1980s was confined to simple calculations and work executable with basic command sets, *Ichinose et al. (2025)*. Generative AI targets multi-step processes that follow the designer's intent, which shifts the designer's role to conveying that intent to the AI, *Ichinose et al. (2025)*. Two things make that possible. Instructions are given in natural language, and the AI reads errors and non-converged results and corrects itself. The technical key is MCP, through which an LLM operates external applications directly, *Anthropic (2024a)*. Put another way: a domain in which only what could be written deterministically in a programming language could be automated is now open to design intent, which carries ambiguity. To handle that intent in stages, *Hatayama and Ichinose et al. (2026a)* divided the knowledge given to the LLM into three tiers of context — design intent, CAD environment and geometric specification (Section 4.1).

Fig.2 shows the structure this produces. The human supplies design intent and objectives. Generative AI decomposes them into design steps and design operations, and issues them through MCP as command sets. Below that, design tools such as CAD, CFD and FEM, together with in-house calculation tools, do the machine work. The context the agent needs comes from two stores: the yard's knowledge, meaning procedures and past-ship data, and knowledge of the tools themselves.

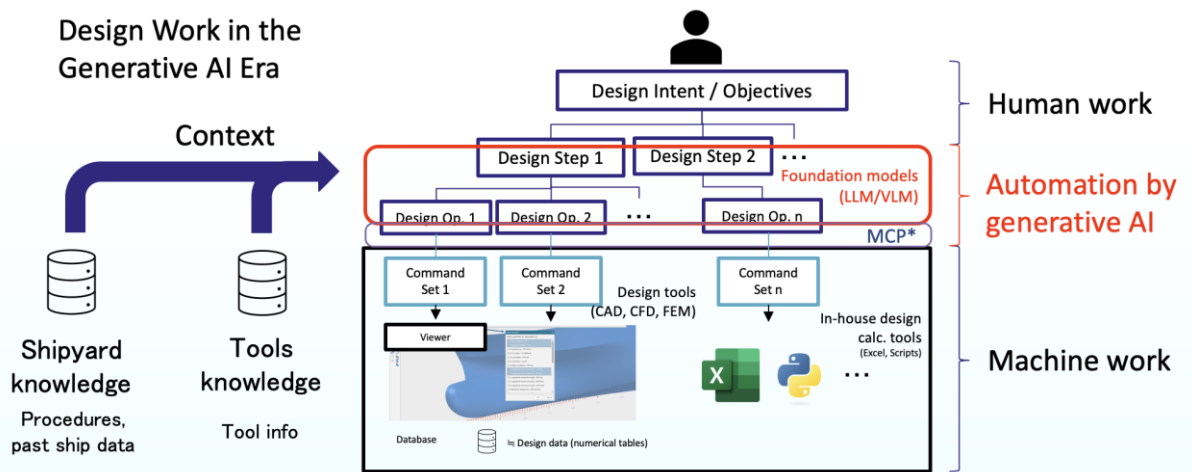


Fig.2: Design work in the generative-AI era: what the human supplies, what generative AI automates, and what the tools execute. Redrawn from *Ichinose et al. (2025)*

2.3. The AI Interpreter

The AI interpreter removes a step that used to stand in front of every AI project: converting documents into a structured form. Data that AI could handle had to be structured, as in a relational database, and most of the knowledge a shipyard holds has no such form. Textbooks, technical reports, in-house design standards, rules, past drawings, specifications which all are unstructured documents. An LLM reads them as they are. Being able to hand a shipyard's accumulated documents to AI without that conversion is, we think, the largest single gain among the four functions. At OCEANS this function is being tried on the drafting of specifications (Section 5.2) and on 3D models.

2.4. AI Agents and Tools Made for One Person

The AI agent combines the three above and carries them out as one piece of work. What is new is that a workflow can be assembled in natural language rather than programmed. The same change is reaching the tools a designer holds. A 3D viewer or a design interface used to demand substantial effort from a programmer, so no individual could have one for a single job. That such an interface can live

in a browser is not a projection: *Ichinose and Gaspar (2023)* ran an interactive simulation of the flow around a hull, with a neural network as its evaluator, in a web environment. With a library such as three.js, AI now takes on the data conversion as well, and a designer can have a viewer made for one person. A carpenter who shapes a plane to their own hand ends up with a tool no one else needs; design tools are becoming the same kind of thing.

2.5. The More Personal the Tool, the More the Data Matters

Once interfaces become personal and AI absorbs format conversion, value moves from the tool to how the data behind it is held. This does not contradict Section 2.3. An LLM can read a document as it stands, but reading a document is not the same as knowing how the objects in it relate to one another across stages and across systems. That relation is what has to be held, and holding it is what industry has been investing in: semantic layers, knowledge graphs and ontologies, with some vendors positioning the ontology itself as their competitive advantage.

The problem itself is not new. In Japanese shipbuilding, how to hold production-design information was already being worked out around 1990 under shipbuilding CIMS, *Amemiya (1991)*, *Funaoka and Kakuno (1992)*, and in concrete terms. Fig.3 places the systems of a shipbuilding CIMS on two axes, work stage and discipline, from basic planning through detailed design to fabrication and assembly, and shows which data each system hands to the next. Fig.4 gives the structure of the information those systems handle, relating product design information, intermediate product information, resource information, process information and work information. An agent that is to act across these stages needs exactly this: what each object is, and how it relates to the others. The vocabulary has changed, but the problem set out under CIMS is the one an ontology addresses today.

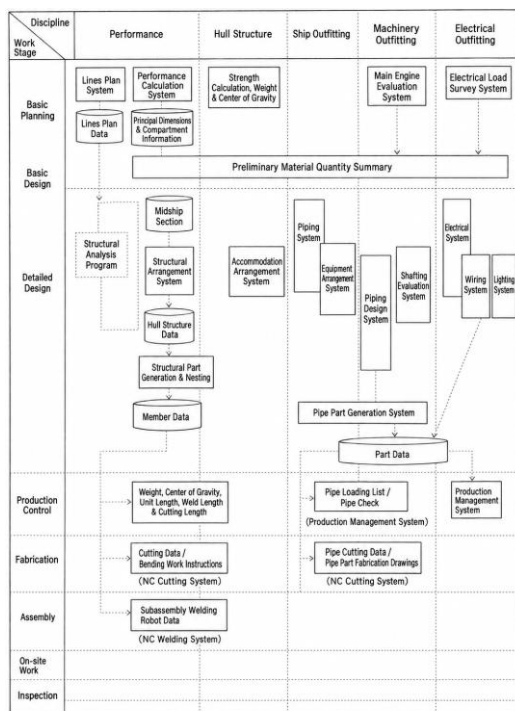


Fig.3: Software configuration of a shipbuilding CIMS, arranged by work stage and discipline. Redrawn after *Funaoka and Kakuno (1992)*

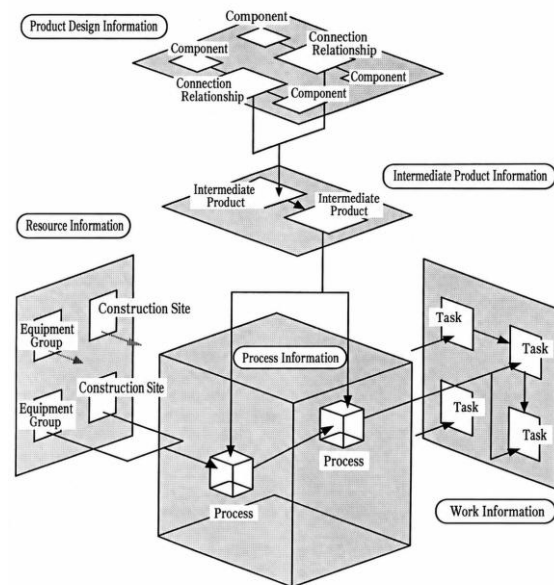


Fig.4: Data structure for process planning information under shipbuilding CIMS. Redrawn after *Amemiya (1991)*

So the work of organising databases and introducing PLM does not go away because generative AI has arrived. As long as the objectives and processes of design and construction stay the same, the architecture of the systems that support them is universal, *Ichinose et al. (2025)*, and the lifecycle

information architecture discussed by *Ando and Seppälä (2025)* is its present-day form. What changed is the party that uses the organised information: an agent rather than a person.

2.6. What These Four Functions Do Not Tell Us

The four functions are an inventory of what AI can do. They do not say what any of it is worth to a shipyard. The same function means one thing when it makes one person's work faster, another when it rearranges the flow of design, and another again when it changes the decision of which ship to build and how — and the preparation each requires, and the size of the investment, differ as much. The next section sets out the three layers we use to tell these apart, and Sections 4 to 6 report what each layer looks like once the four functions are put to work in it.

3. A Three-Layer Model of AI Impact

We read the impact of generative AI in three layers, as Fig.5 shows, *Ichinose and Wakita (2026)*. Each layer is defined by what AI is being asked to do there, and each is judged by a different measure. Read against Section 1, the task layer is where prompt and context engineering land, the process layer is where the harness lands, and the business layer is where a loop would.

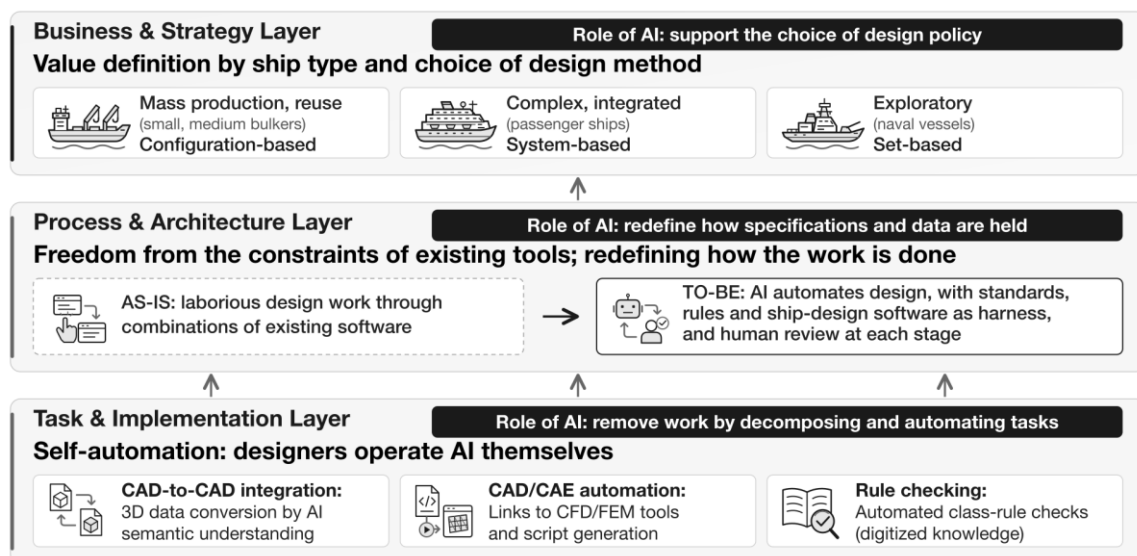


Fig.5: Three-layer structure of generative-AI development grounded in naval architecture

3.1. The Task Layer

The task layer is the individual work a designer does every day. The role of AI here is to decompose manual work, automate it, and so remove the work itself. This is the unit of kaizen: the difficulty is felt by one person, the effect is measurable on the spot as working time, and a failure stays inside that task.

3.2. The Process Layer

The process layer is the flow of work across design stages, across tools and across organizations. In Japanese yards several 2D and 3D computer-aided design (CAD) systems coexist across stages, and design data are not integrated in a form that AI can use (*Ichinose and Wakita 2026*). The role of AI here is to redefine how specifications and data are held, and to build a flow without rework. Where the AS-IS is laborious work connecting combinations of existing software by hand, the TO-BE is automated design by AI with design standards, rules and ship-design software as its harness, and human review at each stage. The measure here is rework avoided.

3.3. The Business Layer

The business layer is the definition of value by ship type, and the choice of design method that follows from it. For series ships with an abundant construction record and a known design space (configuration-based design), the role of AI is to learn from past ships and streamline design within that space. For custom ships whose design space is unknown, such as liquefied-hydrogen or ammonia-fuelled vessels (system-based design), it is to explore the space with many physics-based simulations and to reduce rework caused by information gaps. For exploratory types such as naval vessels, set-based design, which keeps several candidates open, is called for. Which to take is a management decision, and the role of AI is to support it. The measure here is not time but the quality of that decision, which is why this layer is the hardest of the three to verify.

3.4. Why We Build from the Bottom

Our working hypothesis is that the layers are ordered: transformation at the process layer needs task-layer automation to be in place first, and decision support at the business layer needs the data and the speed of evaluation that the process layer produces. We have not tested this order, and a yard might well change its business layer by a management decision alone. But the upper layers take longer to verify and demand decisions we are not in a position to make, whereas the task layer can be worked on today. So we build from the bottom, and the three sections that follow report how far that has taken us.

4. The Task Layer: Automation by Kaizen

The task layer is where automation has gone furthest, and this section presents the four implementations we have there. Each began the way kaizen begins, from someone saying that a particular job was heavy: mastering a CAD macro language, waiting a week for one CFD condition, redrawing the same diagram. Table I lists them together with the two process-layer cases of Section 5, so the whole set can be compared by the boundary each had to cross.

Table I: Implementations at OCEANS, ordered by the boundary crossed

Case (tool)	Knowledge the AI relied on	Boundary crossed	Outcome
Self-automation of a ship-design CAD (NAPA Designer)	Manual and example scripts; task context in three tiers	None	24 of 28 runs built as specified (85.8%)
Surrogate for hull pressure in waves (U-Net)	Unsteady CFD results	None; evaluator for the loop	$r = 0.972$; 1.5 ms per sample
Pre-trained surrogate transferred (GeoPT)	Calm-water CFD results	None; evaluator for the loop	33.5% fewer training hulls
Process flow diagram generation (AutoCAD)	Drawing rules of ship systems, written as a skill	None; same pattern on a second tool	Ballast-line PFD from a short instruction
Local vibration check (NAPA Designer, Excel)	Procedure, written as a skill by one person	Across tools	4 of 4 runs; designer's time 24%
Series-ship specifications	Past specifications; reasons behind decisions, settled in exchange between parties	Across organizations	Repetition reduced; agents to carry exchange and record reasons

4.1. Self-Automation of a Ship-Design Platform

NAPA, the standard platform for basic design, stability and hydrodynamics, has a macro language that takes long to master. At OCEANS an LLM agent instructed in natural language did the work

instead. Through an MCP server the laboratory built once, the agent retrieves the relevant parts of the tool’s manual and example scripts, generates code, executes it through the NAPA API, and reads compile errors and non-converged results to correct itself, *Ichinose and Wakita (2026)*. We call this form, in which designers operate AI themselves, self-automation.

Fig.6 places this form in three stages: create, store and use, *Ichinose and Wakita (2026)*. A veteran designer writing a procedure manual is the creation of a skill, which is to say the formalization of knowledge. The skills so written are collected in a skill hub, and an execution AI reads them and operates the CAD and its database. What the hub holds is then available not only to the person who wrote it but to every designer. The knowledge moves from an individual to the organization at the moment it is written down, not at the moment it is used.

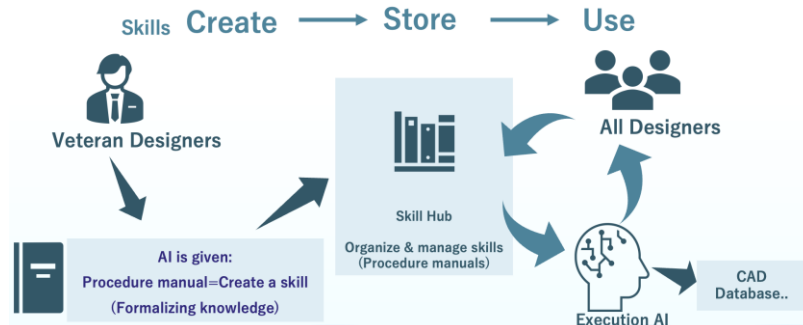


Fig.6: Skills as knowledge that is created, stored and used. Based on *Ichinose and Wakita (2026)*

Fig.7 shows the system in operation. On the left are the NAPA Designer window and the script execution window; in the centre, the agent that takes the instruction and generates the code; on the right, the knowledge base of the tool and that of the shipyard, both reached through the MCP server, *Hatayama and Ichinose et al. (2026a)*.

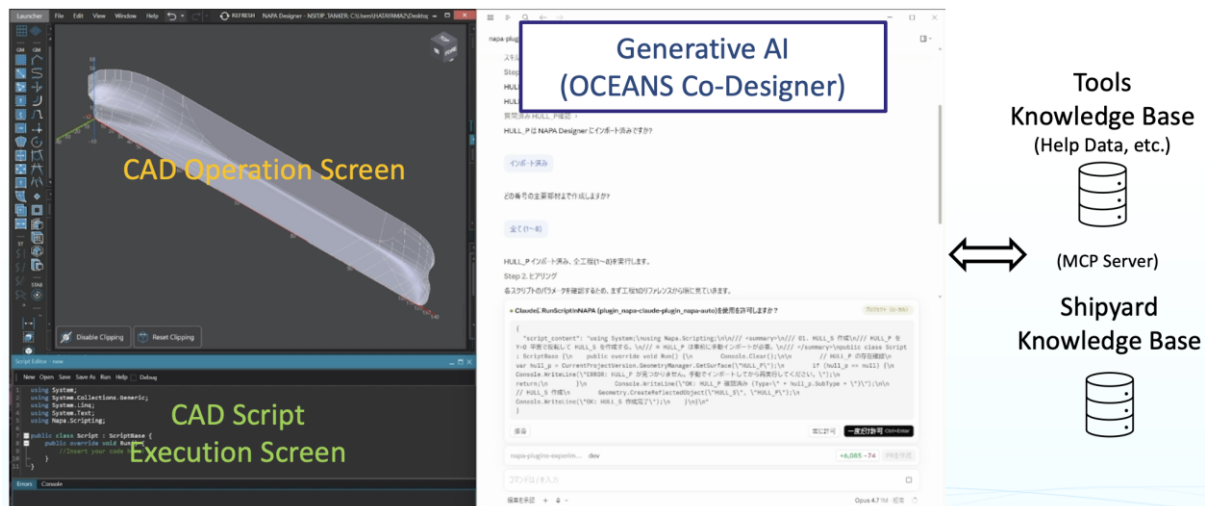


Fig.7: The agent operating NAPA Designer, with the tool and shipyard knowledge bases reached through an MCP server. From *Hatayama and Ichinose et al. (2026a)*

Hatayama and Ichinose et al. (2026a) verified the same problem in a separate setting, using GPT-5 and retrieving NAPA documentation through function calling rather than MCP. That experiment made the knowledge given to the LLM explicit as three tiers of context: design intent, CAD environment (the identifiers of existing objects) and geometric specification. Fourteen structural members of a tanker were run twice each, 28 runs in all, and in 24 of them (85.8%) the member was built in the CAD exactly as specified. The ablation is what matters here. Without design intent, the remaining two tiers acted as noise: the semantic score, an LLM-judged match to the intended operations, fell to 41.6% against 53.1% for intent alone. Without the CAD identifiers, geometric data never reached the

CAD. The first knowledge to hand an AI is therefore design intent, and unless it has been written down, adding other information does not help.

4.2. CFD Surrogates as Evaluators

An automation loop, in which generated designs are evaluated and the result fed back, needs an evaluator cheap enough to sit inside it. Two surrogate studies supply one. *Nakaue et al. (2026)* trained a U-Net on unsteady Reynolds-averaged Navier–Stokes results for the KRISO container ship in regular head waves (25 draft and trim conditions, 50 phase snapshots each) to predict the hull-surface pressure distribution. On unseen conditions it reached a weighted correlation of 0.972 ± 0.013 . Inference takes 1.5 ms per sample, about seven orders of magnitude faster than CFD, which needs about one week per condition. *Miyata et al. (2026)* showed, over a set of 2,846 hull forms, that GeoPT, *Wu et al. (2026)*, a general-purpose surrogate pre-trained without solver data, reaches the error of a model trained from scratch with 33.5% fewer training hulls, about 647 CFD runs saved. What pre-training changes is not the level reached but the speed of reaching it against the number of hulls.

Fig.8 shows the hull-surface pressure for one condition: (a) the CFD ground truth, (b) the U-Net prediction and (c) the difference between them. The prediction reproduces the pressure peak at the bow and the negative-pressure region along the parallel body, and the error concentrates near the waterline. Physics-based simulation, the most firmly established body of explicit knowledge a yard holds, compresses into a millisecond evaluator. That is the premise of the loop discussed in Section 6.

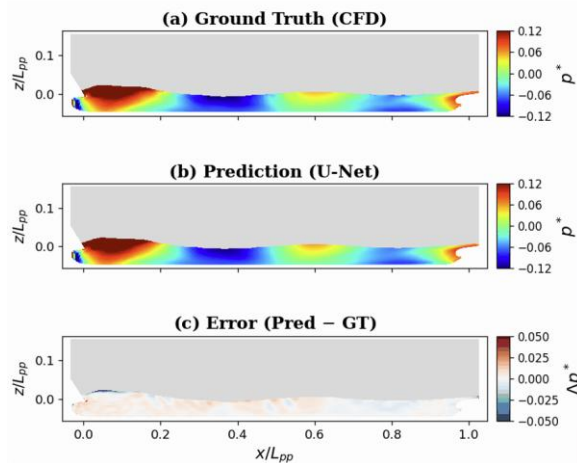


Fig.8: Hull-surface pressure in waves: (a) CFD, (b) U-Net prediction, (c) difference. From *Nakaue et al. (2026)*

4.3. Portability to a Second Tool

The pattern of Section 4.1 is not specific to one tool, as an application to a drafting tool showed. Claude Code, an LLM agent that runs in the terminal, was connected to AutoCAD through an MCP server and given a skill, a procedural document an agent loads when a job calls for it (Anthropic 2025b), written from the drawing rules of ship systems. In the first phase of the project, it drew a ballast-line process flow diagram (PFD) from a short instruction in natural language. Once the tool connection is built, the work of automation is to write down the procedure.

4.4. What the Task Layer Showed

Across the four implementations, the knowledge the AI relied on always had a written source. In two of them it was written already: the tool’s manual and example scripts, and the output of a physics solver. In the other two, it had to be written out, but from something written: the drawing rules of ship systems became a skill, and the intent of a modelling task became a tier of context. In none of them

did anyone have to excavate knowledge that existed only in a person's head. The work of automation moved from writing code to writing down what the work requires.

A word on how this work is situated. OCEANS is a collaborative research course run with industry partners, and these implementations advance step by step, with practising designers from those partners acting as advisors who check each result as it comes. What we work on, and how we judge whether it is any good, both draw on real design data and on advice from the design floor. What we publish is then built on open data, so that others can reproduce it. Within that frame the figures shown cover a limited range: one tanker and fourteen members, one hull form in waves, and part of the scoring judged by an LLM rather than by a person.

5. The Process Layer: Connecting Tasks

Task-layer automations working one by one do not change the flow of design. The process layer changes when they connect across tool boundaries, and when the information they need flows across organizational boundaries. OCEANS is in the middle of implementing this layer, so this section reports two different states: the tool boundary, which we have crossed, and the organizational one, which we are building towards.

5.1. Across Tools

Hatayama et al. (2026b) automated a job that spans tools, the local vibration check of hull panels. The procedure was written as a skill by one member of the laboratory. With it, Claude Code retrieved panel dimensions from NAPA Designer through the laboratory's own MCP connection, computed natural frequencies in Excel, proposed three countermeasures with their costs, and after the designer's approval wrote the chosen change back to the model. All four runs on two members completed. A run took 4 min 27 s on average, of which the designer's own time was 24%, the rest being the AI's reasoning and the execution of tools; the cost was 0.93 to 1.35 US dollars per run at list prices. What crossed the boundary between NAPA Designer and Excel was the intent of the check, not a file format. The scale is small (two members, four runs) but it is the first instance of the mechanism we expect at this layer.

Fig.9 shows a run. The flow on the right goes from reading the skill through retrieving data from NAPA, calculating in Excel and creating the report, to applying the change back to the model. The panels the check concerns are highlighted in the CAD window, and the agent reports each step as it completes.

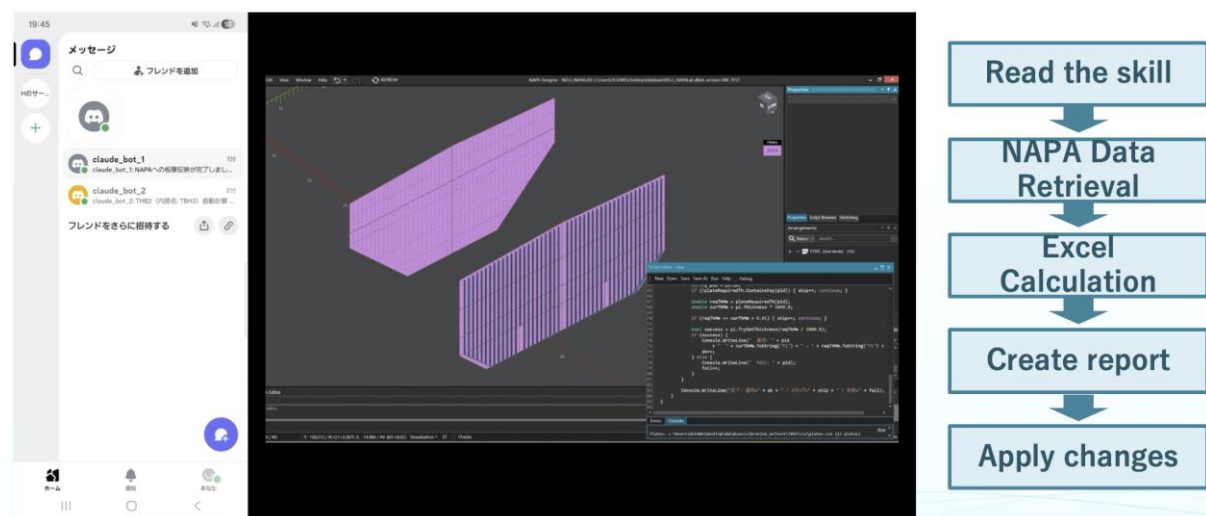


Fig.9: A run of the local vibration check across NAPA Designer and Excel. From *Hatayama et al. (2026b)*

5.2. Across Organizations

The next boundary at the process layer lies not between tools but between organizations. In drafting the specifications of a series ship, an LLM agent working from the documents of past ships already removes much of the repetitive work. What remains is the exchange among the yard, the owner and the other parties, in which the shape of a specification is settled. A specification records what was decided; the reasons are settled in that exchange.

That exchange is now coming within reach, because an agent acting on a person's behalf can carry it: drafting the e-mail, preparing the document, tracking the points still open. When the agent of one party works with the agents of the others, the points needing confirmation can be organized between them, and what is decided — together with the reason it was decided — can be recorded at the moment it is settled. A network of agents of this kind would do two things at once. It would make the work that spans organizations faster, and it would accumulate, as data, the course of a negotiation that has until now lived only in the exchange itself. This is the part we are building next, and it is the point at which automation at the process layer begins to supply what the business layer needs.

5.3. What the Process Layer Showed

What the two sections have in common is that the reach of automation follows the knowledge that has been written down. In the vibration check the procedure could be written down by one person, and the execution across tools moved to the AI as it stood. Across organizations what has to be written down is not a procedure but the reason behind a decision.

What has changed here is larger than it first appears. Until now, making knowledge usable meant that the person holding it had to put it into words and then convey those words to a programmer. Now the negotiation itself can be recorded, and AI can analyse it: what the open points were, what was decided, and on what grounds. The labour of putting knowledge into words no longer rests on the holder alone.

This meets a bottleneck that dates from the era of expert systems. Knowledge acquisition was the hard part then, *Schreiber et al. (2000)*, because everything had to be written out as programming code, and the verbalization that demanded was heavy labour. What generative AI breaks through is that limit of labour, not the need for the knowledge itself. The bottleneck has moved from who can write the code, to whose knowledge has been written down, and now to a point where AI assists in the writing down. Placed against the four stages of Section 1, this is where we stand. The three-tier context of Section 4.1 is context engineering, and the vibration check above is harness engineering: it is putting the harness in order that we currently find most effective. The loop is the stage we have not reached, and Section 6 is about what it would take.

6. The Business Layer: Writing Down What a Yard Values

The fourth stage of Section 1, the loop, is where the business layer is decided. A loop requires an evaluation function, and what that function encodes is what counts as a good design. For hydrodynamic performance the evaluation function is the solver, and the surrogates of Section 4.2 compressed it enough to sit inside the loop. But the quality of a design as a whole is not settled by hydrodynamic performance.

The criteria a yard weighs are of several kinds, and the following list is not exhaustive. Compliance with rules is an absolute condition, not open to negotiation. Construction constraints are set by the facilities: block size is bound by crane capacity. On top of that, cost, safety, delivery date, and how much disturbance to the production schedule can be tolerated are weighed against one another. How much risk to take against which uncertainty belongs there too. These weightings differ completely from yard to yard, and that difference is each company's design philosophy. In most cases it has not been written down.

Preparation on the side of the agent is advancing. Agents that hold a hierarchy and delegate work to sub-agents are now built routinely outside our field, and Section 5.1 is a small instance of the same structure. The layer that gives data its meaning through ontologies is being built out across industries (Section 2.5). What is missing is the yard-specific criteria to be given to it. Only when those are written down can AI move from building fast to choosing what to build. Section 5.2 is where those criteria begin to become available: once agents carry the exchange between organizations, the reasons behind decisions are recorded as they are settled, and a record of reasons is what an evaluation function at this layer is made from.

We expect the largest impact of generative AI to be in this business layer. The effect at the task layer can be measured as working time, and at the process layer as reduced rework. The effect at the business layer appears in the quality of the decision of which ship to build, by which design method, and carrying how much uncertainty. That differs completely from yard to yard, so this layer has no general solution. We have no case at this layer yet. Filling the two layers below by kaizen, while each company turns to writing down its own criteria, is the work we see next.

7. Conclusions

We set out looking for a measure: a way of telling apart the many things called the effect of generative AI. What we propose is to read that effect in three layers (task automation, process automation and business transformation) each judged by its own measure. The task layer is measured in working time, the process layer in rework avoided, and the business layer in the quality of a decision. Knowing which layer a result belongs to tells you what it is worth and what has to come next.

Of the four functions generative AI supplies, only the machine-learning surrogate continues from before; model generation, the interpretation of unstructured data and the orchestrating agent are new. What they have in common is that the person who knows a job no longer needs an intermediary to automate it.

At the task layer, the knowledge every implementation relied on had a written source, and the work of automation became the work of writing that knowledge down. These implementations were carried out in a collaborative research course with industry partners, with practising designers checking the results, and published on open data. At the process layer, design intent has crossed the boundary between tools; the organizational boundary is next, with agents acting on a person's behalf carrying the exchange and recording the reasons behind decisions as they are settled. At the business layer we have no case yet.

The largest impact, we expect, lies in that business layer, though this is a reading rather than a result. What is asked there is whether a yard can write down its own criteria — rule compliance, construction constraints, cost, safety, delivery — which differ from yard to yard. What makes us think it reachable is that the labour of writing knowledge down is itself now shared with AI, and that labour is what blocked knowledge acquisition in the era of expert systems. We will go on filling the three layers from the bottom, one heavy job at a time.

What this paper reports is a prototype: AI built for particular design processes, with particular companies, in what is for those companies a competitive domain. Carrying it further means widening both the span of design it covers and the number of companies it reaches, and that is the work of a national programme. Under BRIDGE, the Japanese government's programme for bridging R&D and Society 5.0, the National Maritime Research Institute called for technology towards the next-generation shipyard and selected a project on AI-assisted design built on 3D design data, NMRI (2026). Led by MTI, with more than thirteen Japanese shipyards taking part and the authors among those carrying it out, that project develops in the collaborative domain the industry shares, and it extends the prototype from concept design through to production design.

Acknowledgements

Part of the work reported here was supported by BRIDGE, the Japanese government's programme for bridging R&D and Society 5.0, under "Technology development towards the realization of next-generation shipyards through the use of AI", commissioned by the National Maritime Research Institute.

References

- AMEMIYA, T. (1991), *Construction of process design system for shipbuilding CIMS*, J. Society of Naval Architects of Japan, pp.843–856
- ANDO, H.; SEPPÄLÄ, L. (2025), *Information architecture in shipbuilding and shipping: Ship lifecycle, digital models, twins, and fragmented data thread*, 24th COMPIT Conf., Pontignano, pp.333–344
- ANTHROPIC (2024a), *Introducing the Model Context Protocol*, Anthropic
- ANTHROPIC (2024b), *Building effective AI agents*, Anthropic
- ANTHROPIC (2025a), *Effective context engineering for AI agents*, Anthropic
- ANTHROPIC (2025b), *Introducing agent skills*, Anthropic
- BRONSON, J.; TEIMOURI, M.; GASPAR, H.; FONSECA, I.; BIERKOWSKA, K.; GINTER, F.; KOELMAN, H. (2025), *A RAG-based LLM approach for data validation and harmonization in ship design*, 24th COMPIT Conf., Pontignano, pp.171–181
- FUNAOKA, K.; KAKUNO, H. (1992), *What is shipbuilding CIMS?*, J. Society of Naval Architects of Japan 759, pp.519–527
- GASPAR, H.M.; BERTRAM, V. (2023), *The use of ChatGPT and similar AI in marine engineering: Limitations and opportunities*, 22nd COMPIT Conf., Drübeck, pp.73–83
- HATAYAMA, K.; ICHINOSE, Y.; WAKITA, K.; MAKI, A. (2026a), *A three-layer context framework for automating ship CAD modeling*, JASNAOE 42, pp.91–96
- HATAYAMA, K.; ICHINOSE, Y.; WAKITA, K.; MAKI, A. (2026b), *Proposal of a vertical automation framework and LLM harness for efficient ship design tasks*, JASNAOE 43
- ICHINOSE, Y.; GASPAR, H.M. (2023), *Interactive ship flow simulation enhanced by neural network model in a web environment*, 37th ECMS Int. Conf. Modelling and Simulation, Florence, pp.155–161
- ICHINOSE, Y.; TANIGUCHI, T. (2022), *A curved surface representation method for convolutional neural network of wake field prediction*, J. Marine Science and Technology 27/1, pp.637–647
- ICHINOSE, Y.; WAKITA, K. (2026), *Toward the development and implementation of generative AI in naval architecture: Redefining ship design processes via a 3-layer model*, JASNAOE 42, pp.79–84
- ICHINOSE, Y.; WAKITA, K.; HATAYAMA, K.; MAKI, A. (2025), *Innovating ship design with generative AI: Initiatives of the OCEANS joint research project at The University of Osaka*, JASNAOE 41
- MIYATA, N.; ICHINOSE, Y.; NAKAUE, D.; WAKITA, K. (2026), *Development of a hull-form performance evaluation model built on a general-purpose CFD surrogate model*, JASNAOE 43

NAKAUE, D.; ICHINOSE, Y.; OHASHI, K.; TANIGUCHI, T.; MAKI, A.; WAKITA, K. (2026), *Development of a U-Net-based surrogate model for predicting hull surface pressure distribution in waves*, JASNAOE 42, pp.829–83

NMRI (2026), Programs for bridging the gap between R&D and the ideal society (Society 5.0) and generating economic and social value (BRIDGE), National Maritime Research Institute, Tokyo

SAHOO, P.; SINGH, A.K.; SAHA, S.; JAIN, V.; MONDAL, S.; CHADHA, A. (2024), *A systematic survey of prompt engineering in large language models: Techniques and applications*, arXiv:2402.07927

SCHREIBER, G.; AKKERMANS, H.; ANJEWIERDEN, A.; HOOG, R. de; SHADBOLT, N.; VAN DE VELDE, W.; WIELINGA, B. (2000), *Knowledge engineering and management: The Common-KADS methodology*, MIT Press, Cambridge

WU, H.; GUO, M.; LI, Z.; DOU, Z.; LONG, M.; HE, K.; MATUSIK, W. (2026), *GeoPT: Scaling physics simulation via lifted geometric pre-training*, 43rd Int. Conf. Machine Learning (ICML)

Advanced CFD Simulations for a Wave-Energy Powered Ship Propulsion System

Sven Albert, Simuneer, Altdorf/Germany, sven.albert@simuneer.de
Romain Hottois, Cadence Design Systems, Brussels/Belgium, hottois@cadence.com
Pierre Weyne, Bluefins, Brest/France, pweyne@bluefins-systems.com

Abstract

This paper describes a zero-emissions wave-devouring propulsion system for ships. The WaveDrive system, a whale-inspired propulsion system, consists of a hydrofoil attached to the stern of the ship and converts the pitching motion of the ship into thrust. High-fidelity CFD simulations were performed to support the design of the system and to furnish a hydrodynamic knowledge base for sensor-based control of the system's motion under various sea conditions.

1. Introduction

Shipping is essential to global trade, as it transports nearly 90% of all traded goods. Large ships consume between 20 to 70 tons of fuel daily, which translates to ~15 million Euro in annual fuel costs. This level of fuel consumption results in emissions of up to 75,000 tons of CO₂ equivalent per vessel per year.

The global shipping industry accounts for about 3% of greenhouse gas (GHG) emissions, making it roughly comparable to the national emissions of South Korea or Germany. It is crucial to implement systems and technologies that can decarbonize shipping and make transportation more sustainable. IMO, www.imo.org, has therefore developed a roadmap to net-zero carbon footprint for shipping by or around 2050, and assorted frameworks to implement this decarbonization process.

2. Bluefins' Bio-Inspired Propulsion System

The idea of using bio-inspired fins to harness wave energy for ship propulsion is not new. *Yasukawa and Ishikawa (2017)* presented a concept for wave-assisted propulsion for a catamaran, giving reference to previous related work: "Wave devouring propulsion using hydrofoils was studied by *Jakobsen (1981)*. In Japan, *Terao (1982)* studied the possibility of the wave devouring propulsion in model tests. *Isshiki et al. (1989)* substantiated the wave devouring propulsion by attaching a hydrofoil to the fore part of a ship with 15.7 m in the length in actual seas. In 2008, the wave devouring propulsor ship "Mermaid II" was designed and built as an experimental ship. It succeeded in sailing from Hawaii to Japan using only wave energy, *Terao (2008)*." *Xing and Yang (2023)* give a more recent overview of wave-devouring propulsion.

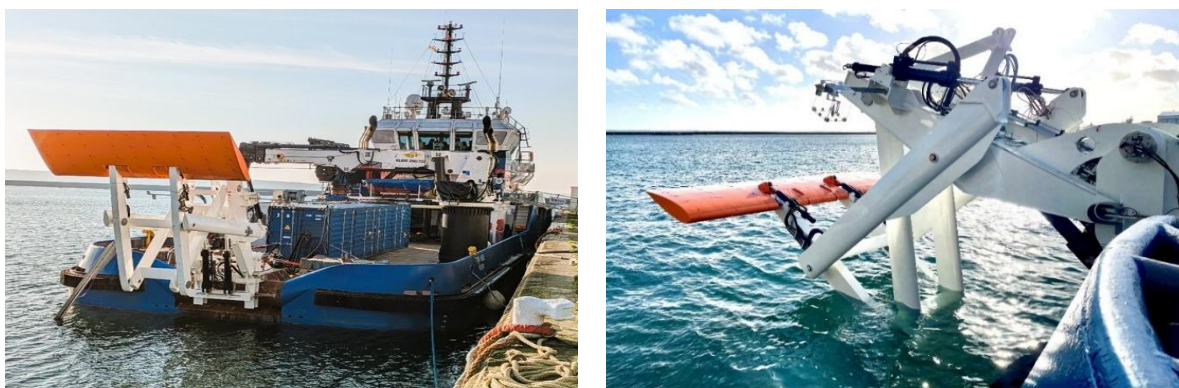


Fig.1: Bluefins WaveDrive system as installed for sea trials

More recently, the French company Bluefins, <https://bluefins-systems.com>, has developed a wave-

devouring propulsion device, named WaveDrive, Fig.1. The device is inspired by the fins of whales, to reduce GHG emissions and fuel operating costs. The system and supporting CFD-based Digital Twin model will be described here. Key features of the WaveDrive system include:

- Articulated hydrofoil attached to the stern of the ship
- Activated by the wave energy transmitted to the ship
- Controlled by real-time algorithms
- Retractable for security and efficiency purposes
- Added benefit: Pitch damping

Preliminary hydrodynamic investigations for the continued development of the WaveDrive device started in 2022 and moved from model tests over numerical simulations to full-scale trials, Fig.2.

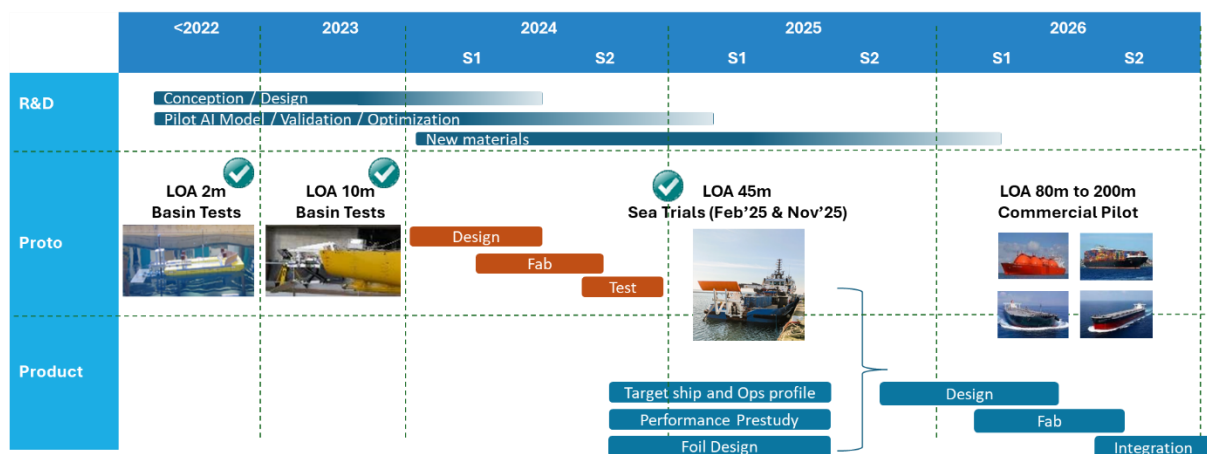


Fig.2: R&D roadmap for the WaveDrive development

First sea trials were conducted in February 2025, off the coast of Brittany, Fig.1 (left), to validate the control command, safety processes, propulsion and the hydrodynamic model. In November 2025, a second sea trial campaign focussed on performance (fuel gain) and the improved control command architecture. 2026 shall see the first commercial pilot project.

The WaveDrive device was developed in partnership with [Ifremer](#), a French research institute. The whale-inspired propulsion system consists of a hydrofoil attached to the stern of the ship and converts the pitching motion of the ship into thrust, Fig.3. This technology can be integrated into newbuild vessels or retrofitted.



Fig.3: WaveDrive fin system in waves in operation (snapshots of simulation)

3. CFD Analysis

Simulations using the [Fidelity](#) CFD (Computational Fluid Dynamics) software were performed to design and optimize the WaveDrive propulsion system under different operating and real-world wave conditions. The software's advanced features facilitate detailed analysis of the flap, the system dynamics, and the prediction of hydrofoil loads to optimize the propulsion.

3.1. Case description

Key objectives of these preliminary numerical investigations were:

- Model the WaveDrive system motions on a vessel with fixed forward speed
- Measure foil loads

The Digital Twin model of the hull was detailed, including all major appendages. For the WaveDrive, we used pre-calculated imposed motions for each component.

3.2. Mesh set-up

The Digital Twin for the hydrodynamic simulations needs to represent a complex system with several components that have high-amplitude movements, Fig.4. We therefore use overset meshing. Each movable component had an independent overset mesh, Fig.5. In total, that resulted in a cell count of 10,600,000 cells for the total model. (Exploiting port/starboard symmetry, only half of the ship and its appendages was modelled).

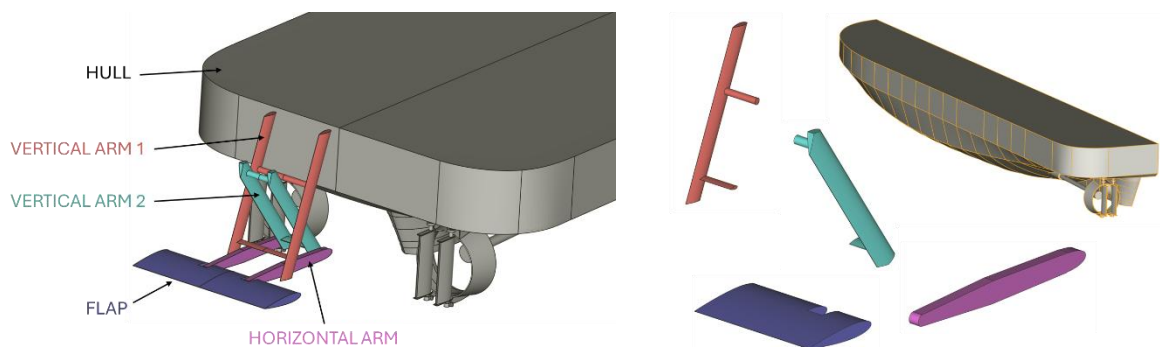


Fig.4: Each component of the mechanical system was meshed separately

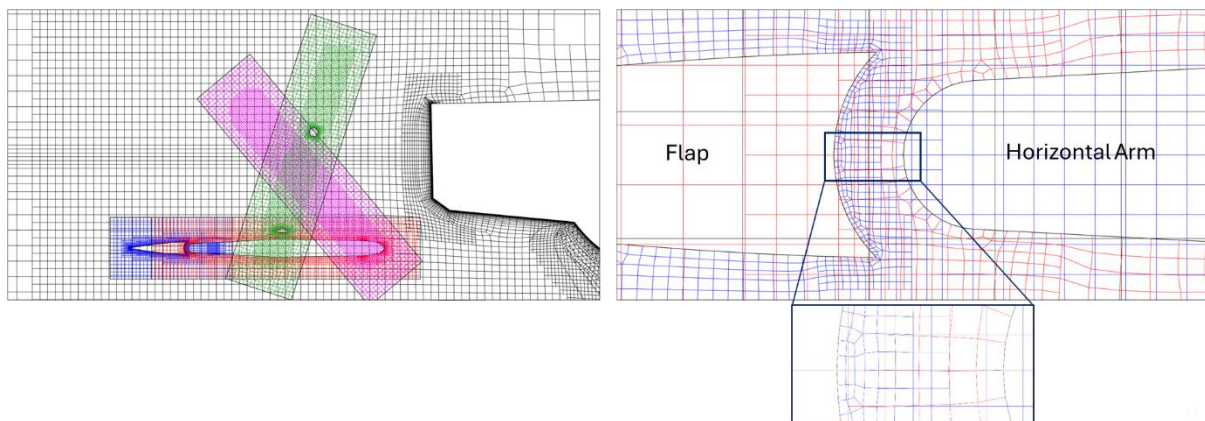


Fig.5: Set up of the total mesh with 10,600,000 cells

The hull was meshed using the new C-Wizard included in Fidelity 2025.1. The C-Wizard is an automated tool designed to rapidly set up CFD simulations for naval architects and marine engineers. It reduces engineering time by automating the entire process – including mesh generation and numerical parameters – for various naval architectural applications like resistance, seakeeping, and propulsion, enabling faster, high-fidelity results.

The C-Wizard features a [Rhino](#) plug-in for direct CAD importing and connects to Fidelity's mesh and solver environment. The streamlined process with minimum human user input has cut set-up time for such analyses drastically in our experience.

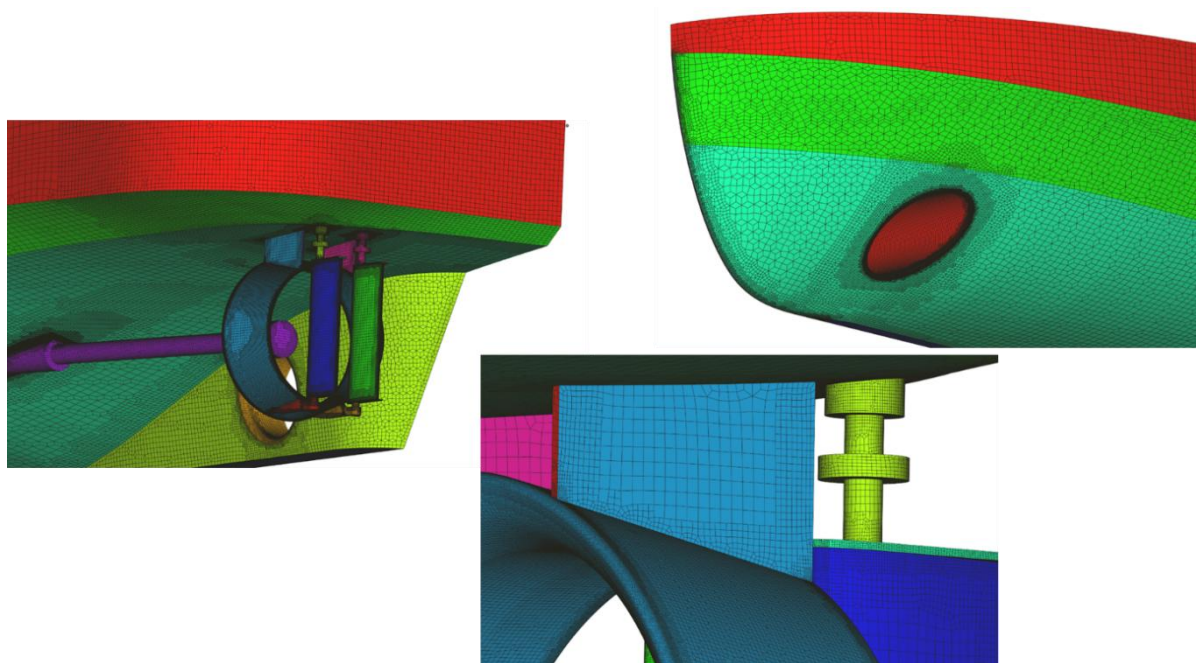


Fig.6: Hull mesh with 8.8 million cells

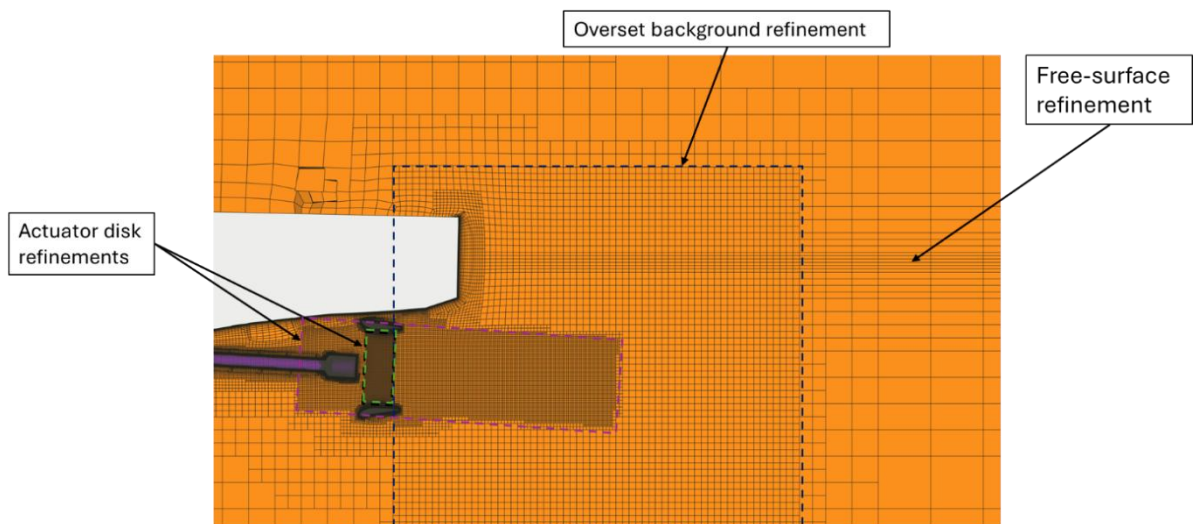


Fig.7: Mesh refinements

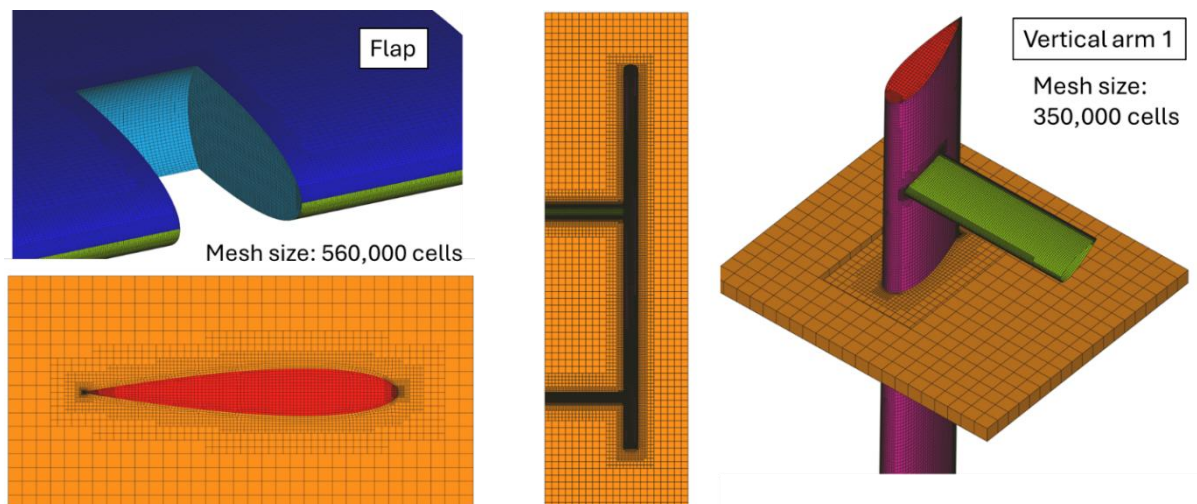


Fig.8: Meshing of appendages

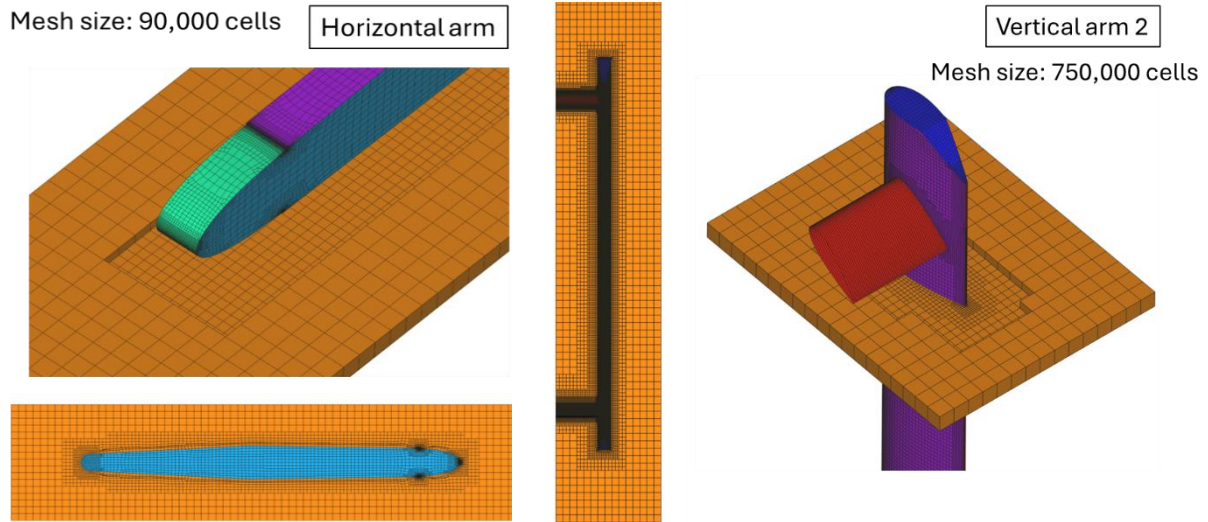


Fig.9: Meshing of appendages

The main mesh discretized the hull with 8.8 million cells, using local mesh refinement for cut-outs and fixed appendages, Fig.6. For computational efficiency at still high fidelity in the regions of interest, we used an actuator disk model for the propeller. To ensure sufficient resolution, free surface, wake area near the hull and part around the actuator disk used local mesh refinements, Fig7. The movable appendages were manually meshed in independent Fidelity setups, Figs.8 and 9.

3.3. Hydrodynamic simulations

The WaveDrive's system dynamics were simulated in [FineMarine](#), a high-fidelity CFD simulation software suite,. Key parameters for the simulation were:

- Modelling: The flow is unsteady 3D Navier-Stokes. A steady initialization is performed. The $k-\omega$ -SST turbulence model with wall function is used.
- Boundary conditions: The solid patches are set to “Wall-function” except for the deck which is set to “Slip” boundary condition.
- Adaptive Grid Refinement: Overset grid continuity is used to ensure grid continuity between the background and overset meshes.
- Motion: The advance velocity of the vessel is imposed to 3.6 m/s (7 kn). Unsteady motions are imposed following a user-defined input.
- Numerical settings: The timestep is set so that 300 points per period are computed.

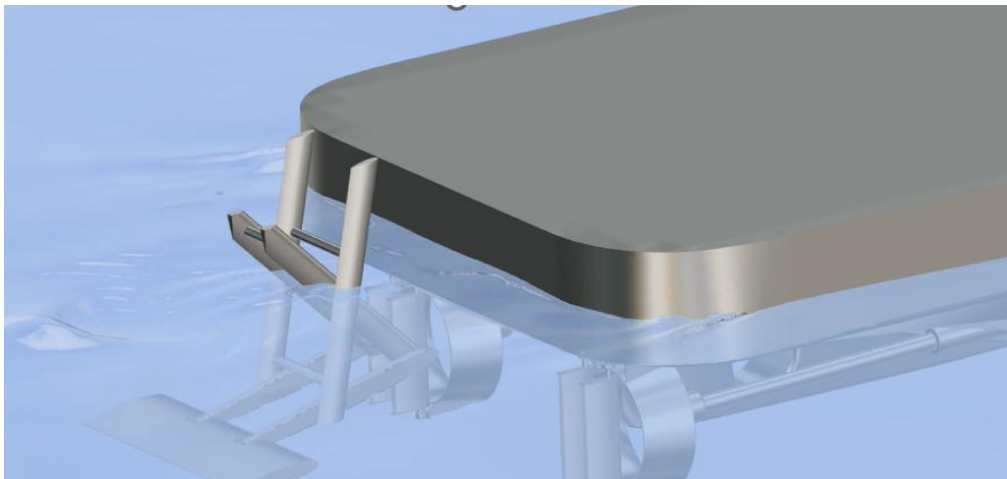


Fig.10: Snapshot of simulation

The simulation yields the free-surface deformation, Fig.10, as well as the forces on the hull and the foils. Table I gives simulation times on a parallel computer with 470 CPUs (2X Xeon Platinum 9242).

Table I: Physical times simulated and simulation times required

	Physical time	Simulation time
Steady	112 s	4.5 h
Unsteady	65 s	15.5 h
Total	177 s	20.0 h

4. Conclusion & Outlook

The simulations are an important milestone in the continuing development of the WaveDrive design. The next steps will include an extension of the simulation model to allow free motions in waves and a Digital Twin for a controller to respond to motions or sensor variations.

References

- HOTTOIS, R.; WEYNE, P. (2025), *Simulate Bluefins system's dynamics with FineMarine*, Cadence User Meeting, Brussels
- ISSHIKI, H.; HATTA, K.; TERAOKA, H. (1989), *Research and development of wave devouring propulsion by oscillating wings*, Bulletin Soc. N. A. Japan, No.719, pp280-288. (In Japanese)
- JAKOBSEN, E. (1981), *The foil propeller, wave power for propulsion*, 2nd Int. Symp. on Wave and Tidal Energy, BHRA Fluid Engineering, pp.363-368
- TERAOKA, Y. (1982), *A floating structure which moves toward the waves (possibility of wave devouring propulsion)*, Trans. West-Japan Soc. N. A. 184, pp.51-54 (In Japanese)
- TERAOKA, Y. (2008), *Development of the wave devouring propulsor ship "Mermaid II"*, Conf. Proc. JASNAOE, No.7E, Paper-No.2008E-G7-3 (In Japanese)
- XING, J.; YANG, L. (2023), *Wave devouring propulsion: An overview of flapping foil propulsion technology*, Renewable and Sustainable Energy Reviews 184, <https://www.sciencedirect.com/science/article/pii/S136403212300446X>
- YASUKAWA, H.; ISHIKAWA, T. (2017), *Improvement of Propulsive Performance of a Catamaran in Waves by a Biologically Inspired Hydrofoil*, 11th HIPER Conf., Zevenwacht, pp.151-159, http://data.hiper-conf.info/Hiper2017_Zevenwacht.pdf

Seamless Structural Design Workflow

Francesco Oneto, NAPA Ltd., Camogli/Italy, francesco.oneto@napa.fi

Abstract

Structural design is often reduced to the result of a strength calculation. This paper argues that such a view is only partial: the design of a ship structure requires teams of specialists with different and complementary skills, whose work must be harmonised and whose information flow must be kept consistent from the concept stage to the workshop. The paper first reviews what a structural design environment must deliver: weight and centre of gravity, bill of materials, document, 2D drawings, interfaces towards classification software and finite element models. It then identifies consistency as the single key requirement, in the sense that structural property must be derived from a single source of truth. Two developments are described against that requirement. NAPA Drafting links each 2D drawing to the 3D model, so that geometry and scantling annotations remain consistent by construction. NAPA FEM allows a finite element model to be generated, constrained, loaded and solved inside the NAPA Designer environment, removing the discontinuity that arises when the mesh is exported to an external preprocessor. A hull girder global analysis is then used as a worked example, showing how the data already held in the NAPA Loading Condition task can be used, through C# scripting, to build and verify a balanced global load case.

1. Introduction

Ships and ship structures have been conceptually synonymous for centuries, before the introduction of the engine and of onboard auxiliary service systems, ships consisted mainly of their structure.

Looking for a definition of what a ship structure is, we can say that the structure is what gives the ship its physical shape.

Ships are self-sufficient floating units whose structure not only ensures sturdiness and creates a watertight barrier but also serves as the support for all the other equipment installed on board.

The boundary lines within which every major drawing is developed – capacity plan, general arrangement and, obviously, steel plan – are structures. Simply turning off the “steel structure” layer on a CAD drawing shows how the remaining components suddenly appear to float in a meaningless space. Despite the fundamental role of the structure, its design sometimes appears to be a secondary element, as it is taken for granted within the naval design process. For instance, the cost of the combat system of a military unit, or of the luxury fit out of a mega-yacht, is vastly higher than that of the structural framework supporting them.

The scope of the present paper is to show that reducing structural design to the result of a strength calculation is a very partial point of view, the ship is an enormous creature, thousands of square metres of steel plate and kilometres of extruded profiles. For decades, designers have faced the challenge of finding a proper way to manage the design, in order to judge its buildability, predict its behaviour and transfer all information from the concept to the workshop. The structural design teams involve numerous specialists with different and complementary skills, those design abilities must be harmonised to reach the final result.

We'll try to illustrate not only the structural design flow but the whole the scenario in which it involves the technical, environmental and human aspects from real shipbuilding experiences.

2. The Nature of the Structural Design Process

2.1. A double geometrical and numerical nature

Trying to define the path along which a structure evolves, we immediately see that it is not a linear one. Structure has a double nature, geometrical and numerical. Even if in the end the two are consequential, at a first stage the primary goal is to estimate its weight and centre of gravity as essential components of the assumed displacement. As the time required to get a reliable structural 3D model is rarely available, those values cannot be derived directly and entirely from the actual ship geometry and must be assumed using different criteria.

An essential part of structural design is therefore to keep track of the differences between the preliminary assumptions and the resulting output of the design under development, and to check whether they are converging or not. On the next chapters the other aspect of the structural design will be illustrated more in detail.

2.2. Drawings, deliverables and industrial practice

A ship is a three-dimensional structure, but it is almost entirely composed of elements having only two relevant dimensions. A flat drawing approach therefore worked, and still works, very well to visualise the majority of the components in a comprehensive and intuitive way. It results in the production of hundreds of drawings containing all the relevant information used by the various intended audiences: Classification Society, other designers, workshop specialists, and welders. The drawing finalization is a step that usually involves the largest part of technicians and still represents one of the critical steps where the continuity of the information may be broken.

The move towards 3D model-based approval through the Open Class 3D Exchange (OCX) standard, *Astrup et al. (2022)*, *Son et al. (2022)*, is gradually reducing the role of the 2D drawing towards the Classification Society, *Luli et al. (2025)*, but the drawings are still needed for production and, as the outcome of the design, to be sent to the ship owner and classification societies to be archived ashore and to be used onboard during the operation of a ship, as required by SOLAS regulation II-1/3-7 on as-built construction drawings to be maintained on board and ashore, *IMO (2004)*, and by the Ship Construction File of regulation II-1/3-10 for bulk carriers and oil tankers, *IMO (2009)*.

Strength calculations must be performed within a reasonable amount of time, in order to complete the geometrical attributes of the structures, update weight, centre of gravity and verify the strength expected behaviour to avoid radical changes at the next stages.

Material purchase must also be considered: plates and stiffeners have to be ordered months in advance, and some tool and procedure must therefore be able to “count the structure” produce a list of the elements to be ordered and to keep it updated at any stage.

A very underestimated role is played by those side aspects of the design for which the structure represents the input data, such as insulation and painting estimates. Apparently secondary, they are very practical, tedious and mandatory aspects that involve large investments at the production stage.

A last consideration: because of the dimensions of the objects involved and of the engineering challenge they pose, shipbuilding is a very conservative field, full of both worldwide shared and customised conventions. Ships are still designed by humans, relying on their own working experience, each shipyard has different facilities, skills, preferences, traditions, well-proven solutions and limitations. That sum leads to a practically endless number of different ways of solving similar problems, not necessarily optimised in an absolute sense, but more frequently following the unwritten golden rules of structural design.

The challenge for software developers and structural design consultants is to be able to gradually substitute existing pieces of consolidated company procedures with more efficient ones allowed by modern tool. The substitution must be smooth and try to use same input and produce some output just more efficiently.

The human aspects of the dynamics, roles and competences, inside a big shipyard plays an essential part in defining the success of a transition from old to new way of working.

3. Requirements for a Seamless Structural Workflow

3.1. Required capabilities

Considering all the above, the design of a ship structure needs tools and procedures capable of, *Masui et al. (2023)*:

- Collecting the greatest possible amount of data into a single source of information.
- Producing numerical outputs – weight, centre of gravity, areas, bill of materials – in a format that is immediately comparable with the expected results from previous buildings.
- Covering at best the production of the 2D drawings, including both geometry and scantling annotations.
- Allowing the export of geometry and structural properties to classification society 2D scantling software for preliminary calculation.
- Minimising the manual work required to update model to the most recent configuration.
- Supporting strength calculation through FE models and solvers at different stages.

The key word in all of this is: data consistency, just an example:

A 6 mm steel plate on a bulkhead has a weight and a cost, it must be counted in order to be ordered and at some point painted, its strength must be verified against various loads and its shape and thickness must be represented and annotated on the corresponding bulkhead drawing.

From the process point of view, the ideal solution is setting up a single place where this 6 mm thickness would be defined. In the next chapter we'll try to explain how, even if this appears self-evident at a theoretical level, reality shows how complex is to achieve it in practice.

4. From 2D Scantlings to the 3D Model

4.1. The section-based scantling approach

Considering the 2D section scantling approach via the selected classification society dedicated software: Nauticus Hull, Leonardo Hull, BV Mars, LR RulesCalc etc. traditionally used to define the preliminary dimensions of plates and longitudinal stiffeners, procedure consists of:

- Sketching or importing a certain number of structurally relevant sections along the ship.
- Add prescriptive loads for deck tanks etc.
- Solving the sections, defining thickness, material and profile dimensions.
- Harmonising the choices made for the single sections before they are transferred to the 3D environment.

The sections chosen to represent the so called “typical aspect” of the structure in a certain area, but they may not correspond to any real geometry in the ship. This means that, even if a direct backward compatibility from 2D sections to the 3D model is available, the perfect geometrical correspondence cannot always be found and a set of decision must be taken.

There can be situations where it is preferable to minimise stiffener size changes along the same profile trace, even if this goes against a strength-to-weight optimisation goal. It may happen that a certain stock of plates has to be used in the shipyard, even if those plates are thicker than the ones needed.

Harmonizing those choices requires experience and skills and a special attention to not pure technical aspects.

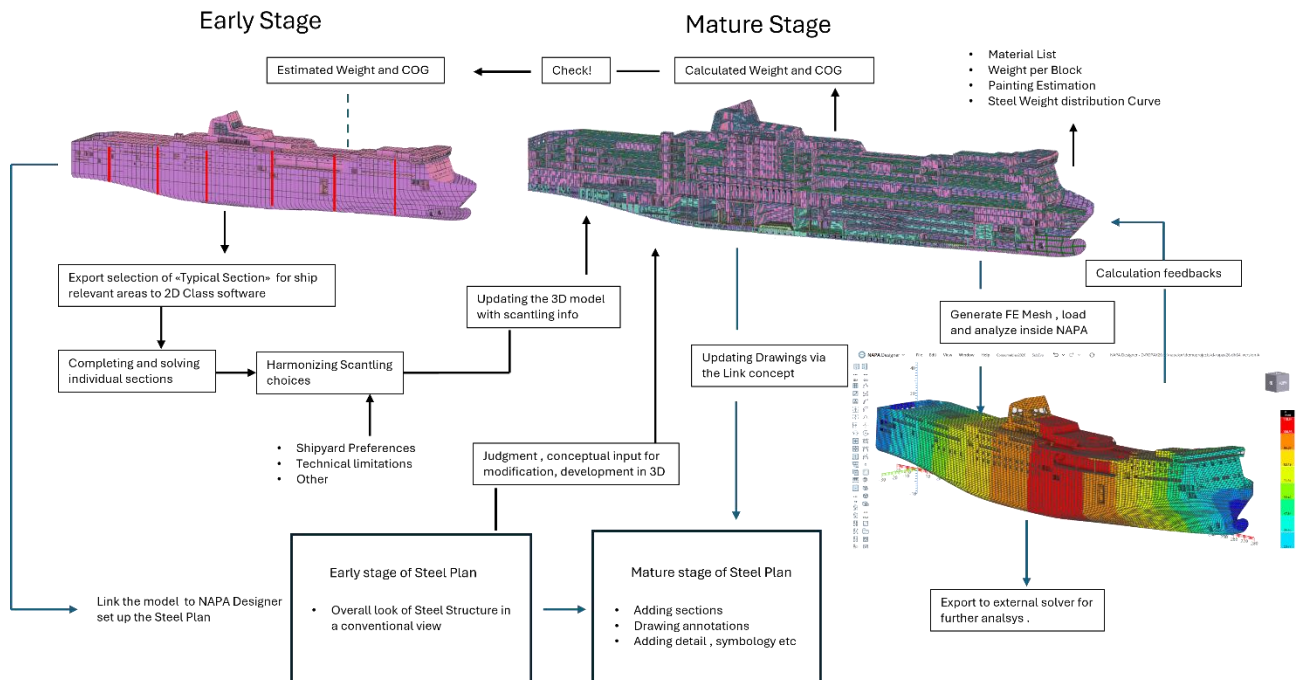


Fig.1: From 2D section scantlings to the 3D structural model

4.2. Discontinuities in the real design flow

In real shipbuilding, the design flow of a ship may not be continuous for various reasons, some of them completely independent from technical problems: a sudden higher priority may move people to work on another ship, and a crucial delayed decision may put the work done on hold for weeks or even months.

In other scenarios, the shortage of time pushes designers to unlink the steel drawings from the 3D model and to finalise them in an independent way; modifications must then be reported manually from the 3D model into each affected section, increasing the risk of errors.

Last, but not impossible scenario, the owner may simply not like the aesthetic of a certain technical choice and the whole area must be refactored.

Restarting the work on an old project where scantlings, 3D model data and 2D drawing annotations have been left inconsistent as stored into independent documents, generates an enormous waste of time. To summarise, in order to ensure a smooth workflow, it is essential to have a tool that assists the designer in:

- Making decisions on the basis of a global overview of the ship structure.
- Generating a 3D model in a short amount of time flexible enough to be modified without requesting an onerous amount of work.
- Building the source of information at an early stage to produce graphical and numerical Output.

5. NAPA Drafting: The First Gamechanger Tool

NAPA Steel represents one of the most advanced solutions in 3D structural design, from the very early stage up to the level of detail needed for classification drawings and more. Geometry and steel properties can be loaded, topologically connected modified, sorted and utilized to produce a huge variety of graphical and numerical input. It is a well-known tool used by the major shipbuilder around the globe.

It has recently been completed with an additional tool, called NAPA Drafting, which manages the drawing generation part, allowing the user to work into a familiar AutoCAD interface.

Each drawing – deck plan, section, elevation, detail – can be visualised as a 2D representation of the correspondent 3D model which is physically linked to, *Son et al. (2022)*.

The link concept means that both the purely graphic part of the drawing – line styles, colours, line thickness – and the information about thickness, profile dimensions and many other data are part of the drawing itself.

The user can annotate each drawing simply by asking the system which property must be displayed. This is a revolutionary tool in structural design, as the consistency between 3D data and 2D drawing is ensured by the link and does not need to be double-checked.

Every modification made in the 3D model, both in terms of geometry and or scantling, is directly reflected on the graphic side, where it is visualised compared to the actual, checked and then accepted or discarded by the user. As result both graphics and annotations are always consistent with the content of the 3D structural model connected.

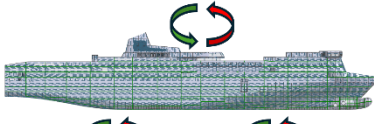
Traditional Workflow 2D to 3D	Paradigm Shift Export 3D to 2D	The Link to Model Concept
Shell Expansion 2D Transversal Sections  2D Elevations 2D Deck Plans	Shell Expansion 2D Transversal Sections  2D Elevations 2D Deck Plans	Shell Expansion 2D Transversal Sections  2D Elevations 2D Deck Plans
Obsolete way of working ❌ Fully manual consistency ❌	Geometrical consistency ✓ Updating problems, loss of data ❌ No metadata transfer ❌	Geometrical consistency ✓ Updating management no loss of data ✓ Selected Metadata transfer ✓

Fig.2: NAPA Drafting: 2D drawing linked to the 3D structural model

Seen from the industrial perspective, NAPA Drafting allows expert designers, who are often used to read drawings, to have a global overview of the Steel plan at every design stage. More people can check different areas of the ship at the same time, give feedback to the 3D modellers and let the drawing creation, be part of the ongoing design, not only the final documentation to be submitted to classification societies.

Until this point, the 3D Model in NAPA Steel ensures the consistency among, geometry, weight, bill of materials and drawings. The same model is also the source of the OCX file exchanged with the

classification society, whose usage is being extended beyond the structural approval itself, *Son et al. (2025)*.

6. The FEM and FEA Dilemma

6.1. A discontinuity in the process

One point still remained open. Because of its complexity, the only way to verify the structural behaviour of a ship is to use finite element analysis FEA, which is mandatory documentation both for the Class and for the designer's confidence. One may notice that, until a not so remote past, the creation of those models was so time-consuming that they were mainly used to confirm the result, sometimes when the building was already at a certain stage.

Nowadays, the generation of a FE model can be obtained with a reasonable effort already at the early stages of the design. The most recognised FE software packages, widely used in shipbuilding, are capable of meshing a geometry imported from the structural 3D model, saving a consistent number of working hours, *Puurula et al. (2024)*, *Son et al. (2025)*. Nevertheless, exporting the mesh from the 3D source model to an external preprocessor creates a discontinuity in the process.

At some point the FEA must be carried out, this means that the structural configuration is “frozen” for a while and somehow exported to the FE software, after which the two models – the FE one and the 3D structural one – continue to be developed in parallel to the next stages of detail but are no longer linked, *Hulkkonen et al. (2019)*.

Selecting the proper timing is not really straightforward: it may depend on the contingency, on the maturity level of the 3D structural model and on other unpredictable environmental variables.

Experience has shown that after imported there is always a certain amount of work needed to fix the model in the new preprocessor before it can be analysed. The skills required for the proper usage of those tools are high and often reserved to a team of FEM specialists.

6.2. The cost-benefit dilemma

Designers and managers often face a cost-benefit dilemma:

- The mesh directly exported from NAPA 3D structure could be not good enough to be immediately used: it may contain duplicated elements, unconnected nodes and various unsolvable mesh issues difficult to spot from NAPA side.
- The evolution of the project requires minor or major changes to the structural geometry reported in the 3D structural model.

Considering the time spent by the specialists in fixing the FE model, is there a real benefit in starting the work again from a new exportation? Is an analysis carried out based on an outdated structural configuration be considered reliable enough to validate the structural solution proposed? Will it cause formal issues if reporting results that are not fully matching the geometry of Steel drawings and 3D model?

6.3. Meshing and solving inside NAPA Designer

NAPA has developed and improved a meshing tool capable of generating a FE mesh composed by shell and beam element directly from the corresponding NAPA Designer model. Mesh size, idealisation tolerances and logic can be fine-tuned, allowing the usage of the same 3D structural model both for global model mesh and a detailed mesh suitable for fatigue analysis, on the basis of the design intent.

The dilemma shown above still remained unsolved, aware of that, NAPA introduced a second great revolution in the structural design workflow: the FE model can be now created, constrained, loaded and solved inside the same environment.

This is an enormous help in cutting the bottleneck caused by the multiple export, import, solvability check process: an experienced NAPA modeller, who knows the ship structure both in terms of geometry and of model technicalities, can generate the mesh following FE specialist advice and check its solvability before exporting it to an external preprocessor.

The mesh generated in NAPA is fully consistent with the configuration of the corresponding 3D structural model, can be massively modified and fixed using dedicated tools, and requires a really reasonable amount of time to be generated, constrained, loaded and solved. A global FE model of medium-size ferry ship suitable for hull girder strength (~255000 shell elements) can be generated from the correspondent Napa steel model in ~20 minutes.

Using the most recent releases, the file exported from NAPA can contain not only the mesh geometry and its properties, but also several helpful options, such as element groups defined by various criteria, constraints, loads and most important the guarantee that it will not create issues in running the analysis.

7. Hull Girder Analysis Inside NAPA

7.1. Required loads and available data

The challenge is then shift to another level: can a calculation carried out inside NAPA give useful results beyond the mesh solvability? Can the seamless process to be extended until FE calculation results inside NAPA software. To answer this question, let consider the case of a Hull girder analysis calculation for a global model and clarify:

- The needed input loads.
- The overall data contained in a NAPA database, looking outside the Steel environment.
- The NAPA tools available to constrain and load the model.

Following the most common rule prescriptions, the hull girder model must be subject to:

- External hydrostatic pressure on the external shell.
- Loads in tanks.
- Local loads defined by masses.
- Weight of the steel structure.
- Weight of the items on board not mentioned above.
- Effect of the wave, in hogging and sagging conditions.

As the ship is floating, the effect of the loads should result in a self-balanced system of forces and in a certain value of resulting bending moment. The model should be constrained on three nodes with specific location and rotation/translation locks, ideally, the total reaction force on those nodes is null; in practice, it must remain below a specific ratio.

Thinking a bit out of the box, the NAPA Loading Condition task is the place that already holds all the information:

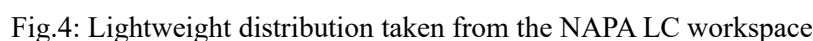
- Floating position: trim, draught and buoyancy distribution.
- Liquid loads and mass item loads for a specific ship loading condition.
- Steel weight and weight distribution curve, which can be derived directly from the corresponding steel model and be part of the lightweight.
- The remaining weight, defined as lightweight items.

A 3D visualization of a multi-layered structure, possibly a building or a large container. The structure is composed of several layers, with the top layer being a flat, light blue surface. Below this, there are multiple layers of a yellow, grid-like material. Red vertical lines are visible, extending from the top layer down through the grid layers, suggesting a vertical flow or a series of columns. The structure is shown from a perspective view, with the layers receding into the distance.

Fig.3: Loads defined in the LC workspace applied as nodal forces to the tank boundaries identified by their topology

Actual tool for FEM model loading are limited to pressures and nodal forces, the usage of C# scripting becomes then mandatory, but it offers the possibility of utilising the strongest NAPA topological functionalities. As example, NAPA automatically can recognize which structures bound a tank, it knows the total weight of the tank content and the corresponding filling level, the script can select the FEM nodes involved and apply the shared force.

The lightweight distribution curve can be discretised, based on the mesh distribution, as shared nodal forces acting at specified sections, and the hydrostatic pressure can be calculated by an algorithm reproducing the pressure field.



The wave load distribution is ruled by a polynomial expression that can be applied to the external shell nodes as a shared force type at specific sections.

Because of the mesh size, the accuracy of the applied forces and pressures can differ from the target value. That discrepancy can be corrected automatically by a factor that the C# script applies to each of the forces and pressures after its computation.

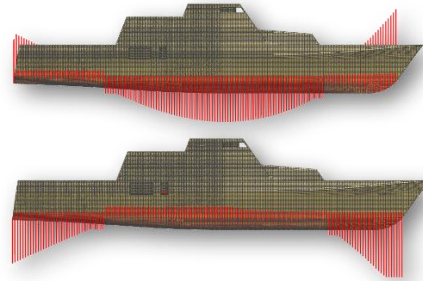
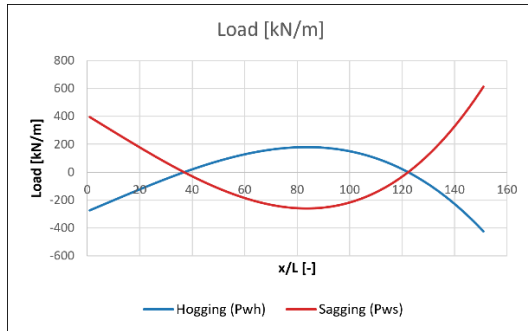


Fig.5: Wave load polynomial distribution applied as shared nodal forces on sections

7.3. Verification of the results

There are three aspects that the user must verify before the results can be considered reliable:

- The constraint reaction stands below the required ratio.
- The bending moment caused by the sum of the pressures and nodal forces applied to the FE Model, matches the intended bending moment: still water only, still water and wave, external bending moment by enveloping curves and so on.
- The deflection shape should look like the theoretical deflection, or at least the differences should be clearly explicable.

A procedure and a set of dedicated C# scripts have been prepared to check what is described above:

- Apply one set of loads at a time and verify that the resultant force on the constraint matches the expected value, for instance buoyancy only, or tank loads only.
- Read the applied pressures and nodal forces, recreate the load distribution curve by reverse engineering, integrate it twice and compare the result with the shear distribution and bending moment distribution coming from the NAPA Loading task; similarly, the same can be done considering the wave effect.
- Let the system read the inertia values at a specific set of model sections, calculate the deflection and compare it with the displacement of a set of selected nodes around the neutral axis.

All values can be compared both graphically and numerically, using Excel spreadsheets automatically generated by the scripts. In terms of quality and speed, the results are quite good.

8. Conclusions

Structural design is not the result of a strength calculation alone. It is the harmonisation of geometry, scantlings, weight, output documentation, procurement data and analysis, produced by teams with complementary skills and under industrial constraints that are rarely purely technical. The recurring cost in this process is not the difficulty of any single task, but the loss of consistency between the different representations of the same object.

The two developments described in this paper address that cost at its two main boundaries. NAPA Drafting removes the boundary between the 3D model and the 2D drawings, so they become part of the

ongoing design instead of a parallel deliverable to be reconciled manually. The generation, loading and solving of the FE model inside NAPA Designer removes the boundary between the structural model and the analysis model, so that the mesh no longer becomes an independent model as soon as it is exported.

The hull girder example shows that the data required to build a balanced global load case is already available in the NAPA database, within the Loading Condition task, and that C# scripting, combined with the NAPA topological functions, can turn it into constraints, pressures and nodal forces that are verifiable against the shear force and bending moment curves of the same loading condition. Results are obtained within a reasonable time and with a quality that makes them usable as design feedback, and not only as a check of mesh solvability.

None of this removes the need for specialist judgement, for independent verification, or for external solvers when the type of analysis requires them. What it does remove is the duplicated preparation work and the manual reconciliation that, in practice, decide whether a structural workflow is seamless or not.

References

- ASTRUP, O.C.; AAE, O.; KUŠ, T.; UYANIK, O.; BITERLING, B.; BARS, T.; POLINI, M.; VIJAYA, G.; YU, K.; SEPPÄLÄ, T.; SON, M.-J. (2022), *Moving towards Model-Based Approval – The Open Class 3D Exchange (OCX) Standard*, Int. J. Maritime Eng., Part A
- HULKKONEN, T.; MANDERBACKA, T.; SUGIMOTO, K. (2019), *Digital Twin for Monitoring Remaining Fatigue Life of Critical Hull Structures*, 18th COMPIT, Tullamore, pp.415-427
- IMO (2004), *As-built Construction Drawings to be Maintained on Board the Ship and Ashore*, MSC/Circ.1135, Int. Maritime Org., London
- IMO (2009), *Guidelines for the Information to be Included in a Ship Construction File*, MSC.1/Circ.1343, Int. Maritime Org., London
- LULI, F.-C.; SON, M.J.; ASTRUP, O.C.; MADSEN, A. (2025), *3D Model-Based Approval in Reality*, COMPIT Conf., Pontignano
- MASUI, T.; SEPPÄLÄ, T.; SON, M.-J.; HUOTARI, J. (2023), *Collaborative Single Model Design Platform for Ships of Tomorrow*, COMPIT Conf., Drübeck
- PUURULA, J.; PUDD, M.; GUŠANI, S. (2024), *Advantages of Model-Based FEA Reporting Integrated with OCX Approval*, ICCAS Conf., Genoa
- SON, M.-J.; SEPPÄLÄ, T.; MERIKANTO, J.; AAE, O.; ASTRUP, O.C. (2022), *Utilization of OCX as Part of 3D Model-Based Approval in Ship Design Process*, COMPIT Conf., Pontignano
- SON, M.-J.; SEPPÄLÄ, L.; ONETO, F.; KWON, B.J. (2025), *The Extension on the Usage of OCX Beyond 3D Structural Approval*, COMPIT Conf., Pontignano

Field Evaluation of the OCX-Based XR Inspection Tool for Ship Structures

Mladen Radonić, 3D BROD d.o.o., Pula/Croatia, 3dbrod@3dbrod.hr
Jan Obić, ULJANIK Brodogradnja 1856, Pula/Croatia, jan.obic@uljanik.hr

Abstract

This paper presents the current development state and on-site validation of a novel extended reality (XR)-based inspection tool designed to support on-site inspection processes in shipbuilding industry. The proposed solution addresses limitations of conventional inspection workflows relying on the 2D drawings. The developed system was evaluated and validated in representative inspection use cases defined in collaboration with a partner shipyard. Evaluation focused on usability, task performance, and operational feasibility within realistic working environments. The system functions as a support tool for engineers, enhancing decision-making rather than replacing human expertise, consistent with emerging human-centric approaches in industrial digitalization. The paper discusses implementation challenges, validation methodology, and insights from real-world deployment.

1. Introduction

This paper presents the early validation of an innovative eXtended Reality (XR) application designed to support inspection and verification processes in shipbuilding environments adopting 5th generation industry concepts. The application is primarily developed for, but not limited to, use with Meta Quest 3 XR headset devices and is being further advanced within the ocXRayz™ development project, currently progressing beyond the demonstrated Technology Readiness Level (TRL) 6.

A key aspect of this work is validation and qualification of the system under real operational conditions. In this respect, the long-standing professional collaboration between the developer company 3D BROD d.o.o. and the partner shipyard ULJANIK Brodogradnja 1856 d.o.o. providing a natural framework for conducting structured field evaluation. The developer's ship design background, combined with the practical experience of shipyard engineers and technical personnel in production, created a reliable basis for realistic validation of the system in representative industrial scenarios. The approach adopted in this work emphasizes realistic field validation, focusing on user interaction, task efficiency, and system usability rather than purely laboratory / office - based demonstrations.

2. Participants on the Validation

2.1. About the Partner Shipyard Collaborating on Validation

ULJANIK, once large, now a medium-sized shipyard – ULJANIK Brodogradnja 1856 d.o.o. in Pula, Croatia, continues the tradition of an established European shipyard with long-standing experience in ship construction and engineering, <https://uljanik.hr/>. The shipyard provided access to real construction environments, structural data, and experienced personnel required for practical validation of the developed XR system.

Within the scope of this study, shipyard engineers participated in the definition of representative inspection scenarios and conducted hands-on testing of the XR application. Their involvement ensured that the evaluation reflected realistic working conditions, including spatial constraints, lighting variability and safety requirements typical for active construction environments.

2.2. About the Developer and the Project

3D BROD d.o.o. is a small, technology-oriented start-up, company based in Pula, Croatia, established in 2023 with the objective of providing next-generation engineering digital services and innovative software solutions for naval architecture and shipbuilding industry in general.

Since its foundation, 3D BROD has actively participated in the development and adoption of solutions based on the Open Class 3D Exchange (OCX) standard, <https://3docx.org/>, supporting interoperability between engineering software systems used in modern shipbuilding environments. In addition to software development, the company continuously provides engineering and digitalisation services to shipbuilding industry clients, including the partner shipyard involved in this validation study.

In early 2025, 3D BROD became an Observer member of the OCX Consortium, further confirming its commitment to the development and practical application of this vessel-specific open data exchange standard in shipbuilding.

3. Operational Considerations in Steelwork Production Environments

The XR application described in this paper has been developed and tested using commercially available and widely accepted head-mounted display (HMD) hardware, specifically the Meta Quest 3 mixed reality headset, <https://treeview.studio/blog/best-mixed-reality-headsets>.

During validation activities, the headset was used primarily in indoor, i.e. sheltered shipyard environments, including production halls and onboard structural areas under construction. Particular attention was given to:

- visibility conditions
- safe operation in confined spaces
- ergonomic usability during inspection tasks, Fig.1



Fig.1: Adjustable position of the HMD on a protective helmet allowing clear sight without removing

One practical consideration identified during field deployment concerns the presence of active welding operations in shipyard environments. Modern mixed reality headsets rely on multiple outward-facing cameras and optical sensors for spatial tracking, hand tracking, and environment reconstruction. Consequently, exposure of these sensors to intense welding arcs may represent a potential operational risk, <https://whatiswelding.com/can-welding-light-damage-camera/>, <https://theweldguru.com/can-welding-light-ruin-a-camera/>. Electric arc welding generates extremely intense visible, ultraviolet (UV), and infrared (IR) radiation. While conventional digital imaging sensors are generally less sensitive to UV radiation than the human eye, prolonged exposure to high-intensity welding arcs has been reported as a potential source of sensor saturation, image artefacts, degradation of optical coatings, and, in extreme cases, permanent sensor damage.

In addition to optical radiation, welding activities introduce secondary hazards including hot metal particles, sparks, airborne dust, and fumes, all of which may negatively affect exposed optical surfaces and tracking performance.

For this reason, the validation activities described in this study were conducted under controlled operational conditions. Direct observation of active welding arcs through the headset cameras was intentionally avoided. Testing was instead performed in completed or temporarily inactive construction areas where structural inspection activities could be evaluated without exposing the equipment to unnecessary optical hazards.

Potential mitigation measures for future deployment may include:

- maintaining a safe stand-off distance from active welding operations,
- avoiding direct line-of-sight exposure of headset cameras to welding arcs,
- implementation of operational procedures restricting XR-assisted inspections during intensive welding activities,
- use of external shielding or physical barriers when inspection tasks must be performed in proximity to welding zones.

Further experimental investigation is required to quantify the long-term effects of accidental exposure to welding radiation on XR headset sensor systems and to establish practical deployment guidelines for mixed reality technologies in shipbuilding production environments.

4. Inspection Procedures Conducted During Validation

Validation activities were structured around representative inspection workflows commonly performed during ship construction. Participants had varying levels of prior experience with digital tools, but no formal training in XR technologies, ensuring realistic user feedback conditions.

Two primary inspection scenarios were selected in cooperation with shipyard personnel.

4.1. Inspection of Completed Structural Block

In the first scenario, inspection was conducted on completed structural blocks within and outside the production hall environment, Fig.2.



Fig.2: 3D model and preparation for inspection of the finalized hull building block (45t steel funnel)

The XR system was used to:

- locate selected structural elements
- verify geometric positioning
- visually compare physical structure with digital model
- identify potential inconsistencies

Participants used the XR headset to navigate around the physical structure while observing the corresponding digital representation aligned within the real environment. The model alignment process is done via the currently implemented 3-step procedure, Fig.3, initiated by the user picking from the application menu, and consists of matching:

1. the first alignment point to its physical counterpart - by simple translation in space,
2. the second point by free rotation (around the 1st point) and
3. finally, the third alignment point by rotation locked around “1st-to-2nd” axis.

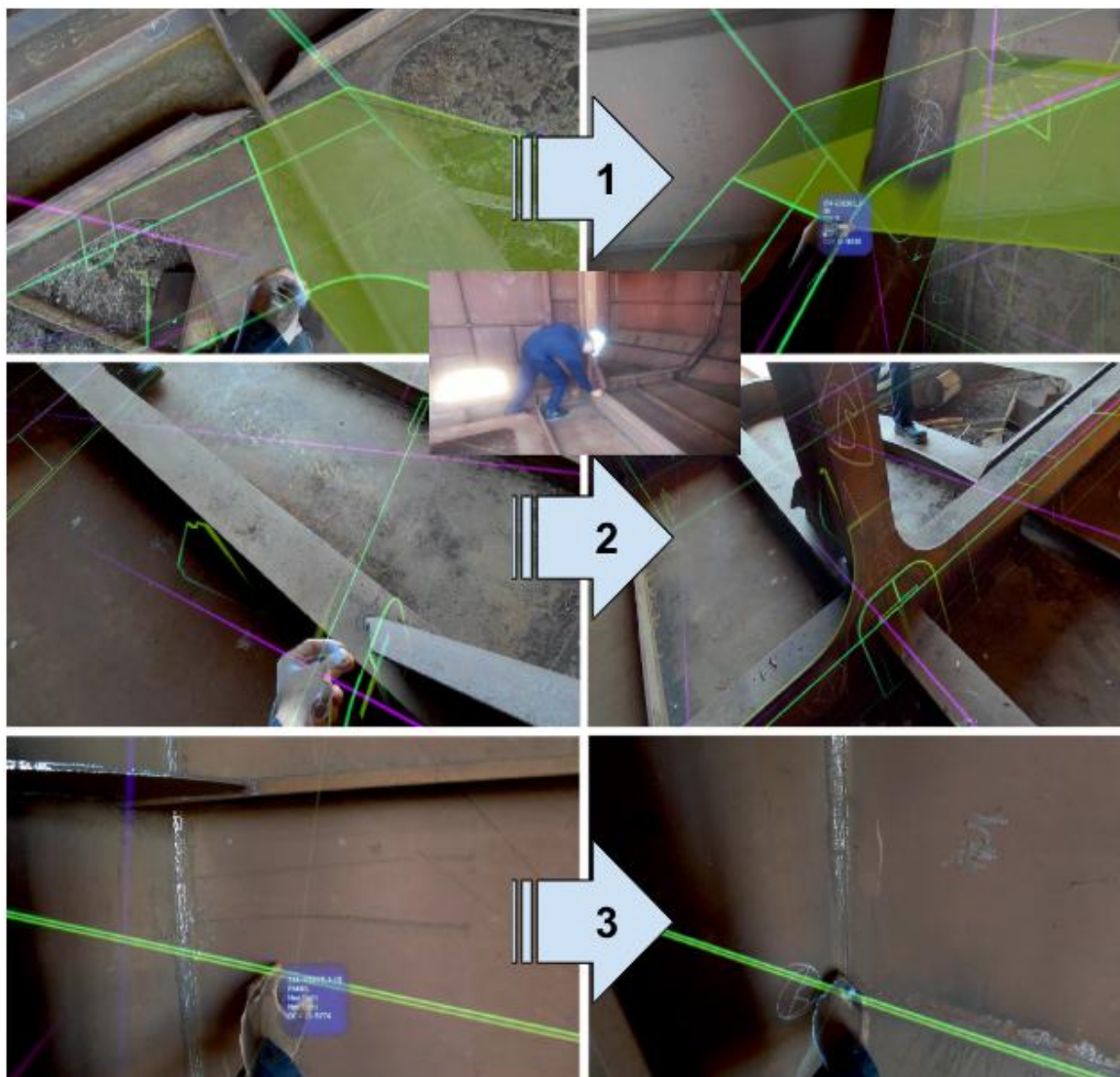


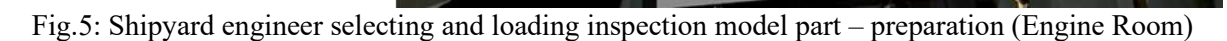
Fig.3: Procedure of 3-step model alignment

There is no need for any prior preparation of the 3D structural models to be loaded at the inspection site, other than to be made available in clean OCX format. The points to be used for the alignment are not predefined, but can rather be any characteristic point of choice by the user to be freely selected (e.g. by pinch/grab gesture) on site.

A photograph of a steel structure under construction, showing a network of steel beams and columns. Overlaid on the image are green and red lines that trace the paths of the steel members, illustrating the digital model's integration with the physical construction. A person's legs are visible in the center, standing on the structure.

This scenario simulated typical quality verification procedures performed prior to structural integration into larger assemblies and/or finally assembled on the slipway.

The second scenario involved inspection activities performed onboard the partially assembled vessel located on the slipway, e.g. deep inside the ship hull as shown in Fig.5 (the aftmost part of the engine room free of hot works has been selected in this case).



Similarly as for the former described use case of inspecting the structural block assembly, also for onboard inspection use cases the above described 3-point alignment procedure need to be run after model load.

Within these scenarios, the XR system was used to support:

- identification of inspection zones
- navigation across structural sections
- verification of structural alignment
- indication of geometrical deviations and misalignments
- interpretation of structural configuration
- dimensional control, Fig.6
- full procedure documenting (by integrated photographing and video recording)

These conditions provided valuable insights into the practical usability of XR technology in real industrial contexts.

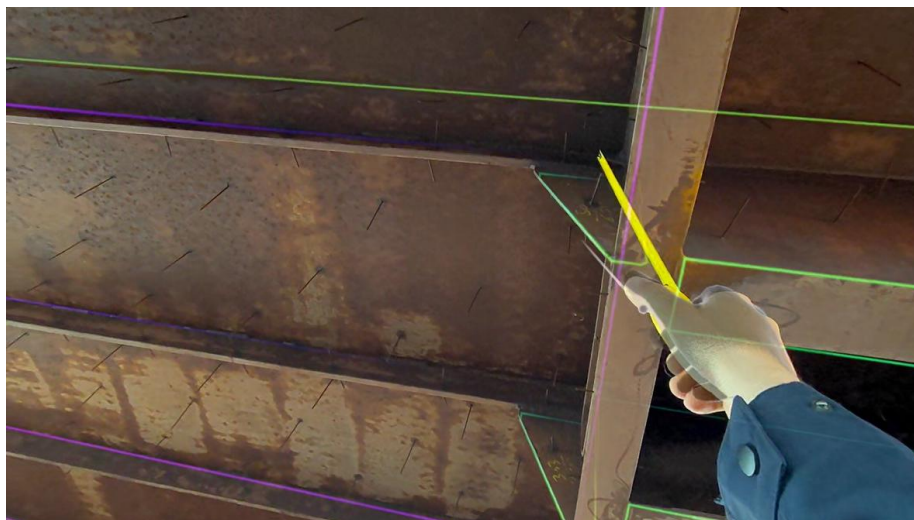


Fig.6: XR assisted direct dimensional control

In both scenarios, indicated was the need to introduce the (additional) initial rough model positioning in space. This rough positioning is planned to be implemented as a initial combined translation + 3° rotation of the loaded model, directly linked to the hand (grab/pinch) movement, in order to allow the fine alignment to be done for each step rather locally (e.g. in a range of hand movement within abt. 1 meter). This way the higher quality final alignment, as shown in Fig.7, is expected to be much simpler to achieve on-site.



Fig.7: Model fully aligned with the built structure (Wheelhouse - ship on a slipway slope)

4.3. Additional validation with actual metadata included

In former described validation examples, specially prepared OCX input model files have been used, having only partially recognized and transferred limited metadata for structure elements (especially plate thickness and materials). The reason was that none of the existing authoring applications capable of producing true and valid OCX format model files was available to authors at the time. Therefore, for the purpose of conducting on-site validation based on actual design and built hull structures, an ad-hoc solution for partial translation of available xml format model input files has been developed (internal XML format to OCX translator).

Subsequently, the additional validation was conducted in a similar manner for the one specific use case and documented, for example as shown in the Fig.8. For that purpose, in the scope of testing an commercially available ship design software capable of OCX format export, it was used to adequately recreate, i.e. remodel one of the structure blocks used for the validation (the funnel block).

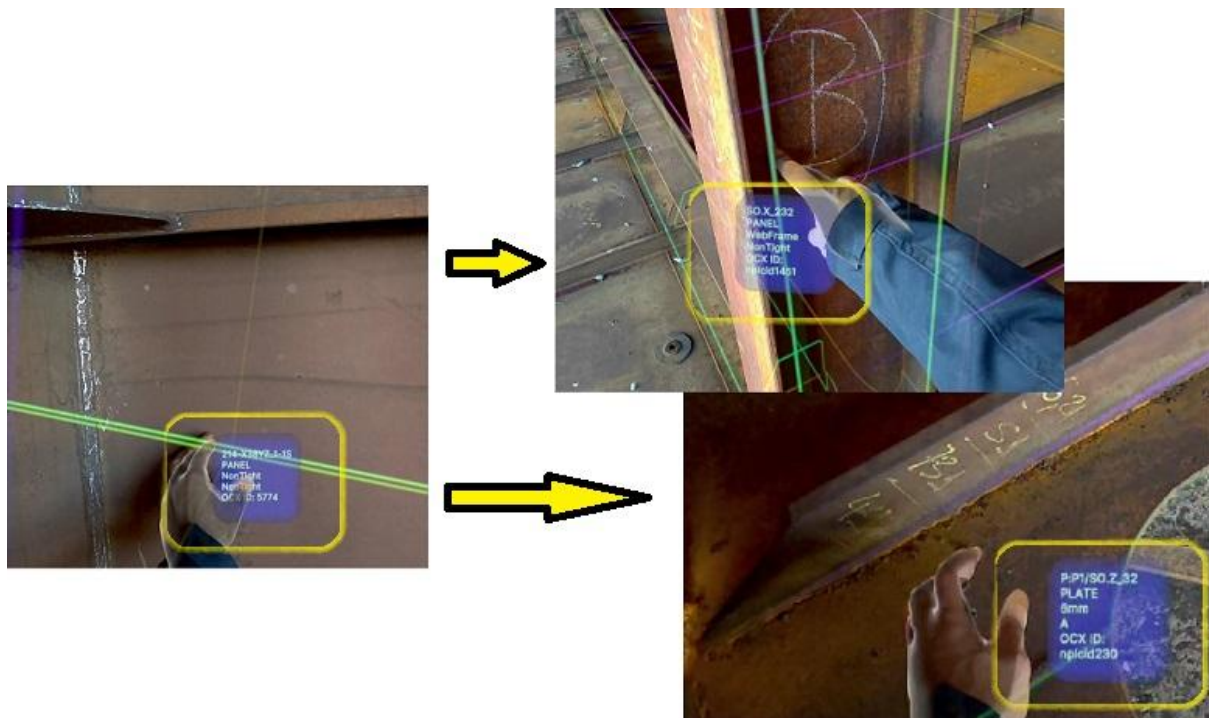


Fig.8: Examples of complete structure properties display out of properly exported OCX model file (images on the right) vs the limited metadata available in the first validation run (left image).

After exporting several proper OCX format model files (with some different output settings, as well as different latest versions of OCX format), additional validation has been performed for one use case, decided to be representative. It was agreed that as such, it will sufficiently close the gap, due to the missing structure properties information, left in the former overall validation scope through all the use cases.

5. Validation Methodology and Field Evaluation Approach

Validation of the developed XR inspection tool was conducted through on-site structured collaboration with practicing shipyard engineers. The evaluation methodology was designed to simulate realistic inspection workflows while allowing systematic observation of user interaction and operational feasibility.

5.1. Selection of Representative Tasks

Four representative task categories were defined in cooperation with shipyard personnel:

1. Basic preparation procedures for the inspection
Establishing safe headset to helmet connection and software start-up procedures
2. Orientation in the newbuilding environment
Selecting and initial alignment of 3D model to real surroundings and user position on-site
3. Structural Element Identification
Locating specific structural members within a constructed block using XR visualization.
4. Verification and Issue Detection
Comparing real construction features with the corresponding digital model representation and presented metadata.

These tasks reflect common activities performed during structural inspections and assembly verification.

5.2. Observed Performance Indicators and Improvement Suggestions

During validation sessions, the following qualitative and semi-quantitative indicators were recorded:

- Time required for preparation after reaching inspection site (hull building location)
- Suitability of use along with basic safety equipment (e.g. helmet, gloves) on site, Fig.9.
- Preciseness of model alignment to real built hull structures
- Accuracy of structural identification
- User-reported ease of navigation
- Frequency of referencing traditional documentation
- Observed workflow interruptions
- User feedback regarding situational awareness

These indicators were selected to provide insight into operational feasibility rather than formal laboratory (i.e. design office) benchmarking.

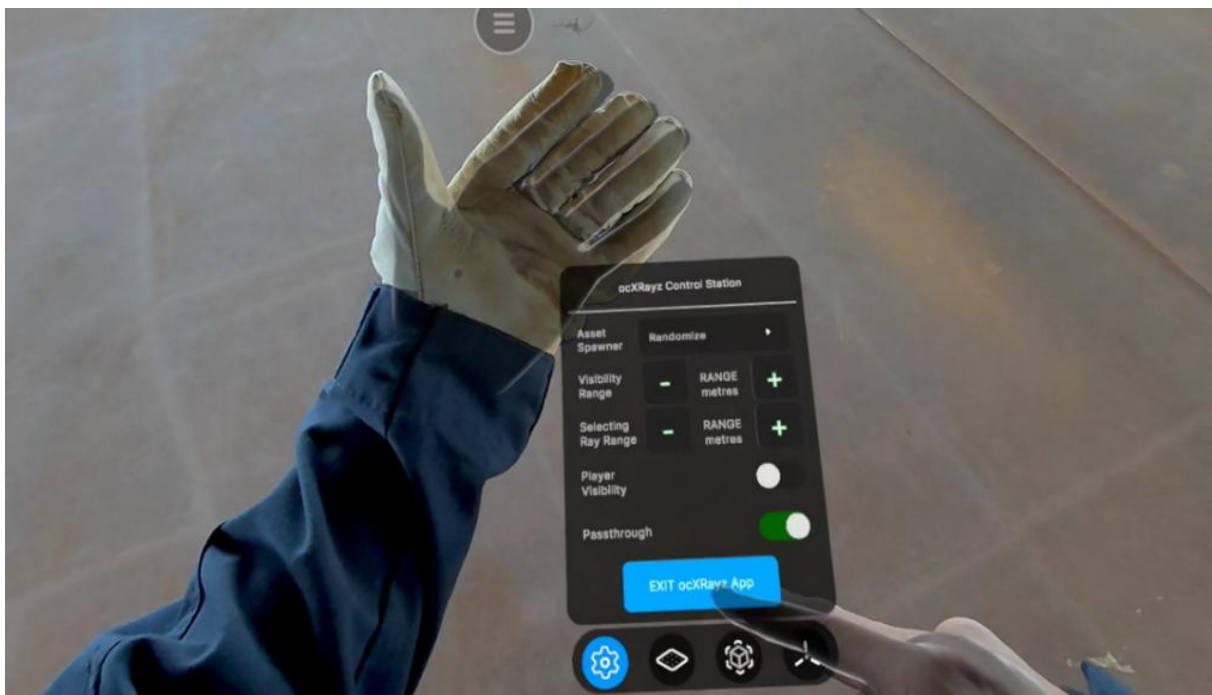


Fig.9: Working with gloves enabled, with precision depending on design fineness

Current use cases consisted of medium to relatively large assembly blocks of dimensions up to ~ 20 m in width/length and up to ~10 m in height. The obtained results indicate that considerably larger structures could likely be inspected without significant changes to the model preparation workflow. Based on the current experience, inspection of structures at least twice this size appears technically feasible; however, systematic validation of larger-scale use cases remains part of the planned future work.

Regarding the model alignment, precision under 10-20 mm was generally achieved after two or three repeated alignment runs. This level of precision is considered sufficient for the intended application, whose primary purpose is to support verification of the correct placement and identification of structural components rather than high-precision dimensional metrology. Detailed dimensional verification and tolerance measurements remain the responsibility of dedicated surveying and measurement equipment routinely employed during ship construction.

As a final result of the collaboration on validation, significant valuable suggestions were pointed out by the main shipyard evaluator:

- Assure clear distinction to active rotation point of the model (while aligning)
- Better user control and indication of model alignment steps (translation / rot.)
- Increase the visibility (by increasing thickness or lowering transparency level) of the model structure graphics for bright light environments, Fig.10
- Structural metadata windows need (various) further improvements in data readability
- Currently available display of the actual structural frame position to be better centered in the user's field of vision, with optional possibility also to display all current coordinates in a ship's reference system (X, Y, Z)
- Further increase of the text (font, 10-20%) size within the model selection menu
- Other potential small software UI/UX improvements suggestions.

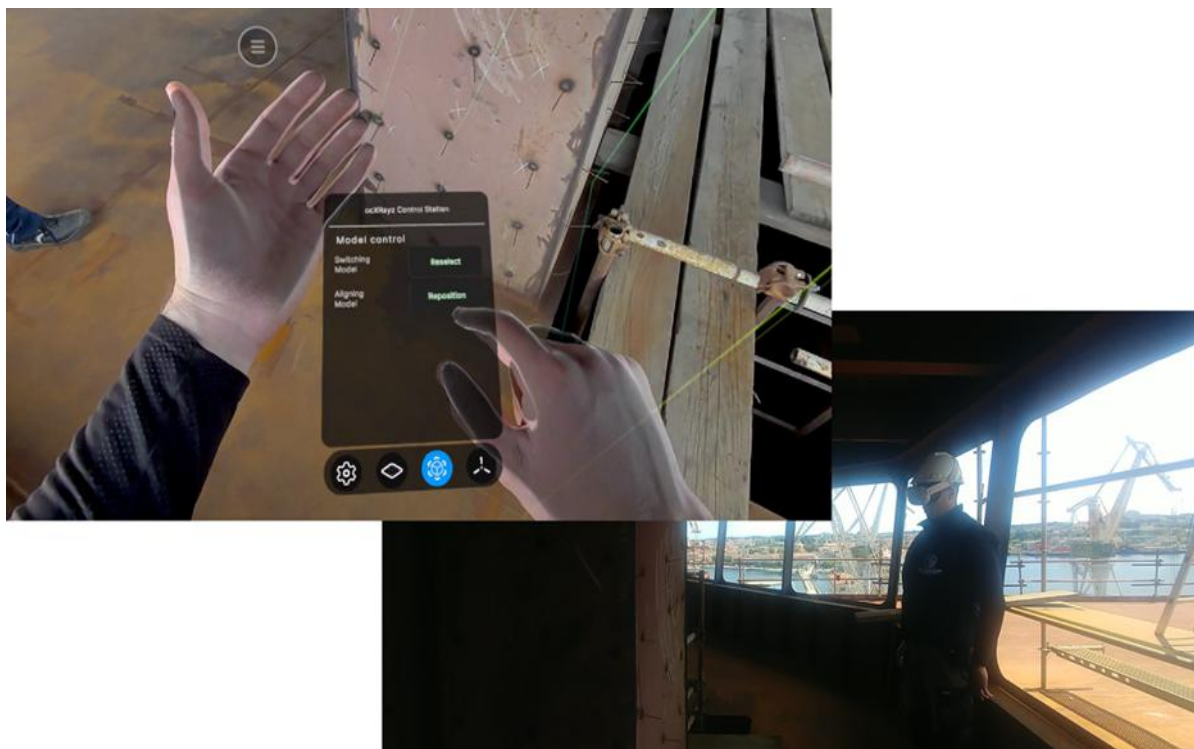


Fig.10: Lower visibility of transparent graphics in a sheltered, but bright light environment

6. Overall Observations from the Field Evaluation

6.1. Benefits Observed During Field Evaluation

The conducted validation activities demonstrated that XR-based inspection workflows have the potential to significantly enhance situational awareness during structural verification tasks.

Traditional inspection workflows rely heavily on interpretation of two-dimensional drawings and manual spatial reasoning. This approach introduces cognitive load, particularly in complex structural environments where multiple components overlap visually. The XR-based approach enables direct visualization of structural information within the physical workspace. This reduces the need for mental transformation between drawing representations and real structures.

Additional flexibility of use comes from the possibility for users to choose to operate with hand controllers or rather controlling the program with hand gestures (incl. gloves), leaving hands free of equipment or various assets in general, rendering the inspection process safer while moving through the hazardous environments. Using the dedicated controllers offer somewhat better handling precision, which is important mostly for the initial model alignment procedure only.

Several operational benefits were observed during real-world testing:

- Improved spatial understanding of structural arrangements
- Faster identification of inspection-relevant components
- Reduced dependency on printed drawings (with potential of complete elimination)
- Increased confidence during structural verification
- Improved communication between team members during inspection tasks

6.1. Overall Challenges and Limitations Identified

Despite the promising results, several challenges were observed:

- Limited visibility under strong outdoor lighting conditions (needs software improvement)
- Need for optimized model simplification strategies
- User fatigue during extended headset usage
- Requirement for improved alignment calibration procedures
- Operation limited in active welding environments
- Dependency on availability of compatible digital model data

These findings provide direction for further development and optimization efforts. Some of the following steps in the application development will be to implement indication of the 3D model loading progress, much clearer currently used actions and possibility for user to activate interactive instructions during the work.

7. Conclusion

This paper presented the development and field validation of an XR-based inspection support tool designed for use in shipbuilding environments. The system integrates structural engineering data based on (yet not strictly limited to) the OCX standard into an interactive XR visualization environment.

Validation conducted in collaboration with practicing shipyard engineers demonstrated the feasibility of using XR technologies to support inspection workflows under realistic operational conditions.

The achieved maturity level corresponds to Technology Readiness Level TRL-6, indicating successful demonstration of the system in a relevant industrial environment.

Observed benefits include improved spatial understanding, faster identification of structural elements, full process documenting and reduced reliance on traditional documentation. At the same time, identified limitations provide clear guidance for further development and optimization.

Acknowledgements

This article is a partial overview of the original article "Development of OCX-Based XR Inspection Tool and On-Site Validation in a Shipyard Environment", published under proceedings of SORTA 2026 - 27th International Symposium on Ship design, Marine Engineering and Technology.

The work was carried out within the scope of the ocXRayz™ development project by 3D BROD d.o.o.: “Razvoj inovativne XR aplikacije za terenske inspekcije kao karike brodograđevne industrije 5. generacije” (engl. Development of an Innovative XR Application for On-Site Inspections as an Enabling Component of Shipbuilding Industry 5.0) which is co-funded by the European Union, within the “Competitiveness and Cohesion 2021–2027” Programme from the European Regional Development Fund (ERDF), supporting innovation and digital transformation of industrial processes.

Automatic Adaptation of Ship Subdivision: An Exploration on Methanol Conversion

Diego De Leon, NHL Stenden, Leeuwarden/Netherlands, diego.de.leon.wug@nhlstenden.com

Herbert Koelman, NHL Stenden, Leeuwarden/Netherlands, herbert.koelman@nhlstenden.com

Sam Otjens, NHL Stenden, Leeuwarden/Netherlands, sam.otjens@student.nhlstenden.com

Abstract

The lack of historical data for alternative fuel designs provides a challenge for data-driven aided procedures and methodologies in early ship design. Aiming to offer a complimentary technology to aid Data-Driven Design, a set of hard-coded scripts designed to automate the transformation of the volume distribution in ships in Binary Space Partitioning (BSP) format for alternative fuel designs, starting with a pilot implementation for methanol. These scripts enable deterministic and reliable adaptations of existing or generated layouts, ensuring compatibility with emerging propulsion systems while maintaining engineering rigor and existing design knowledge. Using the standard BSP architecture allows to integrate these transformation tools with Data-Driven, Generative Algorithm based proposals for layout alternative generation for early ship design. This integration enables the development of alternative solutions by leveraging the certainty of a deterministic approach to combine it with the fast iteration potential of generative algorithms. As the progress in ship design methodology points towards a future where Data-Driven and deterministic technologies interact to compensate their limits, this toolset serves as a first exploration in the integration of both technologies. The paper discusses the implementation, workflow, and implications of this integrated system for the future of early-stage ship design.

1. Introduction

The maritime industry is currently undergoing a fundamental transformation propelled by environmental and legislative pressures, including the International Maritime Organization's (IMO) decarbonization targets, EU FuelEU Maritime, and AFIR regulations. Transitioning to alternative energy carriers such as methanol, ammonia, hydrogen, and battery-electric systems presents severe technical and spatial challenges due to their significantly lower volumetric energy densities compared to conventional fuel oil. Accommodating these larger storage volumes creates a direct competition with revenue-generating cargo space, establishing a critical design compromise particularly in volume-limited vessels like container ships.

Evaluating these spatial trade-offs occurs primarily during the initial stages of ship design, yet current general arrangement workflows remain largely manual, experience-driven, and digitally isolated compared to modern hydrodynamic or structural workflows. Attempting to solve this through purely data-driven approaches reveals a fundamental limitation: the scarcity of historical design data for novel alternative-fuel configurations. Without sufficient historical reference points, standalone generative models struggle to guarantee physical validity or regulatory compliance in unmapped design spaces.

This bottleneck leads to the question of how to combine modern computational automation with established naval architecture logic to accelerate early-stage decision-making. Developed within the EU Horizon SEUS framework, the proposed solution is a hybrid methodology that bridges deterministic, rule-based transformations with data-driven generative exploration. By representing internal volumes using Binary Space Partitioning (BSP) trees, procedural transformation scripts deterministically adapt layout distributions, such as methanol fuel conversions, while preserving naval engineering rigor. Interfacing these standardized BSP representations with generative algorithms enables the fast ideation of multiple physically sound layout proposals, providing naval architects and engineers with a reliable foundation to evaluate options and accelerate the initial design phase.

2. Ship layout rationale and alternative fuel constraints

Naval architects are forced to navigate complex trade-offs during early-stage general arrangement design. Decoupling the energy source from a fixed constant in heavy fuel oil (HFO) towards a multiple solution paradigm in alternative fuels requires balancing bunkering frequency against lost revenue-generating cargo volume

2.1. Historical energy transitions: solution-driven vs. problem-driven

The maritime industry has seen a series of technological shifts in its long history as new energy sources became available and offered more practical means of ship propulsion. Multiple transitions provide historical case studies showing how the maritime engineering landscape adaptively absorbed every change, *Herdzik (2023)*. In historical precedents, every shift was initiated by the availability of a new technology that superseded the previously dominant solution due to operational, economic, or practical superiorities, creating a solution-driven transition, *DNV (2019)*.

The current decarbonization challenge fundamentally differs from these historical precedents. The drive to replace dependency on conventional fuel oil stems not from technical limitations of the fuel itself, but from the severe external environmental impacts of greenhouse gas emissions. This problem-driven transition introduces unprecedented uncertainty: rather than adopting a single superior replacement, the energy source becomes an application-dependent variable that must be continuously evaluated during early ship design.

2.2. Methanol volumetric energy penalty

When evaluating alternative fuel candidates, methanol (CH_3OH) emerges as a promising low-carbon energy carrier due to its liquid state at ambient temperatures and established bunkering infrastructure. However, its lower energy density introduces severe spatial constraints on ship layout.

Methanol possesses a volumetric energy density of ~ 15.6 MJ/L, roughly half that of conventional Marine Gas Oil (MGO) at ~ 36.6 MJ/L. To maintain an equivalent operational range, fuel storage tanks must expand by a factor of 2.2 to 2.5 compared to baseline marine diesel arrangements. In volume-critical vessel types, such as container ships or Ro-Pax ferries, expanding fuel volumes directly reduces space available for cargo or passenger arrangements, turning fuel selection into a primary driver of vessel geometry.

2.3. Regulatory frameworks and mandatory safety constraints

The regulatory requirements governing methanol fuel integration stem primarily from classification society rules and international safety guidelines *BV (2022)*, *IMO (2016)*. To mitigate fire and toxicity risks, methanol fuel tanks are prohibited inside accommodation spaces and Category A machinery spaces, spaces containing internal combustion machinery for main propulsion, over 2675 kW or oil-fired equipment. Where methanol storage is located adjacent to Category A machinery or cargo spaces, boundaries must be protected by a cofferdam of at least 600 mm width in combination with A-60 fire insulation, *BV (2022)*. All fuel tanks must be situated between the collision bulkhead and the aft peak bulkhead. Integral tanks generally require protective cofferdams around their perimeter, except along boundaries formed by outer shell plating below the light waterline, adjacent methanol tanks, or fuel preparation spaces, *IMO (2016)*.

To maintain single-fuel operational reliability, fuel storage is required to be split between at least two independent tanks, each capable of providing sufficient capacity for autonomous operation, *BV (2022)*. Dedicated fuel preparation spaces must remain gas and liquid tight relative to adjacent enclosed areas, feature independent ventilation systems, and be directly accessible from the open deck or through an airlock, *IMO (2016)*. Service tanks fall under the same regulatory regime as main storage tanks and are subject to the same cofferdam and boundary restrictions.

2.4. Operational trade-offs and design decisions

Apart from the changes in volume, placing expanded fuel tanks alters lightship weight distributions, vertical center of gravity (VCG), and longitudinal trim characteristics. Evaluating these interconnected spatial and stability constraints requires adaptable layout methodologies that can rapidly test alternative subdivision topologies while guaranteeing regulatory compliance.

Because methanol's volumetric energy density is roughly 2.5 times lower than Heavy Fuel Oil (HFO), a complete 1:1 endurance replacement significantly reduces internal cargo capacity. To mitigate this penalty, a volumetric compromise such as a 75% energy equivalence factor is applied during spatial allocation. Daily service tanks are sized using the lower calorific value and density ratio with an added reserve margin, balancing necessary fuel autonomy against cargo volume preservation

3. Technological framework: BSP as ship definition

Binary Space Partitioning (BSP) serves as a neutral geometric framework that represents internal vessel compartmentation through a hierarchical, volumetric binary tree structure, a concept that is in use in industry specific applications such as SARC PIAS own subdivision tool, *De Koning et al. (2011)*. By coupling spatial boundaries and deck arrangements with functional metadata inside a standardized JSON schema, BSP bridges domain-specific naval architecture tools with automated parametric workflows. This unified digital definition provides the deterministic foundation required to programmatically evaluate and adapt internal ship layouts during early-stage vessel design.

The framework achieves geometric neutrality by aligning partition planes with primary vessel axes, including transverse bulkheads along the x-axis, longitudinal bulkheads along the y-axis, and horizontal decks along the z-axis. Through this spatial structuring, domain-specific exports such as SARC PIAS XML files are translated into a standardised JSON-format BSP trees where every subdivided sub-volume retains its precise spatial boundaries, calculated volume, and functional designation.

The BSP tree represents internal compartmentation as a hierarchical binary data structure where the root node encapsulates the complete initial hull volume. Internal nodes define individual slicing planes, while leaf nodes represent fully bounded, mutually exclusive child compartments. This recursive decomposition guarantees complete volumetric coverage across the hull without spatial gaps, overlapping geometry, or topological inconsistencies. Each node in the tree maintains rich metadata essential for downstream automation

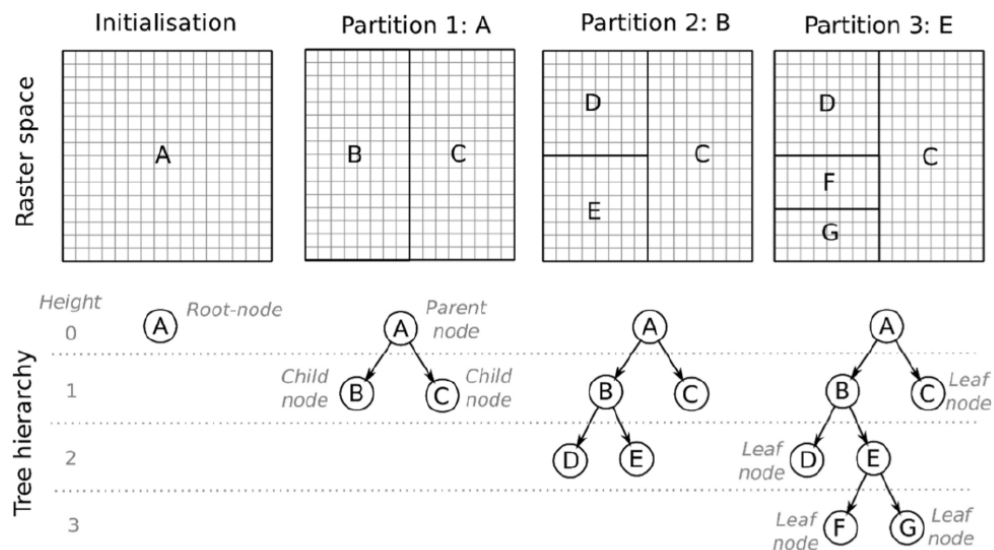


Fig.1: Binary Space Partitioning and corresponding tree hierarchy, *Etherington et al. (2022)*

3.1. Deterministic Layout Adaptation Operations

Layout modifications proceed through a deterministic sequence of rule-based spatial operations executed directly on the BSP tree structure. Automated modules first scan the tree to identify existing spatial markers, including engine room bulkheads, double-bottom heights, and fuel zones. Transverse partition planes are then shifted by exact structural increments, such as adjusting an aft bulkhead by single container bay lengths. Sub-planes are inserted at calculated offsets to introduce specialized spaces like fuel preparation rooms, while protective cofferdam spaces are generated at statutory distances required by classification standards. Subsequent sub-tree reclassifications update compartment use categories seamlessly while preserving top-level topological integrity.

3.2. Data-driven design integration

Continuous STEP geometry and raw Boundary Representations present severe structural challenges for data-driven generative models due to their non-standardized parameters and high-dimensional complexity. The BSP architecture resolves this bottleneck by translating continuous 3D hull volumes into structured, hierarchical tree representations. By encoding spatial splits and sub-volume relationships into parent-child nodes, BSP converts arbitrary physical domains into standardized topological graphs. This structured format enables machine learning algorithms to directly ingest, learn, and generate complex compartment layouts without losing critical spatial logic. The inclusion of ship design specific metadata in the BSP tree permits the learning of certain ship design logic to the algorithm in a way that does not depend on synthesis, but engineering data.

In traditional optimization workflows, evaluating layout modifications requires computationally heavy CAD kernel Boolean operations that choke data-driven training loops. The BSP architecture replaces these expensive CAD operations with rapid tree-traversal algorithms, enabling machine learning models to iterate through multiple candidate subdivisions. Crucially, the planar splitting mechanism guarantees topological validity and watertight compartment boundaries by construction. By eliminating the non-manifold geometric errors common in raw 3D generation, the BSP framework ensures that all data-driven outputs are usable for design tasks.

4. Hybrid implementation and integrated workflow

Transitioning to maritime methanol requires addressing distinct spatial and regulatory constraint, *Otjens (2026)*. Because methanol's energy density is approximately 2.5 times lower than HFO (19.9 MJ/kg vs. 40.2 MJ/kg), integrating methanol is a complex spatial arrangement problem rather than a simple volumetric swap

Under BV NR670 regulations, fuel tanks containing methanol are strictly prohibited inside accommodation spaces or Category A machinery spaces. Integral fuel tanks must be isolated using protective cofferdams measuring between 600 mm and 1000 mm, equipped with A-60 fire insulation whenever they border Category A machinery spaces or high fire risk cargo areas. Cofferdams are not required on tank boundaries adjacent to shell plating below the lowest waterline, neighbouring methanol tanks, or fuel preparation spaces. Single-fuel configurations require storage to be distributed across at least two separate tanks capable of sustaining 8 hours of autonomous operation at Maximum Continuous Rating, with service tanks legally classified under the same rules as fuel tanks. Furthermore, the Fuel Preparation Space must remain gas- and liquid-tight, situated outside Category A machinery spaces, and provide direct access from an open deck or through an airlock.

4.1. Dataset and container ship clustering

To evaluate spatial impacts across commercial fleets 83 container vessels from the Significant Ships publication series, *RINA (2006–2018)* were analysed. Dimensional filtering based on overall length, breadth, and TEU capacity categorized the dataset into three representative vessel clusters for parametric modelling. Cluster 1 comprises ultra-large container ships carrying 17,000 to 22,000 TEU

with an LOA of approximately 400 meters and a breadth of 59 meters. Cluster 2 represents Neo-Panamax vessels with an LOA of roughly 366 meters and a 48.2-meter breadth, while Cluster 3 covers Neo-Panamax designs of identical length but with a wider 51.0-meter beam.

4.2. Methanol conversion impact and spatial logic

To adapt conventional PIAS reference models for these clusters, the study expanded the designated fuel zone longitudinally by one 14.59-meter container bay and positioned the Fuel Preparation Space and service tanks directly forward of the main engine room bulkhead. This structural rearrangement caused a loss in cargo capacity ranging from 278 to 334 TEU per cluster, representing a 4.1% to 4.7% reduction in total hold capacity, *Otjens (2026)*. Required storage volume was determined using an energy-equivalence formula incorporating a 75% factor to balance operational range against payload loss.

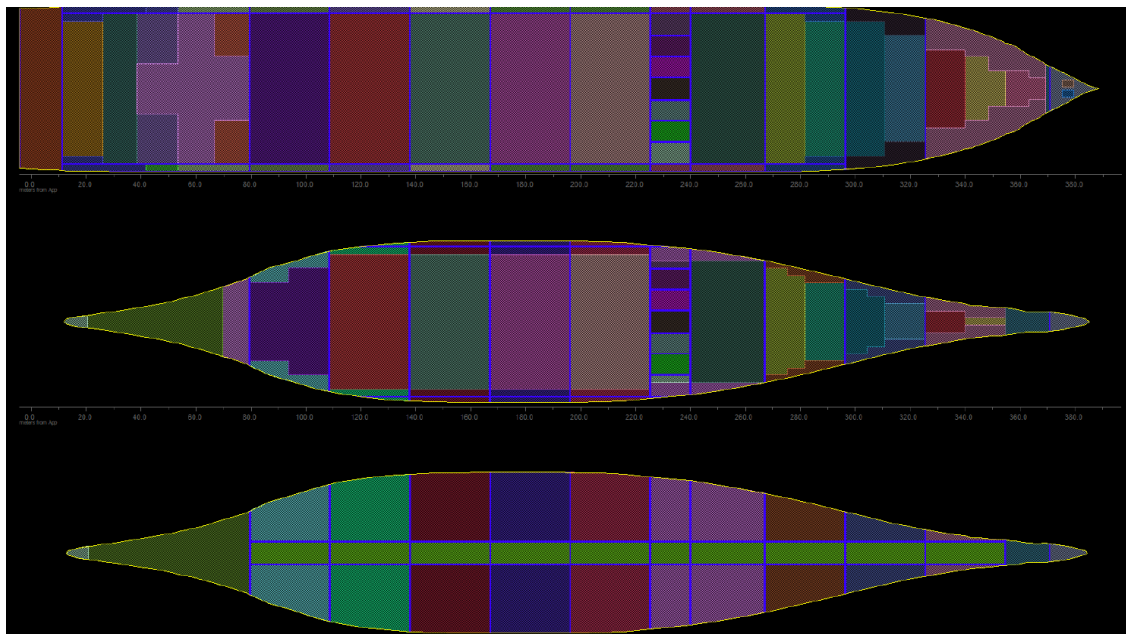


Fig.2: PIAS model of a 400 m generalised fuel oil container ship from the database, *Otjens (2026)*

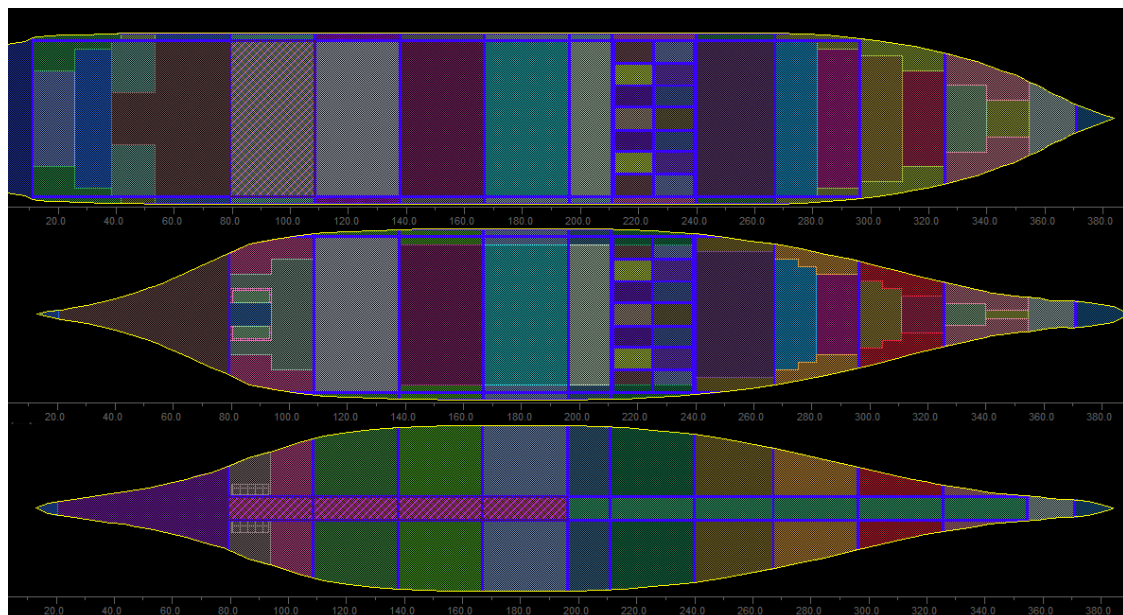


Fig.3: PIAS model of a 400 m generalised container ship from the database after adaptation to methanol, *Otjens (2026)*

4.3. Parametric tool and evaluation

The parametric methanol arrangement tool is a Python-based software application designed to automate the conversion of a ship's conventional internal subdivision into a regulatory-compliant methanol fuel layout at the concept design stage. Operating on a BSP tree hierarchy stored in JSON format, derived from PIAS XML export files formatted under the SEUS framework, the tool represents internal ship geometry as a recursive tree of orthogonal partition planes corresponding to transverse bulkheads, longitudinal bulkheads, and decks. Each leaf node in this tree structure defines a bounded, non-overlapping compartment volume associated with functional properties and weight groups.

To perform spatial reasoning, the tool requires specific semantic input data, relying on predefined PIAS weight groups such as Fuel_oil, Ballast, Cargo, and Machinery, alongside key reference frames like ER bulkhead. The program operates through two parallel algorithmic pipelines: the storage tank pipeline and the service tank pipeline. Both pipelines execute a three-stage modular sequence consisting of detection, rule-based operation planning, and execution. The detection module parses the JSON tree to map boundary coordinates, identifying the fuel zone, double-bottom geometry, engine room bulkhead, and adjacent spaces. The rules module evaluates these geometric boundaries against Bureau Veritas NR670 safety constraints to generate an ordered sequence of spatial tree operations.

The storage tank pipeline executes geometric transformations to accommodate methanol's lower volumetric energy density. It expands the existing fuel zone longitudinally by relocating the aft watertight bulkhead by an amount of container bay lengths (14.59 m), while inserting a non-watertight internal bulkhead at the original position to preserve local compartment boundaries. Horizontal partition planes are inserted at container height increments to establish upper spaces, protective cofferdams with thickness ranging from 600 mm to 1000 mm are generated at boundaries shared with high fire risk cargo holds and surrounding water ballast tanks are reclassified to methanol-segregated ballast capacity. These decisions are hard coded into the tool ensuring hard compliance with requirements for the use of methanol and the volumetric differences with fuel oil.

The service tank pipeline automates the placement of fuel treatment infrastructure directly forward of the Category A engine room. It detects the double-bottom geometry at the engine room bulkhead and constructs a service tank extending above the tank top deck. Directly above this tank, the tool inserts a gas- and liquid-tight Fuel Preparation Room. Mandatory regulatory separations are enforced by creating a 1.0 m cofferdam with A-60 fire insulation along the engine room bulkhead and 0.6 m cofferdams along lateral ballast and cargo boundaries.

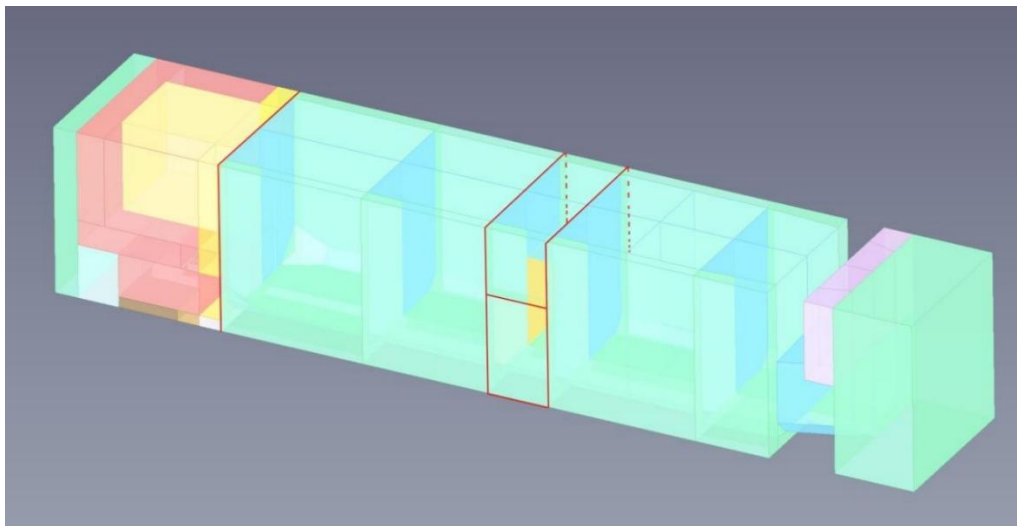


Fig.4: OpenCASCADE 3D visualisation of a base model prior to the Methanol conversion modifications, where the red lines indicate the engine room bulkhead (aft) and fuel oil zone boundaries (mid), *Otjens (2026)*

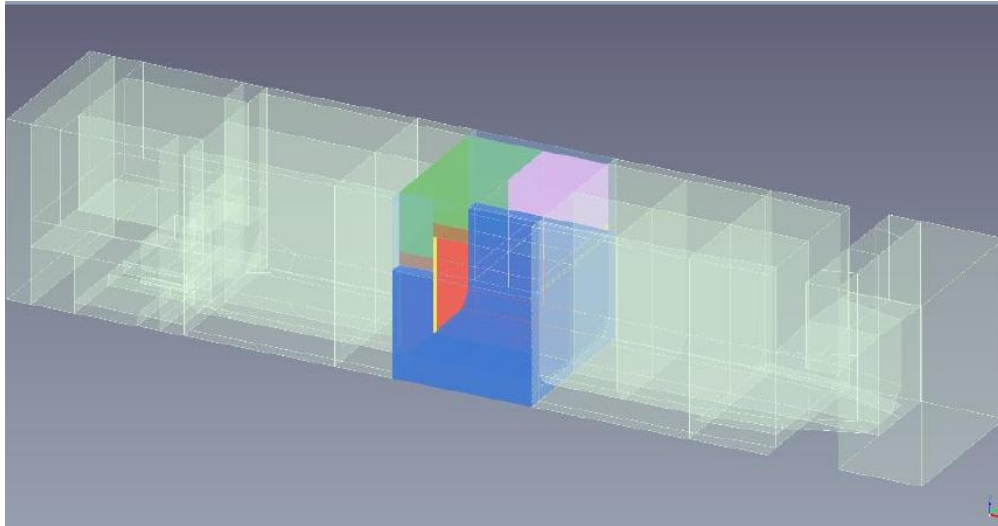


Fig.5: OpenCASCADE 3D visualisation of a base model after scripted Methanol conversion modifications, where blue is MeOH water ballast, red is Methanol tank space and yellow is cofferdam space, *Otjens (2026)*

Upon completing all spatial operations, the execution module updates the BSP tree node properties, names, and spatial coordinates, generating an updated BSP-tree JSON file. This output file can be parsed by pythonOCC to produce a color-coded 3D CAD visualization for geometric verification before being re-imported back into PIAS for downstream naval architectural analyses, such as stability and tank capacity calculations.

5. Ship design methodology impact

The integration of alternative fuels such as methanol represents a fundamental shift in early-stage ship design methodology rather than a simple volumetric substitution. Traditionally, concept design relies heavily on empirical iterative methods like design spirals, designer intuition, and manual geometric drafting within Computer-Aided Design (CAD) software and naval architecture packages such as PIAS. When working with conventional heavy fuel oil, historical heuristics offer clear guidance on space allocation, tank arrangements, and machinery boundaries. However, alternative fuels break these established rules of thumb. Methanol requires approximately 2.5 times the storage volume of conventional fuel to maintain an equivalent operational range due to its lower volumetric energy density.

This operational shift demands a new generation of computational tools that move beyond static drafting toward programmatic, rule-based space management. Rather than manually recalibrating internal subdivisions for every alternative fuel scenario, modern design methodologies must encapsulate regulatory and geometric logic directly into parametric pipelines. In this paradigm, the naval architect transitions from an iterative CAD draftsman to an architect of algorithmic rules. The tool does not replace human decision-making; instead, it codifies design conditions into reproducible workflows. This change allows designers to rapidly evaluate the spatial impact of energy source changes during early concept phases, ensuring that safety constraints and volumetric trade-offs are identified before structural arrangements are locked in.

5.1. Data as potential and bottleneck

A robust modern design framework depends on a continuous feedback loop between historical design data, deterministic spatial tools, and emerging data-driven algorithms. The initial phase of this methodology relies on extracting structural patterns from historical fleet data. By systematically analysing datasets such as the 83 container vessels from the Significant Ships series, recurring geo-

metric relationships emerge, including standard bay lengths, transverse bulkhead spacing, double bottom heights, and pipe duct dimensions. Clustering vessels based on principal dimensions rather than nominal TEU capacity reveals consistent internal layout logic. Mining historical General Arrangements effectively translates implicit industry experience into explicit, quantified design rules that serve as the baseline input for parametric modelling.

The secondary phase of data evaluation expands on using deterministic tools as data generators for future data-driven applications. While historical datasets reflect conventional fuel paradigms, there is an inherent lack of standardized historical data for alternative-fuel vessel layouts. Deterministic tools bridge this gap by executing rule-based spatial transformations, such as bulkhead modifications for fuel tanks and cofferdam generation on baseline designs, systematically generating valid alternative fuel configurations. The output files from these deterministic operations, structured as hierarchical spatial representations, serve as a pragmatic source of training data for machine learning models. Data-driven tools require vast amounts of labelled, syntactically uniform data to learn complex spatial patterns. By using deterministic tools to transform historical designs into standardized synthetic datasets, the methodology creates a self-reinforcing pipeline where past vessel data informs deterministic conversion rules, and deterministic outputs provide the foundation for future data-driven generative design tools.

5.2. Methanol adaptation tool impact

Evaluating the validity of this hybrid approach highlights its transformative impact on ship design, which unfolds across two distinct operational dimensions: its direct utility as a design automation tool and its enabling role within a broader data-driven ecosystem.

As an operational tool in its own right, the programmatic spatial pipeline fundamentally impacts early-stage naval architecture by eliminating manual design friction and accelerating iteration speed. By translating regulatory rules and technical limitations into executable algorithmic rules, the tool automates spatial tasks that traditionally required days of manual drafting across separate CAD and naval architecture software. Instead of manually shifting bulkheads, recalculating tank volumes, and adjusting container grids for every alternative fuel scenario, naval architects can execute deterministic spatial transformations immediately. This eliminates unnecessary human labour and allows designers to rapidly evaluate multi-variable trade-offs, such as exact TEU capacity loss versus operational range, while ensuring strict, baseline compliance with regulatory safety codes.

Beyond its immediate operational efficiency, the tool serves as a critical secondary function as a data engine and execution backend for future data-driven design frameworks. First, it helps solving the fundamental data scarcity problem facing alternative fuel vessel design. Because historical data for alternative fuel layouts is virtually non-existent compared to conventional fuel oil arrangements, the tool acts as a synthetic data generator, producing vast, standardised, and topologically validated spatial datasets needed to train deep generative models and reinforcement learning agents. Second, it operates as a deterministic execution service that higher-level data-driven tools can directly invoke.

In future development, the intention is that when a data agent suggests macroscopic space allocations or complex layout modifications, it can call the programmatic BSP pipeline to handle the fine-grained geometric cuts, topological boundary enforcement, and hard regulatory verification avoiding hallucination risks. This hybrid coupling would allow data-driven tools to explore non-linear design spaces freely, confident that the deterministic backend will consistently ground their predictions in real-world geometric feasibility.

5.3. Digital ship representation and impact on data driven and deterministic approaches

Developments on the D^3 :SEA early design platform, *De Leon et al. (2025)* focus on refining the architecture moving towards a Variational Autoencoder-Generative Adversarial Network (VAE-GAN) system to better handle complex volumetric arrangements in the Generation part of the GAN. By employing a Binary Space Partitioning (BSP) tree structure as the primary spatial representation, the

VAE encoder directly processes the hierarchical, boundary-defined geometry of ship subdivisions and their metadata. Rather than relying on textual descriptions or natural language proxies, which limit Large Language Models (LLMs) due to their lack of spatial and geometric grounding, the BSP representation encodes multi-dimensional geometric relationships natively. This structured geometric input enables the latent space of the VAE generator to capture spatial and volumetric logic, and functional compartment boundaries directly, operating on exact spatial geometry rather than indirect linguistic approximations.

Despite the exploratory power of generative models, purely data-driven approaches present fundamental limitations in engineering design. Data-driven algorithms naturally produce continuous, probabilistic outputs that lack the exact precision and absolute repeatability required for formal naval architecture. Engineering workflows demand deterministic solutions that can be rigorously replicated and audited. For instance, regulatory compliance checks such as classification society require absolute mathematical verification. Similarly, practical design constraints like locking in standardized container bay dimensions (TEUs) require exact discrete snapping; while data-driven models suggest continuous spatial envelopes, deterministic tools adapt better to execute discrete, rule-based modifications and establish precise structural grids.

To reconcile continuous exploration with rigorous engineering precision, a hybrid framework is proposed that places a data-driven generative core works alongside deterministic tools. In this workflow, the data-driven VAE-GAN operates as the initial ideation engine, rapidly generating diverse candidate spatial subdivisions across a broad design space. Once a layout proposal is selected, the system allows for calls to deterministic engines to execute specific spatial adjustments, enforce discrete structural boundaries, and run automated statutory verifications.

Furthermore, these deterministic engines support the framework by generating verified synthetic datasets during offline preprocessing, providing structured training data to continuously refine the generative core. This hybrid synthesis preserves the creative flexibility of data-driven design while upholding the mathematical rigor, repeatability, and compliance required for real-world naval architecture.

6. Conclusions and the future of maritime software

The integration of hybrid computational workflows marks a fundamental shift in early-stage ship design methodology. Traditionally, naval architecture relies on sequential, iterative design iterations where early spatial synthesis is decoupled from rigorous engineering validation. Transitioning toward a hybrid paradigm transforms this workflow from reactive manual iteration to continuous, automated design space exploration. By pairing data-driven generative models with deterministic evaluation loops, the design process evolves from simply drawing geometry to actively navigating high-dimensional solution spaces. The principal takeaway of this methodology is that machine learning does not replace classical naval architecture principles; rather, it amplifies them, allowing exploratory spatial generation to operate strictly within established engineering boundaries.

This methodological evolution redefines how deterministic logic functions within maritime software ecosystems. Rather than using rule-based algorithms and mathematical filters purely as downstream auditing tools, the hybrid workflow embeds them directly within the generative loop. Deterministic algorithms act as dynamic constraint enforcement layers, converting probabilistic spatial outputs into discrete, rule-compliant geometries. Additionally, this standardisation of digital ship representation enables synthetic data generation, leveraging rule-based layout engines to continuously generate validated baseline configurations for model training.

Looking toward future developments, the immediate priority is the full integration of this hybrid pipeline into the D^3 :SEA early design platform. A critical aspect of this development will be the systematic expansion of the synthetic database used to train the Variational Autoencoder (VAE) architecture. Enriching the training corpus with a broader variety of hull forms, vessel types, and

complex spatial topologies will enhance the VAE's latent representation, allowing it to capture subtle structural adjacencies and varied volumetric relationships. Ultimately, scaling the database and embedding these features within D^3 :SEA will establish a flexible, intelligent backbone for the next generation of automated vessel design and decarbonization assessment toolkits.

Acknowledgements

With strong gratitude to the EU Horizon SEUS project and its partners: NTNU, NHL Stenden, UTU, SARC B.V., CADMATIC, Gondan Shipyards & Ulstein Shipyards. To the European Union as the main sponsor of the project.

References

BV (2026a), *NR467 Rules for the Classification of Steel Ships*, Bureau Veritas, Paris

BV (2026b), *NR670 Methanol and Ethanol-Fuelled ships*, Bureau Veritas, Paris

DE BERG, M.; CHEONG, O.; VAN KREVELD, M.; OVERMARS, M. (2008), *Computational Geometry: Algorithms and Applications*, Springer, pp.259-270

DE KONING, D.; KOELMAN, H.J.; HOPMAN, H. (2011), *A novel ship subdivision method and its application in constraint management of ship layout design*, COMPIT Conf., Berlin

DE LEÓN, D. (2026), *Research and Innovation Plan: Early Data Driven Ship Design D3: SEA*, Unpublished Report, NHL Stenden University of Applied Sciences

DE LEON, D.; KOELMAN, H. (2026), *Generative Algorithms in Early Ship Design: An Exploration of Hull Subdivision Generation*, International Conf. Computer Applications and Information Technology in the Maritime Industries (COMPIT 2025), Pontignano.

DNV (2019), *Comparison of alternative marine fuels*, DNV, Hovik, https://sea-lng.org/wp-content/uploads/2020/04/Alternative-Marine-Fuels-Study_final_report_25.09.19.pdf

ETHERINGTON, T.R.; PERRY, G.L.W. (2022), *Binary space partitioning generates hierarchical and rectilinear neutral landscape models suitable for human-dominated landscapes*, Landscape Ecology 37, pp.1761-1769

HERDZIK, J. (2023), *Marine Fuel from the Past to the Future*, Scientific Journals of the Maritime University of Szczecin, <https://repository.am.szczecin.pl/handle/123456789/2780>

IMO (2016), *International Code of Safety for Ships Using Gases or Other Low-Flashpoint Fuels (IGF Code)*, Int. Mar. Org., London

OTJENS, S.B. (2026a), *From General Arrangement to Parametric Tool: A Data-driven Approach to Methanol Fuel Integration in Ship Design*, Bachelor Thesis, Maritieme Techniek, NHL Stenden / Maritiem Instituut Willem Barentsz

OTJENS, S.B. (2026b), *Impact of a Methanol Adaptation on the Internal Design*, Unpublished Research Report, NHL Stenden University of Applied Sciences / Maritime Institute Willem Barentsz

SAVVIDES, N.; HALFHIDE, R. (Eds.) (2006–2018), *Significant Ships Series (2006–2018)*, The Royal Institution of Naval Architects, London

SWZ|MARITIME (2026), *A New Era of Digital Ship Design?*, SWZ|Maritime, February

Borderless Virtual Training Simulators in Maritime Industry

Helge Bjordal, Nagelld, Bergen/Norway, helge.bjordal@nagelld.no

Abstract

We all acknowledge that training is important to both build and run ships optimally. Training is typically performed onshore, making it costly for both ship operators, crew members and the environment. New technologies make training possible from anywhere in the world, even on ships sailing on deep sea waters, far away from the common broad-band technologies. Starlink opens new training and operation possibilities. In this paper I will showcase reality and discuss future possibilities. Due to the new broad-band technology, combined with AI-driven multi-user virtual reality content and VR/AR glasses, a new era has emerged in smart collaborative training from ship design to operational crews. I will also show how we can prepare for this new and immersive groundbreaking way of operation and what tech is needed to pull this off. The possibility has arrived, but are the industry ready for it?

1. Introduction

Shipping depends on competence distributed across crews, fleets, yards, equipment manufacturers, class societies and shore organisations. Yet much formal training is still tied to a place: a classroom, a full-mission simulator or a vendor facility. Travel, accommodation, relief planning and lost working time make this model costly. It also creates a contextual gap. A generic engine-room exercise ashore may teach principles well, while leaving the learner to translate them later to a specific vessel, alarm philosophy and team structure.

The combination of low Earth orbit (LEO) broadband, standalone headsets, edge computing and AI-assisted content offers a different operating model: training can follow the ship. Instructors and subject-matter experts may join a distributed exercise; ship-specific digital assets may be used before delivery and throughout operation; and performance data may support debriefing.

This is not a claim that remote VR should replace approved courses, onboard drills or hands-on assessment. It is a claim that the boundary between training, design review and operational support is becoming more permeable.



Fig.1: A picture from “Land to sea virtual collaboration” film

The paper asks “under what technical and organisational conditions can connected immersive training create safe, scalable value across the maritime lifecycle?” It synthesises regulatory guidance, human-factors research and current technology characteristics into (1) a use-case map, (2) a practical reference architecture, (3) a readiness and risk framework, and (4) a phased adoption roadmap.

Because vendor coverage and service performance change over time and place, the architecture avoids dependence on a single advertised throughput figure and treats connectivity as variable rather than guaranteed.

2. Virtual Reality (VR) and Digital twin explained

To fully understand this paper, it is necessary to explain what this paper means by VR. VR is short for Virtual Reality, which means just that. When you put on the VR equipment you are “transported” from your normal reality and into a virtual reality. In this virtual reality you can communicate with other virtual participants and execute tasks. The other virtual participants may be human, like yourself, or digital avatars that are pre-programmed or are AI controlled. The tasks can be designing vessels, training onboard, planning vessel operations and so on.

Most important, VR itself has no limits to what you can do and how you can operate. Limits are created by the technology, budgets, timelines and you imagination.

VR is a part of the XR (Extended Reality) family, which is VR, AR (augmented reality) and MR (mixed reality). VR is already explained in the beginning of this chapter. Meta Quest 3, Pico and Varjo is such products.



Fig.2: Meta Quest 3 VR set

AR means a combination of reality and graphics, where the graphics doesn’t interact with reality. Example: If you “throw” a 3D object into the room it just go “through” any real object, like a table, a chair or even a person without interacting with the objects. XREAL One is such a product.

MR means a combination of reality and graphics, where the graphics interact with reality. Example: If you “throw” a 3D object into the room it will interact when it hits a real object, like a table, a chair or even a person. Microsoft HoloLens and Meta Quest 3 are such products.

A typical VR set consists of a head mask, often called “goggles” and two hand controllers. The VR set can communicate with other VR sets via Wi-Fi or cabled to a computer. The computer then communicates with other VR sets directly, via Wi-Fi, or via another computer with a VR set attached to it. The VR set attached to a computer can use cable or wi-fi, i.e. you can always work “cable-free”.



Fig.3: Modern VR sets are easy to use – no cables required!

Different VR sets have different features and different technical specifications. The most obvious one is visual quality. The higher the resolution of the lenses in the VR set, the more realistic the VR experience can be. However, price and resolution are not necessarily comparable. VR sets also have different amount of storage and different CPU's which influences on the price. VR sets have a price-range from €350 to more than €10 000! Usually, your content deliverer will give you good advice regarding which VR set to choose.

The VR sets can contain the software it runs itself, via the computer it is attached to, via a cloud-based service or via a server located somewhere on the internet. Let's investigate the pros and cons by showing some types of projects that are up and running today.

Digital twin, in its most used form, means that you have a digital (3D) model of a real asset. Typically, this can be a 3D model of a sailing ship. More complex digital twins are 3D models connected to a data-stream from one or more sensors onboard the actual sailing ship. This can be used to plan operations, when to perform maintenance or investigate accidents. More on this later in this paper. Let's look at digital twins without the sensor feed first.

3. Why the opportunity is different now

3.1. Connectivity changes the feasible interaction

Traditional geostationary satellite services can support e-learning and communications, but high latency and constrained capacity have limited natural multi-user interaction and rich media. LEO constellations reduce the signal path and provide substantially more usable broadband in many maritime regions. Starlink markets mobility and maritime service with global-ocean coverage subject to its service map, regulatory approvals and plan conditions, <https://www.starlink.com/business/maritime>. The relevant change is therefore not merely faster downloads. It is the possibility of sustained bidirectional sessions between a vessel, an instructor and other participants.

Even so, a shipboard training system must be designed for variation. Rain fade, obstructions, antenna handovers, congestion, plan limits and coverage restrictions can affect a session. Safety-critical learning must not rely on a continuous cloud connection. The robust pattern is to pre-position the simulation, vessel model and media on an onboard edge server or headset. The wide-area link then carries voice, avatars, instructor commands, compressed state updates and selected video. If the link degrades, the scenario continues locally, records events and resynchronises later.

3.2. Affordable immersive endpoints

Standalone VR headsets have removed the need for a high-end computer at every learner station. Optical hand tracking, pass-through video and inside-out positioning simplify setup, while AR can overlay instructions on physical equipment. These capabilities matter at sea because space, maintenance capacity and time are limited. However, consumer hardware is not automatically suitable for maritime use. Hygiene, replaceable face interfaces, battery management, low-light tracking, heat, storage, motion sickness and safe physical boundaries require explicit controls.

3.3. AI changes content and facilitation

AI can lower the marginal effort required to create variations of a scenario, generate role-specific prompts, support multilingual explanation and identify patterns in event logs. A virtual trainee can practise fault isolation repeatedly without consuming equipment or exposing the plant to risk.

A multi-user scenario can add AI-driven non-player roles when a complete team is unavailable. Yet plausible output is not the same as correct output. Maritime procedures, alarms and limits must come from controlled sources, and consequential feedback must be traceable to approved rules or instructor judgement. NIST frames trustworthy AI as a lifecycle risk-management problem rather than a one-time model selection exercise, *NIST (2013)*.

3.4. Regulation still defines the competence outcome

The International Convention on Standards of Training, Certification and Watchkeeping for Seafarers (STCW) establishes internationally recognised requirements for training, certification and watch-keeping, *IMO (1978)*. Simulation can contribute when it meets applicable performance standards and is used with qualified supervision and assessment. The technology does not itself confer approval.

Operators must distinguish informal familiarisation, company training, evidence for competence development and formal assessment leading to certification. This distinction prevents attractive technology from being mistaken for a compliant training system.

Table I: What has changed and what has not

Layer	New capability	Persistent constraint
Connectivity	Live collaboration far from terrestrial broadband	Variable service; cyber exposure; jurisdiction and plan limits
Immersion	Repeatable spatial and procedural practice	Human factors; physical safety; fidelity and validation
AI	Adaptive scenarios and scalable coaching	Hallucination, bias, explainability and content control
Training system	Learning closer to the workplace	STCW, company procedures and assessor accountability remain

4. Use cases across the ship lifecycle

4.1. Design and construction

Before steel is cut, a collaborative virtual model can place naval architects, yard personnel, equipment suppliers and future crew in the same spatial review. Participants can inspect sightlines, access routes, maintainability, escape paths and control-station ergonomics. A chief engineer may identify that a service point is reachable in a CAD model but impractical with protective clothing and tools. A deck officer may test whether information needed during a manoeuvre is visible from the working position. These reviews do not replace engineering verification, but they can reveal human-system integration issues earlier, when correction is cheaper. Nagelld's software Spawn VR may be used in these cases.

The strongest design use case is not photorealistic presentation. It is structured review with decisions captured against model objects: issue, owner, rationale, due date and verification status. AI may

cluster observations or suggest similar lessons from previous projects, but a named engineer must accept any design decision. Data ownership and export control also matter because the shared environment may contain commercially sensitive vessel and equipment models.

4.2. Commissioning and familiarisation

A vessel-specific environment can be used while the ship is still under construction. Crew members learn locations, routes, interfaces and operating sequences before joining. Equipment vendors can deliver remote sessions using the same model, and recorded exercises can expose knowledge gaps before commissioning pressure peaks. The model must show the correct revision. Training against an obsolete arrangement or alarm response may create negative transfer, so configuration management is a safety function, not an administrative detail.

4.3. Routine operations and maintenance

For routine work, AR can support just-in-time preparation: identify components, rehearse isolation steps and review hazards before the physical task begins. Remote experts can share a learner's view during non-critical troubleshooting. The safe pattern separates rehearsal from execution. During live work, overlays should be concise, hands-free where possible and subordinate to permits, lockout/tagout, toolbox talks and the authority of the responsible officer. The technology should reduce cognitive load, not add another stream of competing instructions.

4.4. Team and emergency training

Multi-user VR is particularly relevant where coordination is the competence of interest: bridge resource management, engine control-room response, enclosed-space rescue, fire-team communication, passenger management and cyber incident response. Participants can join from different vessels or shore locations. The scenario engine can inject equipment failures, ambiguous cues and escalating consequences while capturing communications and decisions for debriefing. Physical drills remain necessary for heat, weight, fatigue, dexterity and equipment handling that a headset cannot reproduce.

4.5. Remote expertise and learning from operations

The same infrastructure can connect training with operations. De-identified fleet events can be converted into scenarios; recurrent errors can inform microlearning; and new procedures can be rehearsed before rollout. This creates a closed learning loop, but it also creates surveillance risk. Performance data gathered for learning should not silently become disciplinary data. Purpose, retention, access and appeal rights need to be defined with crew representatives and privacy specialists.



Fig.4: Picture from “Land to sea virtual collaboration” film

Table II: Example cases and evidence of value

Lifecycle stage	Representative activity	Primary measure
Design	Collaborative access and ergonomics review	Issues found before design freeze
Pre-delivery	Vessel-specific familiarization	Time to verify task proficiency
Operations	Maintenance rehearsal and expert support	First-time completion; avoided delay
Emergency readiness	Distributed team scenario	Decision quality and communication behaviors
Fleet learning	Convert events into scenarios	Recurrence rate and corrective-action closure

5. Reference architecture for deep sea delivery

A workable system is layered. The training service should be useful when the satellite link is excellent, degraded or temporarily absent. It should also be isolated from operational technology (OT), because a training convenience cannot justify a new path into navigation, propulsion or safety systems. Table III is expressed as a control flow rather than a product list.

Table III: Edge-assisted collaborative training architecture

Zone	Core functions	Design rule
Immersive endpoints	VR/AR display, audio, interaction, local safety boundary	Managed devices; kiosk mode; offline scenario copy
Onboard training edge	Session host, content cache, identity proxy, telemetry buffer	Continue locally; encrypt storage; signed content
Ship IT security zone	Firewall, QoS, monitoring, update gateway	No direct route to safety-critical OT
LEO / backup links	Encrypted collaboration traffic and updates	Application tolerates loss, jitter and handover
Shore platform	Instructor console, learning records, content pipeline	Least privilege; audit logs; regional data controls

5.1. Network engineering

Capacity planning should begin with the interaction model. Full-resolution rendered video for several learners is bandwidth-intensive and fragile. State synchronisation is far lighter: each device renders locally while exchanging position, voice and events. Quality-of-service policies should protect operational communications and cap training traffic. A separate business or crew network segment, a secondary link where justified and a clear session-abort policy reduce operational interference. Tests should include realistic sea areas, weather, antenna masking and peak vessel demand—not only a harbour speed test.

5.2. Identity, security and data

Every device, instructor and content package needs an identity. Multi-factor authentication should protect privileged roles, while certificates can authenticate managed devices and servers. Content should be signed, versioned and scanned before distribution. Logs should record who launched a scenario, which procedure revision it used and whether results were synchronised. IMO guidance calls for cyber risk to be addressed within existing safety-management processes, *IMO (2022)*. Relevant threats include compromised headsets, malicious content, credential theft, insecure remote support, data exfiltration and a pivot from the training network toward business or OT systems.

5.3. Content pipeline and digital thread

The content pipeline connects design data, procedures and training without assuming they are interchangeable. Engineering models are often too complex for real-time use and may contain sensitive details. They should be simplified, classified and transformed into a training asset. Each scenario then binds objects to learning objectives, approved procedures, triggers, scoring rules and debrief evidence. When the vessel changes, the scenario enters change control. A digital thread is valuable only if its provenance and revision status are visible.

5.4. AI services

AI services should be modular and bounded. Speech recognition can produce a transcript; a language model can draft a debrief; an analytics model can detect missed checks; and a scenario agent can vary non-critical dialogue. These outputs should be labelled and reviewable. High-consequence judgments - whether a learner is competent, whether a procedure is correct, or whether an action is safe - remain with authorised people. Retrieval from an approved document set is preferable to an unconstrained model, and the system should cite the source paragraph used for any procedural coaching.

5.5. Human factors aboard

A ship is a moving workplace. Headset sessions require a seated or secured area away from active operations, watch duties and escape routes. Session duration, breaks, visual comfort and susceptibility to motion sickness should be managed. Learners need an immediate way to exit and regain situational awareness. An observer should be present for early deployments. Accessibility alternatives are also required for users who cannot tolerate immersive displays. These controls are part of the training design, not footnotes to a hardware purchase.

6. Pedagogy and assurance

6.1. Start with competence, not immersion

A useful scenario begins with an observable performance objective. “Understand the purifier” is too vague; “diagnose a loss-of-pressure alarm, select the correct response and communicate the operational consequence within the stated limits” can be observed and debriefed. The designer then decides which modality is appropriate. Spatial orientation and team coordination may justify VR. A short video may be better for explanation; a physical drill is better for donning breathing apparatus; a supervised onboard task is necessary where tactile skill and real consequences matter.

Immersion can increase presence and engagement, but learning gains depend on instructional design. A systematic review of immersive VR in higher education found promising applications alongside uneven evaluation quality and a need for stronger pedagogical grounding, *Radiani et al. (2020)*. Maritime adopters should therefore resist measuring success through headset hours or learner excitement alone. Evidence should cover knowledge, behaviour, retention and transfer to work.

6.2. Scenario structure

Each scenario should include a briefing, practice opportunity, performance phase and facilitated debrief. The briefing establishes objectives, roles, controls, limitations and the fiction contract. During practice, the system may provide prompts. During assessment, prompts are reduced and the conditions are standardised. The debrief reconstructs what happened, why it happened and what the learner will do differently. Event logs and replay are aids to reflection; they are not a substitute for instructor skill.

6.3. Fidelity should be purposeful

Physical fidelity is not always the goal. A simplified pump room can teach a decision sequence if the relevant cues and consequences are represented. Conversely, small mismatches in an emergency shutdown interface may be unacceptable when muscle memory is being trained. Fidelity should be specified against the task: visual, functional, interaction, data and team fidelity. Validation should involve equipment specialists, experienced seafarers, instructors and, where relevant, the authority or organisation accepting the evidence.

6.4. Assessment and records

Automated scores are useful for timing, sequence and rule-based events, but they can reward superficial compliance. Team competence also includes prioritisation, communication, leadership, workload management and recovery from ambiguity. A defensible assessment combines machine-captured evidence with structured human observation. The record should state the scenario version, conditions, assistance given, assessor, outcome and remediation. When used for formal certification, the applicable STCW and administration requirements govern acceptance, *IMO (1978)*.

6.5. Instructor and designer competence

Remote delivery changes the instructor's job. The instructor must manage technology, observe several channels, maintain psychological safety, detect simulator artefacts and lead an evidence-based debrief. Content teams need maritime expertise, learning design, 3D production, software testing and cyber assurance. Operators should avoid a single "VR champion" becoming an unsupported point of failure. A small cross-functional product team with named owners is more sustainable.

Table IV: Minimum assurance evidence for a scenario release

Evidence	Question answered	Release owner
Objective-to-task map	Does every activity support a defined competence?	Training lead
Source and revision register	Are procedures and model data current?	Technical authority
Functional test	Do cues, controls and consequences behave correctly?	Product owner
Human-factors trial	Is the session usable, tolerable and physically safe?	HSE / human factors
Cyber and privacy review	Are access, interfaces, logs and retention controlled?	Security / privacy
Instructor acceptance	Can the exercise be briefed, observed and debriefed?	Lead instructor

7. Risks and industry readiness

7.1. The main risks are socio-technical

Connectivity failure is visible and can be engineered around. More difficult risks sit between technology and organisation: an unvalidated scenario becomes trusted because it looks realistic; a learner follows an AI suggestion that conflicts with the safety management system; performance data changes employment decisions; a vendor update breaks an approved exercise; or management counts virtual completion as evidence of practical competence. These are governance failures expressed through technology.

Table V: Minimum assurance evidence for a scenario release

Risk	Operational consequence	Key controls
Loss or degradation of link	Interrupted collaboration; incomplete records	Local execution, buffered telemetry, reconnect logic, abort criteria
Incorrect or stale content	Negative transfer; unsafe action	Named source owner, version binding, expiry, acceptance test
Cyber compromise	Data loss or pathway toward ship systems	Segmentation, managed endpoints, least privilege, signed packages
AI error or opacity	Misleading coaching or unfair scoring	Bounded uses, approved retrieval, human review, audit trail
Cybersickness / physical injury	Learner harm; rejection of programme	Screening, seated zone, short sessions, observer, alternatives
Privacy and labour misuse	Loss of trust; legal exposure	Purpose limitation, minimisation, retention rules, consultation
False equivalence with competence	Paper compliance without capability	Blended assessment, physical drills, qualified assessors

7.2. A practical readiness model

Readiness is not a binary answer to whether “the industry” is ready. Fleets differ by vessel type, trade, connectivity, safety culture, digital maturity and regulatory context. An operator is ready for a narrow pilot when it can name an accountable owner, select a low-risk use case, isolate the system, manage content revisions, support devices on board and evaluate learning. It is ready to scale only after proving reliability, transfer and sustainable support across more than one vessel and crew cohort.

Five dimensions should be assessed. Strategic readiness asks whether the problem and value are clear. Technical readiness covers coverage, edge capability, integration, device management and support. Learning readiness covers objectives, instructors, validation and assessment. Governance readiness covers safety management, cyber, privacy, procurement and AI accountability. Workforce readiness covers consultation, accessibility, time allocation and acceptance. A weak score in any one dimension can undermine the whole programme.

7.3. Economics and environmental claims

The business case should compare total systems, not headline travel cost against a headset price. Costs include connectivity, devices, spares, cleaning, content creation, model conversion, licensing, integration, instructor time, cyber controls and lifecycle maintenance. Benefits can include avoided travel, fewer relief disruptions, earlier design findings, faster familiarisation, reduced equipment downtime and more frequent practice. Some benefits are risk reductions rather than cash savings and should be stated as such.

Remote delivery can reduce travel-related emissions, but the net environmental result is not automatically positive. Satellite terminals consume power; devices and servers embody materials and energy; and rich 3D content has a development and data footprint. A credible evaluation compares the avoided travel and facility use with added equipment, electricity and connectivity over an expected lifetime. The paper therefore treats environmental benefit as a hypothesis to measure, not a guaranteed property of digital training.

7.4. Vendor and regulatory dependence

Starlink is an enabling example, not an architectural monopoly. Coverage, terms, terminal approvals and prices may change. Operators should define application requirements, preserve exportable learning records and avoid proprietary content formats where practical. They should also engage flag administrations, recognised organisations, unions and training providers early when a use case approaches formal competence assessment. Portability and regulatory dialogue are readiness controls.

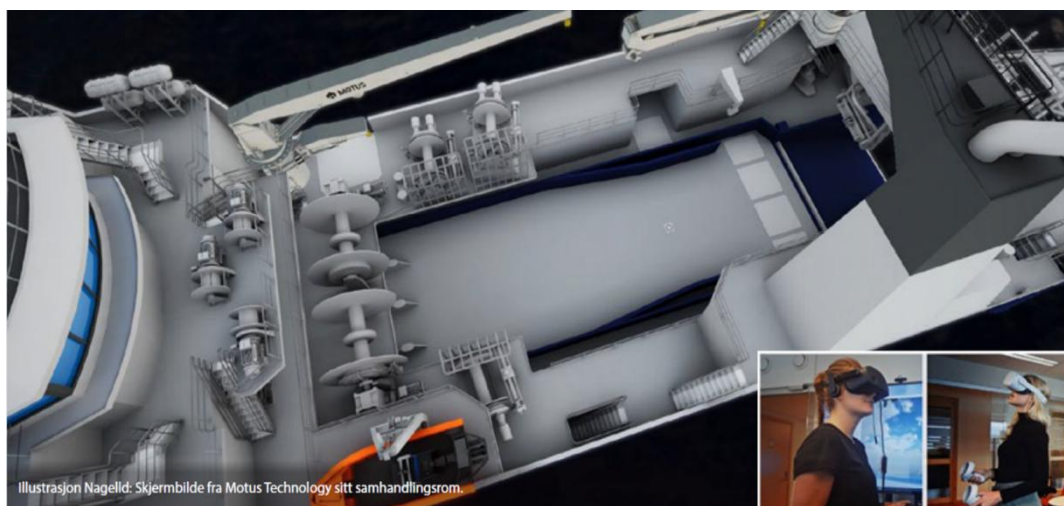


Fig.5: Snow crab vessel in VR

8. Implementation roadmap and pilot design

8.1. Phase 1 Frame the problem

Select one operationally meaningful but non-safety-critical use case, such as vessel familiarisation, a design-access review or rehearsal of a maintenance procedure before supervised work. Establish the current baseline: travel days, time to proficiency, error rate, instructor workload, delay or learner confidence. Define the competence objective and determine what evidence would justify continuing. Map stakeholders and data flows before selecting hardware.

8.2. Phase 2 Build a minimum viable system

Create a small, version-controlled scenario that can run locally. Connect one shipboard group to one remote instructor under controlled conditions. Configure managed identities, a segregated network, content signing, logging and a simple fallback. Perform a hazard assessment covering the physical session area, watchkeeping, device hygiene, accessibility and cyber interfaces. Train the instructor and onboard facilitator before inviting learners.



Fig.6: MOB boat in its cradle

8.3. Phase 3 Conduct an instrumented pilot

Run the exercise across representative users and network conditions. Include a planned loss-of-link event to demonstrate graceful degradation. Collect only the data defined in advance. Compare learning with a reasonable alternative, not with no training. Where sample size is small, use mixed evidence: task performance, structured observations, interviews, system logs and delayed retention checks. Document defects and near misses with the same seriousness as positive feedback.

8.4. Phase 4 Validate transfer and operations

A learner who performs in VR must still demonstrate relevant behaviour in a real or approved simulated setting. Supervisors can use a structured observation during subsequent work, without placing the learner in an unsafe situation. At the same time, measure support burden: failed starts, replacement parts, update time, bandwidth consumed and instructor preparation. A technically impressive pilot that depends on the project team joining every session is not scalable.

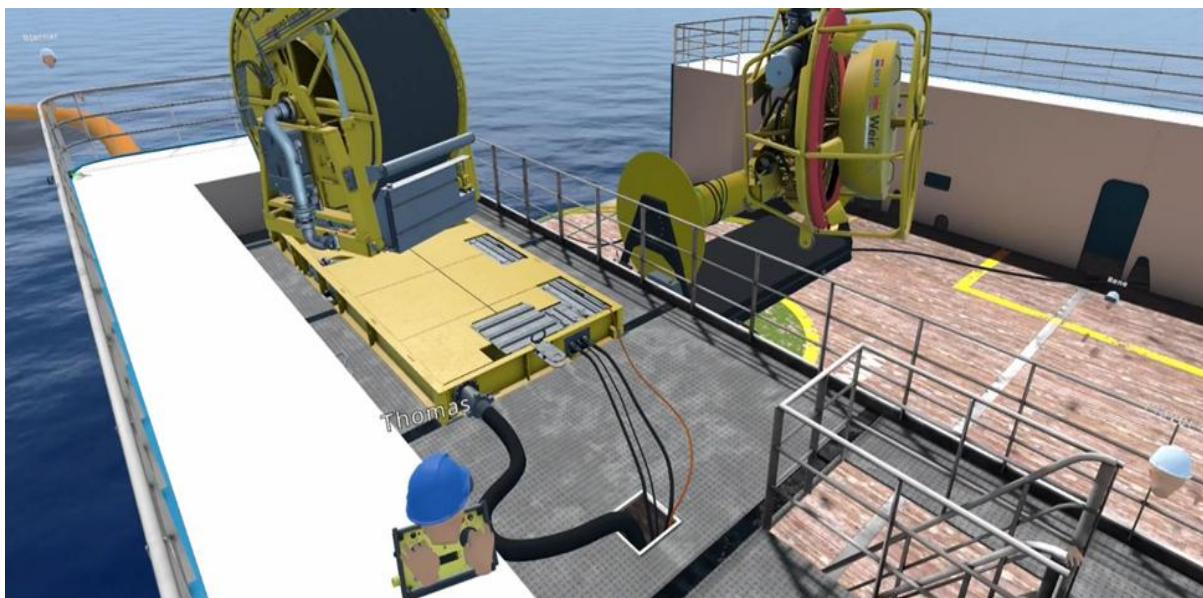


Fig.7: VR collaboration in the NOFO Offshore Oil Spill simulator

8.5. Phase 5 Scale through governance

Scale by standardising the platform services—identity, device management, telemetry, content release and learning records—while allowing vessel-specific content. Establish a release board and periodic revalidation. Define service levels for shore and ship support. Introduce new use cases in risk order and retain the ability to withdraw a scenario quickly. Procurement should require data portability, security evidence, update commitments and clear ownership of converted vessel models.

Table VI: Suggested pilot scorecard

Domain	Example metric	Decision signal
Learning	Critical steps completed; retention after 30 days	Non-inferior or better than current method
Transfer	Observed onboard performance under supervision	Correct action without unsafe prompting
Reliability	Successful starts; reconnect recovery; crash rate	Stable under representative voyage conditions
Human factors	Symptoms, comfort, workload, accessibility	No unresolved high-severity issue
Operations	Instructor prep; support minutes; bandwidth	Supportable by normal organization
Economics	Avoided travel/delay minus lifecycle cost	Credible positive case at target scale
Trust	Crew acceptance; privacy concerns; appeals	Transparent use accepted by stakeholders

Go or no-go gates: A pilot should stop or be redesigned if content accuracy cannot be assured, network isolation is not demonstrated, learners experience unacceptable symptoms, privacy purposes remain ambiguous, or stakeholders propose substituting virtual completion for required practical evidence. It should progress when learning and transfer are credible, the system fails safely, support is repeatable and governance owners accept ongoing responsibilities.

9. Discussion

The central shift is not that ships have acquired another channel for video calls. It is that the ship can become a node in a persistent learning network. Designers, vendors, instructors and crews can interact around a shared representation of the vessel while the vessel is being designed, built and operated. That continuity may shorten feedback loops and make training more specific, frequent and collaborative.

The same continuity creates new obligations. A ship-specific virtual environment can carry sensitive design data. An AI coach can create authority without accountability. A detailed event log can support excellent debriefing and intrusive surveillance at the same time. The architecture must therefore combine local resilience with organisational restraint: collect less, separate networks, define human authority and keep evidence traceable.

The most likely near-term success is a blended model. Shore simulators remain valuable for high-fidelity and regulated assessment; physical drills remain essential for bodily and equipment skills; onboard experts remain responsible for safe work. Connected VR and AR add preparation, repetition, access to scarce expertise and collaborative review between these established modes. In this model, the technology extends the training system rather than claiming to replace it.



Fig.8: A picture from “Land to sea virtual collaboration” film

10. Conclusion

The possibility has arrived, but industry readiness is uneven. LEO broadband can support live collaboration at sea, and modern immersive hardware can run capable simulations without continuous cloud rendering. AI can help vary scenarios, translate interaction and analyse evidence. Together they make distributed, vessel-specific training feasible across much of the ship lifecycle.

Readiness, however, is demonstrated by controls and outcomes rather than procurement. Operators need an edge-first architecture, separation from operational technology, assured and versioned content, bounded AI, qualified instructors, humane data practices and evidence that learning transfers to work. The prudent route is to start with a narrow use case, test under realistic voyage conditions, include failure scenarios and scale only when safety, learning, reliability and supportability are proven. The competitive advantage will not belong simply to the fleets with the most headsets or bandwidth. It will belong to organisations that turn connectivity into a disciplined learning capability



Fig 9: A picture from “Land to sea virtual collaboration” film

Acknowledgements

All VR work presented in this paper is developed by Nagelld. Nagelld is recognized as one of the frontrunners of VR experiences to the maritime industries worldwide. One of our VR training scenarios won “The Jury Award” at Nordic VR Forum 2024. In 2023 we won “Best VR training experience” at Nordic VR Forum for a cruise ship lifeboat VR training experience. Nagelld has also developed Spawn, a metaverse collaboration platform for design reviews and various operation and training scenarios.

References

- IMO (1978), *International Convention on Standards of Training, Certification and Watchkeeping for Seafarers (STCW)*, Int. Mar. Org., London
<https://www.imo.org/en/ourwork/humanelement/pages/stcw-conv-link.aspx>
- IMO (2022), *Guidelines on Maritime Cyber Risk Management*, MSC-FAL.1/Circ.3/Rev.2,
<https://www.imo.org/en/ourwork/security/pages/cyber-security.aspx>
- NIST (2013), *Artificial Intelligence Risk Management Framework*, NIST AI 100-1, National Institute of Standards and Technology, <https://doi.org/10.6028/NIST.AI.100-1>
- RADIANTI, J.; MAJCHRZAK, T.A.; FROMM, J.; WOHLGENANNT, I. (2020), *A systematic review of immersive virtual reality applications for higher education*, Computers & Education 147, <https://doi.org/10.1016/j.compedu.2019.103778>

Data-Driven Prediction of Residual Resistance of Platform Supply Vessels

Thor Albrektsen, SINTEF Ocean, Trondheim/Norway, thor.albrektsen@sintef.no

Endre Sandvik, SINTEF Ocean, Trondheim/Norway, endre.sandvik@sintef.no

Øyvind Rabliås, SINTEF Ocean, Trondheim/Norway, oyvind.rablias@sintef.no

Sverre Steen, NTNU, Trondheim/Norway, sverre.steen@ntnu.no

Abstract

This paper applies machine learning to the prediction of calm-water residual resistance for Platform Supply Vessels. Using an anonymized selection of SINTEF Ocean's model-test database, feed-forward neural networks are trained to predict the residual resistance coefficient C_R from hull geometry, and the input set is extended with features derived from the sectional area curve. On hull forms held out from training, over-parameterized networks reach an R-squared of about 0.81 from as few as three hull parameters and improve on both empirical baselines over the Froude-number range of the training data. Adding further hull ratios does not raise hold-out accuracy, and the sectional-area-curve features lower it to about 0.74, indicating that with 42 distinct hull forms the detailed geometry is fitted hull by hull rather than learned as a transferable trend. Three models are recommended for different design phases, giving accurate C_R estimates from as little as three hull parameters.

Nomenclature

L	Length
L_{PP}	Length between perpendiculars
L_{WL}	Waterline length
B	Beam
T	Draught
S	Wetted surface area
V	Ship speed
g	Acceleration of gravity
F_N	Froude number
C_T	Total resistance coefficient
C_F	Frictional resistance coefficient (ITTC-57 line)
C_R	Residual resistance coefficient
C_W	Wave resistance coefficient (Holtrop–Mennen)
C_{AA}	Air resistance coefficient
C_{BD}	Transom drag coefficient
C_{App}	Appendage resistance coefficient
C_B	Block coefficient
C_M	Midship section coefficient
C_P	Prismatic coefficient
$1 + k_0$	Form factor
LCB	Longitudinal centre of buoyancy
$TRIM$	Trim
L/B	Length-to-beam ratio
B/T	Beam-to-draught ratio
L/T	Length-to-draught ratio
$S/(L \cdot B)$	Wetted-surface ratio
LCB/L_{WL}	Longitudinal centre of buoyancy, non-dimensionalised by waterline length
ANN	Artificial neural network
CFD	Computational fluid dynamics
OP-NN	Over-parameterized neural network
PSV	Platform supply vessel
SAC	Sectional area curve

1. Introduction

Accurate prediction of ship resistance is essential in both the design and operational phases of a vessel. Resistance is closely tied to fuel efficiency, propulsion requirements, and environmental impact, and it must often be estimated early in a project, before detailed hull geometry or full-scale measurements are available. Reliable and fast prediction methods are therefore an important part of the design process.

Traditionally, empirical methods such as those of *Holtrop and Mennen (1982)* and *Hollenbach (1998)* have been widely used because they are simple and practical. They are built on regression of model-test data and rely on a small number of input variables. This makes them suitable for quick estimates, but it also limits their ability to capture finer geometric features that influence hydrodynamic performance. Some of these methods also demand parameters that are not readily available in early design, which restricts their usefulness precisely where a fast estimate is most wanted.

Computational fluid dynamics (CFD) resolves the flow around the hull with high fidelity and can capture subtle geometric effects, but it requires substantial computational resources and expert setup, so it is generally reserved for later design stages. Model testing remains the benchmark for validation, yet it is time-consuming and costly and is likewise concentrated in the later phases of a project. This leaves a gap between accessible but limited empirical methods and accurate but expensive high-fidelity tools.

Machine learning, and in particular artificial neural networks (ANNs), is a promising way to bridge this gap. Neural networks learn complex, non-linear relationships from data and can give robust estimates from limited input, while still being able to exploit richer geometry when it is available. This adaptability makes them a useful complement to established methods, combining the simplicity of empirical formulas with accuracy closer to that of high-fidelity simulation.

This study uses a subset of SINTEF Ocean database from model tests, specifically an anonymized selection of main dimensions and results from towing tests. The study focuses exclusively on Platform Supply Vessels, a class that is well represented in the database but has received little attention in the resistance-prediction literature. The objective is to develop and validate a data-driven method for predicting the residual resistance coefficient C_R , to test whether features derived from the sectional area curve improve prediction, and to compare the resulting models with two established empirical baselines, Holtrop–Mennen and Hollenbach, Table I.

1.1. Related work

Regression on systematic-series and model-test data has a long history, *Shaher Sabit (1971)*, *Radojcic et al. (2014)*, and data-driven resistance prediction has received growing attention. *Koushan (2001)* trained an ANN to estimate residual resistance and wetted surface on the same internal database, using a compact set of hull ratios and the Froude number. More recently, *Margari et al. (2018)* applied ANNs to the MARAD systematic series, and *Cepowski (2020)* predicted added resistance in waves at the preliminary design stage. *Yildiz (2022)* addressed trimaran residual resistance, and *Yang et al. (2024)* compared several machine-learning models across ship types. Two recent NTNU theses used SINTEF Ocean’s database for other vessel classes: *Wolleb (2023)* for fast catamarans and *Kleppestø (2024)* for fishing vessels. Most of these studies rely on standard geometric descriptors or parametric hull series. The present work differs by focusing on a single vessel class and by testing whether more advanced, derived hull-form features improve prediction.

1.2. Methods compared

For brevity, the three data-driven models are referred to as Predictors A–C throughout, and the two empirical baselines by their authors’ names, as defined in Table I.

Table I: Methods compared in this study and the labels used in the figures and text.

Label	Type	Definition
Predictor A	This work	Neural network, 3 inputs: F_N , C_B , L/B (rough estimation)
Predictor B	This work	Neural network, 9 inputs: F_N , C_B , L/B , B/T , $S/(L \cdot B)$, C_M , C_P , LCB/L_{WL} , $TRIM$ (preliminary design)
Predictor C	This work	Neural network, 11 inputs: replaces trim with L/T and adds bulb area and aft-shoulder position (advanced design)
Holtrop–Mennen	Empirical	<i>Holtrop and Mennen (1982)</i>
Hollenbach	Empirical	<i>Hollenbach (1998)</i>

2. Resistance and empirical prediction

The total calm-water resistance of a ship is commonly expressed through the total resistance coefficient C_T , which is split into a viscous part and a residual part. In the standard form-factor method, *ITTC (2017)*, *Birk (2019)*, the viscous resistance is obtained by scaling the frictional coefficient C_F (from the ITTC-57 correlation line) by a constant form factor $(1 + k_0)$, assumed identical for model and full scale. The residual resistance coefficient C_R then collects the remaining contributions, dominated by wave-making and viscous pressure (form) drag:

$$C_R = C_T - (1 + k_0) C_F - C_{AA} - C_{BD} - C_{App}$$

where C_{AA} , C_{BD} and C_{App} are the additional model-scale coefficients used in the SINTEF Ocean procedure - air resistance, transom drag and appendage resistance — and C_R is assumed equal for model and ship. Because C_R carries the speed- and shape-dependent part of the resistance, it is the natural target for a geometry-driven prediction method. The wave-making behaviour is governed by the Froude number:

$$F_N = \frac{V}{\sqrt{gL}}$$

where V is the ship speed, g the acceleration of gravity and L the length. The Froude number is the single most important predictor of C_R .

Empirical methods estimate these components from regression formulas. The two baselines used here, Table I, are the industry-standard method of *Holtrop and Mennen (1982)* and the updated single- and twin-screw model of *Hollenbach (1998)*. Both are fast and require few inputs, but they use a limited set of variables and therefore cannot resolve the effect of detailed hull features. They are also only valid within the range of the hull forms they were fitted to; outside that range the predictions can become unreliable. These limitations motivate a method that is specialised to one vessel class and can make use of more geometric detail.

3. Data and feature extraction

The data comes from SINTEF Ocean’s internal model test database, which stores model-test resistance results together with main particulars and hull geometry in the form of transverse sections. For this study the PSV subset was extracted and cleaned into a resistance dataset. Each entry corresponds to one model, one loading condition (waterline) and one speed, so that a single hull contributes several rows across its tested speed range. After cleaning, the base dataset contains 960 speed points from 111 model runs. All results are treated in an anonymised, non-dimensional form, both because the underlying test data is commercially sensitive and because non-dimensional inputs simplify comparison between hulls.

Standard hull descriptors were used as a starting point: the Froude number F_N , block coefficient C_B , midship and prismatic coefficients C_M and C_P , length-to-beam L/B , beam-to-draught B/T , length-to-draught L/T , wetted-surface ratio $S/(L \cdot B)$, longitudinal centre of buoyancy LCB (non-dimensionalised by the waterline length, LCB/L_{WL}), and trim. To go beyond what conventional methods can represent,

additional features were extracted from the sectional area curve (SAC) of each hull, for every available waterline.

3.1. Features from the sectional area curve

The SAC describes how the submerged cross-sectional area varies along the length of the hull, and it encodes information about the bulbous bow, the transom, and the fore and aft shoulders. Fig.1 shows the points from which features were extracted for a representative hull. The extraction rules were found iteratively and the resulting values were reviewed and, where necessary, corrected by hand.

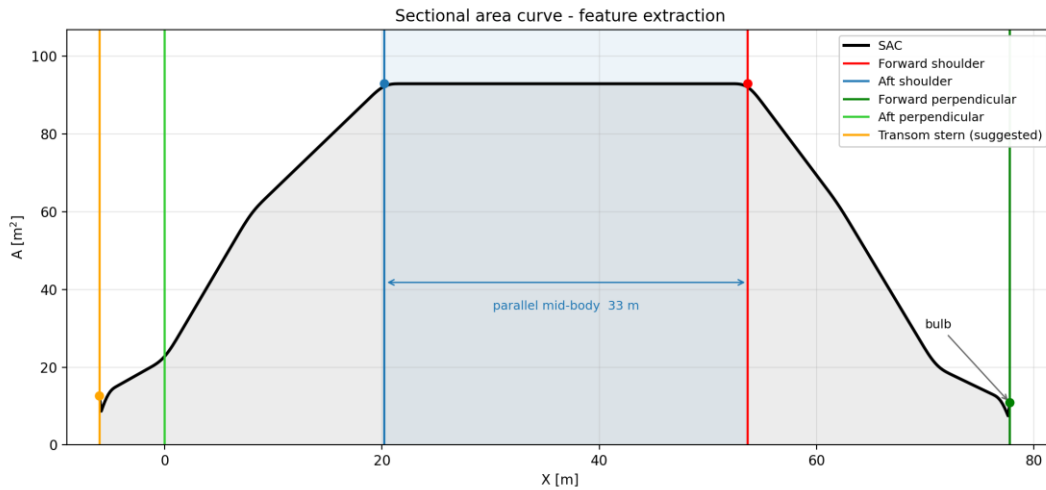


Fig.1: Feature extraction from the sectional area curve (SAC), reconstructed from the extracted values of a representative hull: fore and aft shoulders, fore and aft perpendiculars, and the suggested transom stern area.

The bulbous bow is represented by its transverse sectional area, taken at the fore perpendicular and non-dimensionalised by the product $B \cdot T$; the transom is represented by the submerged sectional area near the aft perpendicular, likewise non-dimensionalised by $B \cdot T$. The shoulders are the longitudinal positions where the SAC reaches its maximum area; they were located using a fixed tolerance on the slope of the curve. For each shoulder, three quantities were extracted, namely the longitudinal position, the sectional area, and the local slope of the SAC. The slope at the forward shoulder is of particular physical interest because it reflects the pressure gradient in that region and is linked to wave generation. Finally, the length of parallel middle body between the shoulders was recorded. Shoulder positions and lengths were non-dimensionalised by L_{PP} , and slopes by the waterline beam.

The SAC could be reconstructed for 87 of the model runs, giving a smaller merged dataset (963 speed points from 42 hull forms, including one synthetic zero-speed anchor per run, Section 4) whenever the advanced-geometry features are used. This reduction in sample size is one of the trade-offs of working with more detailed geometry.

4. Neural network models

The prediction models are feed-forward neural networks (multi-layer perceptrons). Networks of this type can represent arbitrary continuous relationships given sufficient capacity, *Hornik et al. (1989)*. The networks were trained with the Adam optimizer, *Kingma and Ba (2015)*, with a learning rate of $1e-3$ and a batch size of 32. Hidden layers use ReLU activations and the output layer is linear. Dropout (rate 0.2) and early stopping on the validation loss with restoration of the best weights were used to limit overfitting. Models were trained on 963 speed points from 87 towing-tank runs covering 42 hull forms, including 87 synthetic zero-speed points ($F_N = 0$, $C_R = 0$), one per run. Inputs and target were standardised with a StandardScaler fit on the training rows only. The train/validation/test split was grouped by hull form, so that all speed points and loading conditions of one hull fall on the same side

of the split, and every configuration was trained with three random seeds; accuracy is reported as the mean and standard deviation of the test-set R-squared over those seeds. The reported metrics therefore measure generalisation to hull forms not seen in training. An earlier random split over individual speed points, in which every test hull was also present in the training set, gave R-squared values of 0.90–0.98 for the same configurations; those figures measure within-hull interpolation only and are used here solely for the architecture sweep in Section 4.1, where the relative ranking of network sizes is of interest rather than the absolute scores.

4.1. Network size and over-parameterization

The study began with a broad joint sweep over input variables and network architecture together, rather than tuning them in isolation. Two base feature sets were used - a minimal set (F_N , C_B , L/B) and a larger hydrodynamic set adding L/T , $S/(L \cdot B)$, B/T , C_M and C_P - and each was extended with every cross-combination of up to three further descriptors drawn from the remaining hull ratios and SAC-derived geometry. This gave 101 feature combinations for the small base and 125 for the larger one, each trained across several architectures ranging from a compact [20, 10] network to wide over-parameterized layouts ([1024, 512], [1024, 1024] and [2048, 1024]) - 401 and 377 runs respectively, 778 in total. The purpose was to map in a single pass which input sets and network sizes were worth pursuing, before the more focused size and over-parameterization studies described below.

A systematic size study trained a large number of architectures on two benchmark input sets. The clearest result was that the best-performing networks had far more trainable parameters than there were data points. Rather than overfitting, these large networks generalised well. This is consistent with the double-descent behaviour reported by *Belkin et al. (2019)*, where test error first rises and then falls again as model capacity grows past the interpolation point, especially when early stopping and regularization are applied.

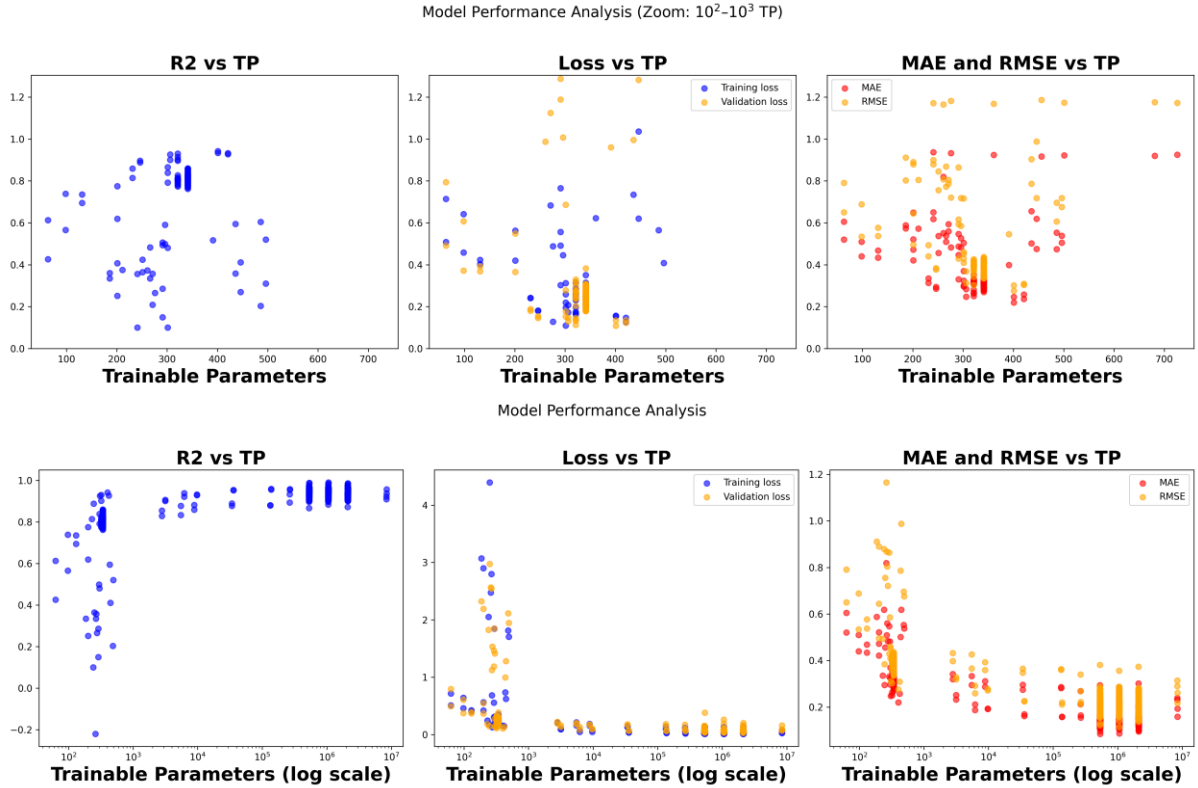


Fig.2: Model performance (R-squared, loss, MAE/RMSE) against the number of trainable parameters for all trained networks. Accuracy improves with capacity, with a dip near the interpolation threshold.

This motivated a dedicated study of over-parameterized networks (OP-NNs) with two wide hidden layers. Fig.2 shows R-squared, loss and error against the number of trainable parameters for all networks trained in the initial sweep. The sweep was run under the earlier random split over speed points (Section 4), so the absolute scores in Fig.2 are within-hull values and higher than those in Table II; the sweep was not repeated under the grouped protocol, and only the trend with capacity is read from it. There is a clear improvement in accuracy as capacity increases, aside from a dip near the point where the number of parameters is comparable to the dataset size, which is consistent with a double-descent transition, although Fig.2 pools networks trained on different input sets and cannot isolate capacity as the only variable. A follow-up variable sweep over many input combinations confirmed that wide, over-parameterized networks performed best, and that a compact set of inputs was often enough to reach the top of the ranking.

5. Results and validation

The models are assessed in three complementary ways: against the reference methods on a representative mean hull, against unseen model-test data for a single ship, and against full-scale CFD for a real vessel. Throughout, emphasis is placed not only on statistical scores but also on whether the predicted C_R curves behave as expected, in particular whether C_R rises with F_N as expected.

5.1. Comparison with empirical methods

A synthetic mean hull was built from the dataset averages and evaluated over a wide Froude-number range. Fig.3 compares Predictors A–C with the two reference methods, together with the cloud of measured C_R values from all ships. The predictor curves follow the central trend of the data closely. Holtrop–Mennen underestimates strongly at low Froude numbers. This is largely definitional: *Holtrop and Mennen (1982)* apply a form factor, so their residual term is a wave resistance coefficient C_W rather than C_R , and excludes the viscous pressure drag that C_R retains. The curves are deliberately shown over a wide range to illustrate how the methods behave beyond their intended scope. The networks show some unphysical behaviour only in the very low range (below $F_N \approx 0.15$), where training data is sparse.

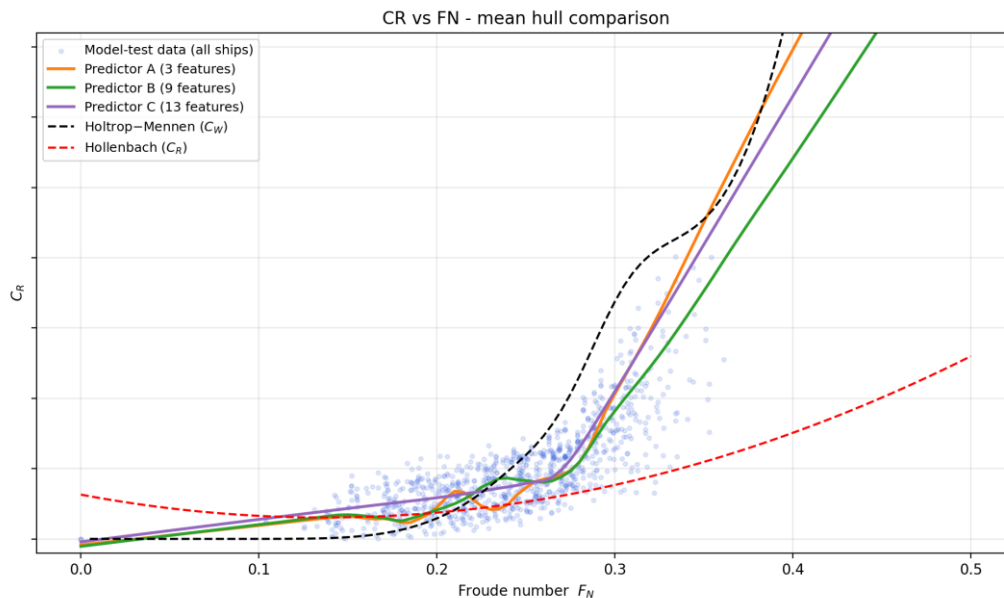


Fig.3: C_R versus Froude number for the mean hull. Predictors A–C (solid) compared with Holtrop–Mennen and Hollenbach (dashed) and the measured data cloud.

5.2. Test-set accuracy

Table II summarises the hold-out accuracy of the candidate configurations, including the three recommended predictors, on hull forms not seen in training. Three basic parameters already give an R-squared of 0.81 ± 0.04 . The nine-parameter preliminary-design sets reach 0.81 ± 0.02 (L/T variant) and 0.81 ± 0.03 (trim variant), i.e. no measurable gain, and the eleven-parameter advanced-design set, which replaces trim with L/T and adds the bulb area and aft-shoulder position from the sectional area curve, falls to 0.74 ± 0.05 ; adding trim and the fore-shoulder position as well (13 inputs) lowers it further to 0.71 ± 0.06 . Values are the mean and standard deviation of the test-set R-squared over three random seeds with whole hull forms held out. Within the seed-to-seed spread the three- and nine-input models are indistinguishable, and the SAC-derived features reduce hold-out accuracy rather than improving it.

Table II: Hold-out accuracy of candidate configurations on merged dataset. Grouped split: train/validation/test split grouped by hull form, so that all speed points of a hull fall on same side; mean $\pm$ standard deviation of test-set R-squared over 3 random seeds. 5-fold CV: every hull form held out exactly once. Standardiser fit on training rows only. Predictors A–C as defined in Table I.

Use case	Input features	Architecture	R-squared, grouped split (mean $\pm$ s.d., 3 seeds)	R-squared, 5-fold CV by hull form
Rough estimation (Predictor A)	$F_N, C_B, L/B$	3-1024-512-1	0.813 ± 0.036	—
Rough estimation, wider	$F_N, C_B, L/B$	3-1024-1024-1	0.808 ± 0.029	—
Preliminary design, L/T variant	$F_N, C_B, L/B, L/T, S/(L \cdot B), B/T, C_M, C_P, LCB/L_{WL}$	9-2048-1024-1	0.814 ± 0.022	0.769 ± 0.047
Preliminary design (Predictor B)	$F_N, C_B, L/B, B/T, S/(L \cdot B), C_M, C_P, LCB/L_{WL}, TRIM$	9-1024-1024-1	0.809 ± 0.031	0.750 ± 0.099
Advanced design (Predictor C)	$F_N, C_B, L/B, L/T, S/(L \cdot B), B/T, C_M, C_P, LCB/L_{WL}$, bulb area, aft-shoulder position	11-1024-512-1	0.744 ± 0.047	—
Advanced design, extended	+ $TRIM$, fore-shoulder position	13-2048-1024-1	0.712 ± 0.059	—

As a stricter check, the two nine-input configurations were also evaluated by five-fold cross-validation grouped by hull form, in which every hull form is held out exactly once. This gives R-squared values of 0.77 ± 0.05 for the L/T variant and 0.75 ± 0.10 for the trim variant (Predictor B), slightly below the single grouped split; the larger spread for the trim variant indicates that a few hull forms are markedly harder to predict than the rest.

The synthetic zero-speed anchor points described in Section 4 (one per run, enforcing $C_R = 0$ at $F_N = 0$) improve consistency in the sparse low-speed region. Because they are assigned to the same side of the split as their hull, some fall in the test set and contribute to the metrics in Table II, which makes the statistical picture slightly optimistic.

5.3. Single-ship validation

A model that was not part of training, with data across its tested speed range, was used for a direct comparison. Fig.4 shows C_R against F_N for Predictors A–C, the reference methods, and the measured

(original) C_R . The first points at very low F_N are validation anchors. The predictors give realistic predictions across the range and follow the measured C_R more closely than either reference method, reproducing the expected exponential rise of C_R while Holtrop–Mennen stays near zero until high speed and Hollenbach remains comparatively flat.

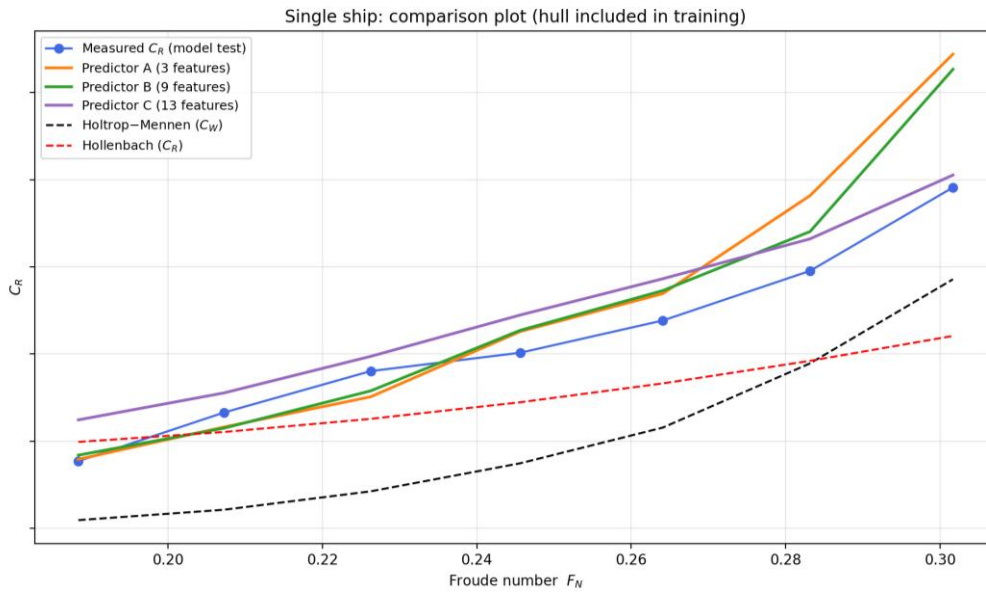


Fig.4: Single unseen ship (waterline 0). Predictors A–C against the measured C_R and the Holtrop–Mennen and Hollenbach estimates.

5.4. Full-scale CFD case study

Finally, the models were tested against a full-scale CFD study of a real PSV in operation, in which resistance was validated against measured power. The vessel is representative of the dataset but is not among the hulls used for training, which makes it a useful independent check. Fig.5 shows the predictor curves together with the CFD values and the two reference methods over the operating range. The predictors lie on either side of the CFD points and follow their trend far more closely than Holtrop–Mennen, which diverges sharply at the upper end. As this is a single vessel, and the CFD results are themselves validated against measured power rather than against model tests, the case study should be read as a plausibility check rather than as a quantitative validation.

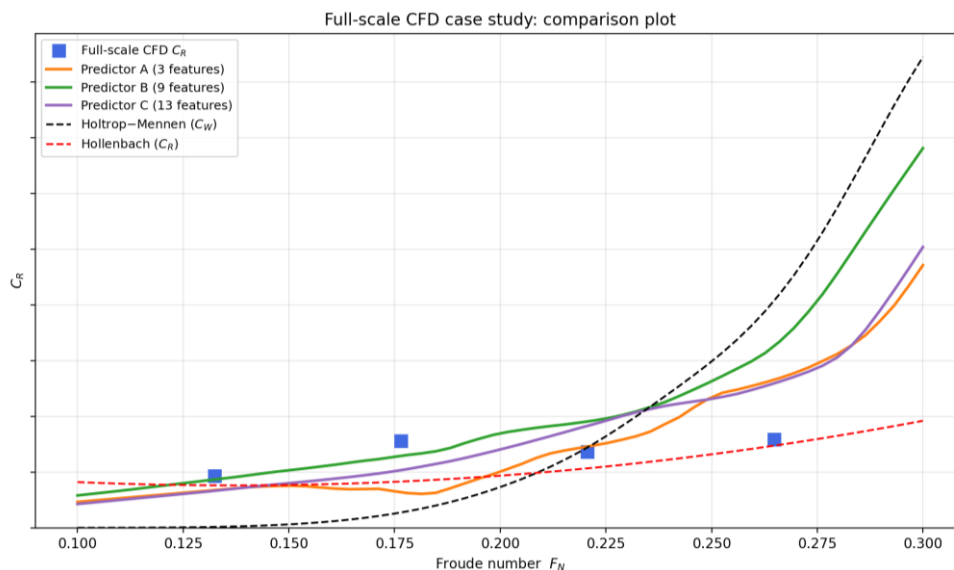


Fig.5: Full-scale CFD case study. Predictors A–C compared with CFD-derived C_R and the reference methods over the operating range.

5.5. Effect of advanced geometry

The original hypothesis was that adding features from the sectional area curve would improve prediction. On hull forms held out from training the opposite was found: R-squared fell from 0.81 with the nine-input sets to 0.74 ± 0.05 when the bulb area and aft-shoulder position were added, and to 0.71 ± 0.06 when trim and the fore-shoulder position were added as well. With only 42 distinct hull forms, features that vary mainly between hulls give the network a way to fit hull-specific behaviour that does not transfer to new hulls. The same mechanism explains why an earlier random split over speed points, in which the test hulls were also present in training, had shown the SAC features improving the score: there the features helped the network recognise hulls it had already seen, which is precisely what a split grouped by hull form removes. The extra inputs did not destabilise the networks — the predicted C_R curves in Figs.3-5 remain physically plausible — but on present evidence detailed SAC geometry should be added only once a substantially larger set of hull forms is available.

6. Recommended models by design phase

Based on the full set of experiments, three predictor configurations are recommended, one for each stage of design. They differ in how much hull information they require and in how flexible they are.

Rough estimation: A stiff, robust model using only F_N , C_B and L/B . All inputs follow directly from the main particulars (length, displacement, beam, draught, speed), so a usable C_R estimate is available almost immediately.

Preliminary design: A flexible model using F_N , C_B , L/B , B/T , $S/(L \cdot B)$, C_M , C_P , LCB/L_{WL} , $TRIM$. All inputs can be derived from standard design parameters, with trim set to the intended design condition (typically even keel), letting the designer study their effect on resistance during dimensioning.

Advanced design: A model that replaces trim with L/T and adds the non-dimensional bulbous-bow area and aft-shoulder position to the main hull parameters, so that no loading-condition input is required. On held-out hull forms it did not match Predictors A and B (Table II); it is retained as the configuration for exploring SAC-level design changes that the coarser models cannot see, with the caveat that its absolute accuracy is lower until more hull forms are available.

Table III: Recommended predictors (A–C) for different design phases

Use case	Model	Input features	Architecture
Rough estimation	Predictor A	F_N , C_B , L/B	3-1024-512-1
Preliminary design	Predictor B	F_N , C_B , L/B , B/T , $S/(L \cdot B)$, C_M , C_P , LCB/L_{WL} , $TRIM$	9-1024-1024-1
Advanced design	Predictor C	F_N , C_B , L/B , L/T , $S/(L \cdot B)$, B/T , C_M , C_P , LCB/L_{WL} , bulb area, aft-shoulder position	11-1024-512-1

7. Conclusions and further work

This paper presented a data-driven method for predicting the residual resistance coefficient of Platform Supply Vessels using feed-forward neural networks trained on SINTEF Ocean’s internal model test database. The main finding is that over-parameterized networks reach an R-squared of about 0.81 on hull forms held out from training using only three hull parameters, and give lower errors than both reference methods on held-out model-test data and in the full-scale CFD case study, although the reference methods were in part evaluated outside the hull-form range for which they were fitted. Adding further hull ratios did not raise hold-out accuracy, and features derived from the sectional area curve lowered it to about 0.74, which points to the number of distinct hull forms (42), not model capacity, as the binding constraint. The networks nevertheless remained stable when detailed geometry was added, so the same framework can be carried from the earliest estimate through to detailed design as more hull

forms become available. Three models are recommended for different design phases, giving accurate and stable C_R predictions from as few as three basic hull parameters.

Several directions remain open. The same framework can be extended to the propulsion dataset to predict thrust deduction, wake fraction and propulsive efficiency, which together with resistance would give a complete data-driven power estimate. A physics-informed loss term, for example enforcing that C_R tends to zero as F_N tends to zero, could improve behaviour in the sparse low-speed range. Finally, broadening the dataset beyond PSVs, with ship type as an additional input, could yield more general models while preserving type-specific accuracy.

Acknowledgements

This work was funded by the Norwegian Research Council through the project ‘Towards sustainable zero-emission offshore logistics’ (project no. 344364) led by SINTEF Ocean and builds on the first author’s master’s thesis at NTNU. The authors thank Kristian Sagmo for providing the CFD results used in the case study.

References

- BELKIN, M.; HSU, D.; MA, S.; MANDAL, S. (2019), *Reconciling modern machine-learning practice and the classical bias-variance trade-off*, Proc. National Academy of Sciences 116/32, pp.15849-15854
- BIRK, L. (2019), *Fundamentals of Ship Hydrodynamics: Fluid Mechanics, Ship Resistance and Propulsion*, Wiley
- CEPOWSKI, T. (2020), *The prediction of ship added resistance at the preliminary design stage by the use of an artificial neural network*, Ocean Eng. 195, 106657
- HOLLENBACH, K.U. (1998), *Estimating resistance and propulsion for single-screw and twin-screw ships*, Ship Technology Research 45/2, pp.72-76
- HOLTROP, J.; MENNEN, G. (1982), *An approximate power prediction method*, Int. Shipbuilding Progress 29/335, pp.166-170
- HORNIK, K.; STINCHCOMBE, M.; WHITE, H. (1989), *Multilayer feedforward networks are universal approximators*, Neural Networks 2/5, pp.359-366
- ITTC (2017), *1978 ITTC Performance Prediction Method, ITTC Recommended Procedures and Guidelines 7.5-02-03-01.4*, International Towing Tank Conference
- KINGMA, D.P.; BA, J. (2015), *Adam: A method for stochastic optimization*, 3rd Int. Conf. Learning Representations (ICLR)
- KLEPPESTØ, K. (2024), *Empirical prediction of resistance of fishing vessels*, Master Thesis, NTNU, Trondheim
- KOUSHAN, K. (2001), *Empirical prediction of ship resistance and wetted surface area using artificial neural networks*, Practical Design of Ships and Other Floating Structures, Elsevier, pp.501-507
- MARGARI, V.; KANELLOPOULOU, A.; ZARAPHONITIS, G. (2018), *On the use of artificial neural networks for the calm water resistance prediction of MARAD Systematic Series hullforms*, Ocean Eng. 165, pp.528-537
- RADOJCIC, D.; ZGRADIC, A.; KALAJDZIC, M.; SIMIC, A. (2014), *Resistance prediction for hard chine hulls in the pre-planing regime*, Polish Maritime Research 21/2, pp.9-26

SHAHER SABIT, A. (1971), *Regression analysis of the resistance results of the B.S.R.A. series*, Int. Shipbuilding Progress 18/197, pp.3-17

STEEN, S. (2014), *General modelling and scaling laws*, Lecture notes, NTNU, Trondheim

WOLLEB, E. (2023), *Empirical prediction of residuary resistance of fast catamarans*, Master Thesis, NTNU, Trondheim

YANG, Y.; ZHANG, Z.; ZHAO, J.; ZHANG, B.; ZHANG, L.; HU, Q.; SUN, J. (2024), *Research on ship resistance prediction using machine learning with different samples*, J. Marine Science and Engineering 12/4, 556

YILDIZ, B. (2022), *Prediction of residual resistance of a trimaran vessel by using an artificial neural network*, Brodogradnja 73/1, pp.127-140

Toward a Trustworthy Shipyard Digital Twin: Spatio-Temporal Grounding for LLM Surveillance Video Understanding

Chenwei Gui, The University of Tokyo, Tokyo/Japan, gui@m.sys.t.u-tokyo.ac.jp
Kazuhiro Aoyama, The University of Tokyo, Tokyo/Japan, aoyama@race.t.u-tokyo.ac.jp

Abstract

Shipyard digital twins need trustworthy process information from production sites, where fixed-camera surveillance video is a practical source. LLM-based video understanding can interpret such video, but generic video agents lack the shipyard's vocabulary of workers, activities and work locations and may return plausible but ungrounded answers. This study proposes a spatio-temporally grounded process memory that anchors shipyard-specific observations to explicit times, work locations and source frames, from which an LLM agent answers process questions with verifiable evidence. On a small test set, accuracy rose from 32.5% to 55.0% and grounded accuracy from 7.5% to 42.5% over a VideoAgent-style baseline.

1. Introduction

Shipyards need timely information about production progress and ongoing work to coordinate resources and respond to deviations, and digital twins provide a means of bringing this information into production decision making. A shop-floor digital twin is a virtual representation of the physical production system, connected to the shop floor through data and services that describe its operation, *Tao and Zhang (2017)*. For that representation to follow the process as it actually unfolds, it requires observations from the production site rather than plans and aggregate records alone. Monitoring provides these observations by converting physical activities into process information that can be incorporated into the twin.

Shipyards obtain this process information through a range of monitoring methods. Progress management once relied mainly on manually collected data and the judgement of supervisors, while markers on components, welding-machine time series and image analysis were later introduced to measure assembly and welding progress directly, *Kim et al. (2015)*, *Choi and Seo (2020)*. Fixed cameras have further enabled systems to observe blocks, welding activity and workers within a work area, and to organise what they observe into progress indicators, Gantt charts or event logs, *Aoyama et al. (2016)*, *Kratsch et al. (2022)*, *Umemoto et al. (2024)*, *Tanaka et al. (2025)*, *Wördehoff et al. (2025)*. Video has thereby become an important source of process observations in shipyards. Each system, however, delivers the indicators or records decided at design time, whereas the questions raised on site change with the production situation. How continuously recorded video can answer such varying questions is therefore a further problem for monitoring.

With the development of large language models (LLMs) and vision-language models (VLMs), recent studies have used LLM agents to understand and query long videos, *Fan et al. (2024)*, *Zhang et al. (2025)*, *Shen et al. (2025)*. A video agent of this kind converts a recording offline into a memory of clips, captions, objects and frames. From that memory, a language model retrieves what a question requires and forms an answer over several steps of reasoning, so that the analysis a video supports is stated in language at query time. The authors' earlier work brought this approach to a shipyard site, *Gui and Aoyama (2026)*, with a detector trained on shipyard footage, a memory holding both captions of the recording and a spatio-temporal graph of the entities observed, and a qualitative evaluation that reconstructed worker trajectories.

Bringing such an agent into shipyard monitoring raises the question of how the process information a shipyard site cares about, which worker performed which action at which time and place, *Aoyama et al. (2016)*, can be stored in a structured memory that is easy to retrieve. A memory built for this setting has to carry observations of these elements, each relating one worker, one activity and one work location over a time interval. Since generic visual categories do not express the workers, activities and plates of

a shipyard work area, such observations call for perception configured for that area, which reads the identity, location and state of every visible worker as the work proceeds. Moreover, for what an LLM agent reports to be trusted in production use, the process information retrieved from the memory has to remain traceable to the original visual evidence.

To bring LLM-based video understanding developed for general video to the production monitoring of a shipyard, and to explore a form of site information acquisition for shipyard digital twins, this study proposes a grounded process memory, which organises the fixed-camera video of a shipyard into a memory of observations at several granularities. At the finest granularity the memory keeps, at a fixed time step, one observation per visible worker, recording the activity, the work location and the frame in which they were read. Above it, the recording is divided into segments according to how these observations change over time, and each segment gathers the observations of its span together with the keyframes that represent it and a caption, carrying the context of continuous work. A question that concerns a longer stretch, up to the whole recording, is answered at query time by combining the two kinds of record. A record at either level keeps its own time interval and source frames, so that retrieved process information leads back to the original visual evidence, and an LLM agent reads these records through complementary access paths, each returning what it finds together with its interval and the frames behind it.

The method of this study has three stages, Fig.1. Shipyard-specific perception (b) identifies in the fixed-camera video (a) each visible worker together with the plate the worker stands on and the activity under way, second by second. Memory construction (c) organises these observations into the grounded process memory, in which segments of the recording are linked to the workers, plates and frames behind them. Retrieval (d) is carried out by an LLM agent, which reads the memory as the question requires and returns the answer together with its evidence interval and the frames behind it (e).

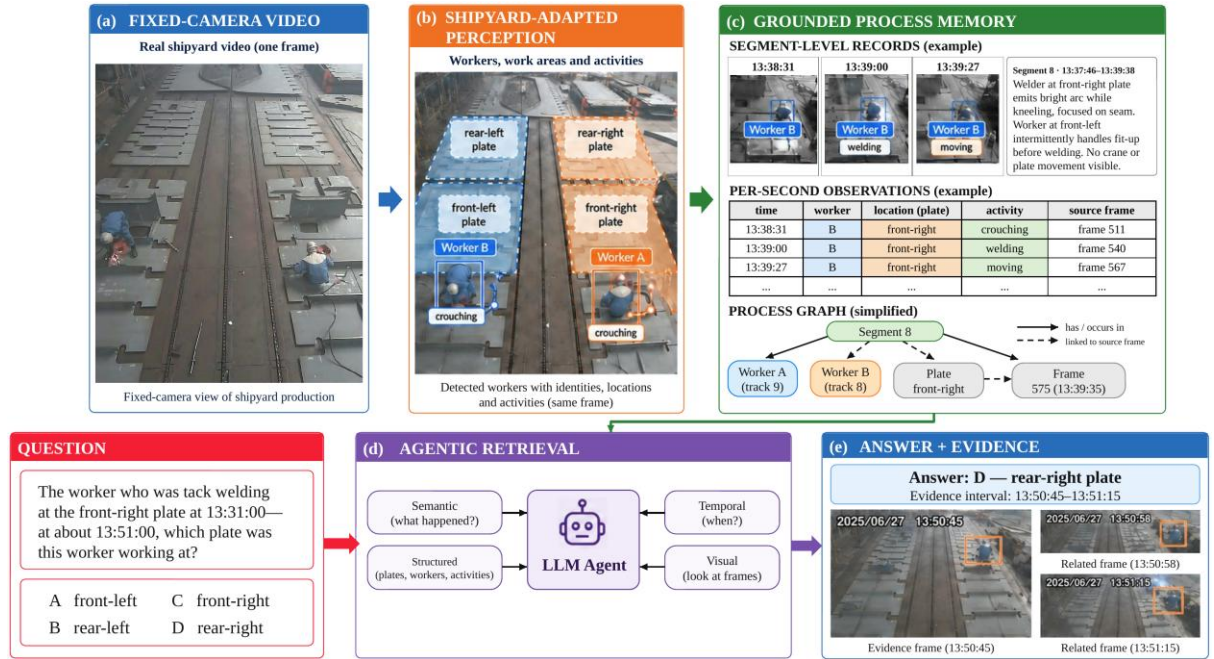


Fig.1: Overview of method. Fixed-camera video (a) passes through perception (b), memory construction (c), retrieval (d), and a question is answered with its evidence interval and frames (e).

To examine the effectiveness of the proposed memory, this study built a small test set from a ground-truth annotation of video of a subassembly work area and compared the proposed system with a VideoAgent-style baseline. The two systems share the same per-frame detections, the same language and vision models and the same reason-and-act executor, while they differ in how workers are identified, in how the recording is segmented, captioned and sampled for visual inspection, and in the memory and

the tools that read it. In this small-scale experiment, the proposed system answered more process questions correctly and returned temporally aligned evidence more often than the VideoAgent-style baseline. The main contributions of this study are as follows.

- Explores how a vision pipeline perceives site information, and proposes a VLM-based tracklet fusion method for the difficulty of tracking worker identities continuously in factory surveillance video.
- Organises these observations into a grounded process memory whose per-second and segment-level records keep their time intervals and source frames, and opens it through complementary access paths by which an LLM agent retrieves what a question requires.
- Evaluates the proposed system as a whole through grounded process question answering on one annotated hour of shipyard video, against a VideoAgent-style baseline on the same questions.

2. Background and Related Work

2.1. Monitoring of Shipyard Processes

Production management in a shipyard needs to know the progress of assembly and the work under way on the floor, and for a long time this process information rested on the eye of experienced workers and on production data reported by the workers themselves, which made it hard to obtain in real time and inaccurate, *Kim et al. (2015)*, *Choi and Seo (2020)*. To obtain the same information by measurement, cameras and sensors have been brought to the work area in a series of studies. *Kim et al. (2015)* estimated the completion of block assembly from ceiling-mounted cameras against reference images projected from the CAD model; *Aoyama et al. (2016)* defined process information as which worker performed which action at which time and place, obtained block and worker action labels by image processing, and drew Gantt charts of the work linked to the footage; *Choi and Seo (2020)* combined markers on mounted components with the time series of welding machines to measure the physical progress of hull block assembly in real time. Camera data have since been carried into the records of process management and towards finer objects on site. *Kratsch et al. (2022)* converted camera data into an event log over activity classes defined in advance, for use in process mining; *Umemoto et al. (2024)* detected blocks in a fabrication shop, transformed their positions into shop coordinates and recorded when each block entered and left; *Tanaka et al. (2025)* summed detections of welding arcs and of standing or sitting workers within an analysis region of network-camera footage into a work index over time; and *Wördehoff et al. (2025)* identified the process steps in video without supervision, so that the activities of the log need not be defined beforehand.

These studies show that video can already supply process information at several levels, from the completion, position and timing of blocks to area-level work indices, worker-level work charts and activity-level event logs. Their limits are equally clear. First, what a system delivers is settled at design time by the classes it detects and by the way it aggregates them, so when the work or the concern on site changes with the production situation a new detection and processing pipeline is needed, and the dynamics and flexibility of the site are hard to follow. Second, some perception tasks on site remain difficult, since workers in identical clothing are hard to tell apart, which left *Umemoto et al. (2024)* unable to collect information on workers, and position data can be missing or mistaken while some work cannot be assigned to a block, *Aoyama et al. (2016)*. Third, what is delivered is mostly an aggregated indicator or record, and only a few works, such as the video-linked Gantt chart of *Aoyama et al. (2016)*, keep the link between a record and the footage it rests on, so a user can rarely return to the observation to check a record. Monitoring therefore has to support questions that change with the production situation and to return process information that leads back to the observation behind it.

2.2. LLM-based Video Understanding and Video Agents

Large language models (LLMs) understand a question stated in natural language, plan the steps needed to solve it, and alternate between reasoning and calls to external tools until an answer is reached, *Yao*

et al. (2023). This ability makes them promising for questions that were not fixed in advance, since the analysis need not be written as a fixed pipeline at design time but can be organised by the model when the question is asked. With the development of LLMs, several studies have used LLM agents to understand long videos. Hours of footage cannot be passed to a model at once, so these methods first convert the recording offline into a memory that can be queried and then let the model search it over several steps once a question is asked, as in VideoAgent, *Fan et al. (2024)*, Deep Video Discovery, *Zhang et al. (2025)*, and Vgent, *Shen et al. (2025)*. Deep Video Discovery holds its memory at three granularities, a global registry of subjects, captioned clips with their embeddings and frames indexed to their clips, and Vgent connects clips into a graph through the entities they share and retrieves clips through that graph.

VideoAgent serves as the baseline of this study, so its architecture and workflow are described here in more detail. Its memory has two parts. The temporal memory follows the time axis, in which the recording is cut into uniform segments of 2 s, each segment is captioned by a video captioning model, and the text embedding of the caption is stored together with the visual feature of the segment. The object memory follows the objects, which are detected, tracked and re-identified across segments, and each object is recorded as one row holding its identifier, its category and the indices of the segments in which it appears. The agent itself is an LLM with tool use, and four tools are available to it. Caption retrieval returns the captions of a range of segments, segment localisation finds the segments relevant to a query by an ensemble of the similarity between the query and the caption-text embeddings and between the query and the visual features, visual question answering puts a question about one segment to a VLM, and object memory querying lets the LLM write SQL against the object table.

The workflow has an offline and an online step. Offline, the segmentation, the captioning, the embeddings and the tracking with re-identification are computed once and written into the two memories. When a question arrives, the LLM reads it together with the descriptions of the tools and alternates between reasoning and tool calls, for example localising the relevant segments first, then retrieving their captions, checking the frames with visual question answering where needed, or querying the object table for a question about objects, and finally forms the answer from what the steps returned. The method outputs the answer itself, while the segments and frames on which the answer rests remain inside the reasoning process.

The authors' earlier work brought this line to shipyard monitoring, *Gui and Aoyama (2026)*, with a detector fine-tuned on shipyard footage, change-point segmentation in place of segments of equal length, and a spatio-temporal graph of the observed entities and their who-where-when relations, and evaluated it qualitatively by reconstructing worker trajectories. The trustworthiness of its answers nevertheless remained open, since the evaluation showed only that the agent could reconstruct plausible trajectories and did not test whether the answers rested on the correct times and frames. How the process information an agent returns can correspond to the workers, activities, work locations and time intervals on site and be traced to the frames and times it came from is the problem this study addresses.

3. Grounded Process Memory from Shipyard Video

3.1. Framework and Information Flow

The method proposed in this study turns fixed-camera video of a shipyard work area into a grounded process memory, which an LLM agent then reads to answer questions about the production process (Fig. 1). Vision-based perception first obtains the process observations of the site, that is, which worker showed which activity at which work location at which instant or over which interval. These observations are then organised into a memory at several granularities. The per-second observations keep the state of every instant, while segments cut where the work changes gather a stretch of continuous work with a caption. Records at both levels are stored in one process graph together with their time and source frames. Finally the LLM agent retrieves this memory as a question requires and returns the records it finds together with their time intervals.

Information flows through three stages in one direction. Perception takes the frames as input, detects the workers and the steel plates frame by frame, and joins the detections of one worker into an identity along the time axis. It then assigns to every worker at every instant the plate and the work state, so that it outputs per-second observations. Memory construction takes these observations as input and cuts the recording into segments where the work state changes. It selects representative keyframes for each segment, writes a caption anchored on its workers and plates, and connects segments, workers, plates and frames as the nodes of a process graph. Retrieval takes the question as input, and the agent reads the graph through several access paths, each corresponding to one way a question enters the memory. The following three subsections describe perception, memory construction and retrieval in turn.

3.2. Process Perception for the Shipyard Site

Perception converts the video into process observations that identify the visible worker, activity, work location, time and source frame. Generic object detectors do not provide the categories needed to describe work in a shipyard, so the detector is configured for the vocabulary of the observed work area. We use YOLO26x, *Jocher et al. (2026)*, fine-tuned on annotated shipyard images, to detect standing, crouching and welding workers together with steel plates and cable reels. These detections provide the common visual basis for worker tracking, activity recognition and location assignment.

Individual detections describe only single frames, whereas process observations require worker identity to persist over time. Hybrid-SORT, *Yang et al. (2024)*, associates worker detections into tracklets by combining motion with other detection cues, without requiring a separately trained appearance model. This is suitable for the site because the workers wear similar overalls and helmets, which limits appearance-based re-identification. Each tracklet therefore provides a locally continuous identity, but occlusion, missed detections and workers leaving and re-entering the field of view may divide one person into several tracklets.

To recover identity across these gaps, this study uses a VLM-assisted tracklet fusion method. Candidate pairs are first restricted to non-overlapping tracklets whose endpoints can be connected under feasible walking motion. Unambiguous candidates are resolved by two deterministic rules: one joins tracklets separated by a short gap with negligible displacement, and the other follows continuity at the same work location when no competing worker appears there. The remaining candidates are assessed by a VLM using annotated frames around the gap, showing the candidate workers and their work locations. Candidates with insufficient evidence remain separate, and identity consistency is maintained as accepted merges are applied. This design assigns routine cases to explicit physical constraints and uses the VLM only where visual judgement is needed.

After fusion, observations belonging to the same worker share one identity across their tracklets. At each visible instant, the worker position is mapped to a work location registered in the camera view, while the detector supplies the observed activity. Welding is recorded only when the arc is visible, so the activity label represents visual evidence at that instant rather than an inferred continuation of the task. The resulting per-second observation contains the worker identity, activity, work location, time and source frame. These observations form the input to the grounded process memory constructed in the next section.

3.3. Construction of Grounded Process Memory

Memory construction organises the per-second observations into a memory that can be retrieved in language and led back to the picture, and it proceeds in three steps. The recording is cut into segments by change-point detection, a caption anchored on the workers and the plates is written for each segment, and the segments, workers, plates and frames are then connected into a process graph, Fig.2.

Per-second observations record the site one instant after another, while a question about continuous work concerns a stretch of it, and describing a recording in language also needs a unit of time, so the memory needs segments. A general video agent cuts segments uniformly at a fixed length, which does

not suit a shipyard work area, since a bout of work is not of a fixed duration. Segments of equal length either cut one bout into several pieces or mix successive work states within one segment, and a description written for such a segment cannot say what happened in it. The system therefore cuts segments by change-point detection (CPD), which works on per-second signals derived from the tracked workers. These signals include how often welding and crouching are detected, how much of the work area the workers occupy and how far they move. They stay steady while the work state does not change and jump where it changes, and the boundaries are placed at these jumps, Fig.2(a). Each stretch in which no work is visible becomes one segment, so that a pause is not cut into pieces. Within a working stretch the boundaries are placed by the PELT algorithm, *Killick et al. (2012)*, under a composite cost that combines counts with continuous signals, *Truong et al. (2020)*. A segment obtained in this way is a stretch over which the work state stays steady, so it may cover more than one activity and a boundary may fall inside a bout of work that continues into the next segment.

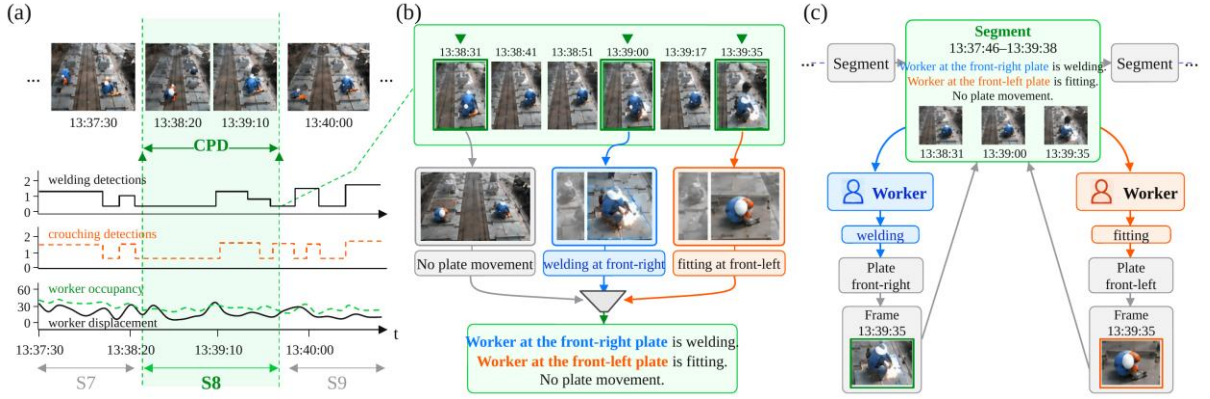


Fig.2: Memory construction for one segment. (a) Boundaries at jumps of the signals. (b) AKS keyframes and per-worker drafts merged into an object-centric caption. (c) The segment in the graph.

A segment is defined by signals and cannot yet be reached by a question posed in natural language. Since language is often the first step by which the agent enters the memory, every segment needs a description in words. Asking the VLM to describe the whole stretch of video is not enough, since a caption of that kind states the general impression of the scene, in which several workers dressed alike are hard to tell apart. Who works at which plate is then often left unsaid, and the description may fail to correspond to the tracked objects, so that it cannot be led back to a particular worker or picture. CAT-V, *Tang et al. (2025b)*, segments the objects of a video one by one and generates a description of each object over a temporal unit. This study takes that idea to the workers and plates perception has already identified and writes an object-centric caption for each segment. A scene description is first obtained from the keyframes of the segment. For each worker visible in the segment, the box of that worker and the label of the plate they stand at are then drawn on the same keyframes, and the VLM is asked to describe what this worker is doing. The drafts for the workers and the scene description are finally merged into one segment-level caption, in which each worker takes one sentence saying who stood at which plate and did what, Fig.2(b). Every sentence of the caption therefore hangs on a worker and a plate that perception has identified. A semantic match on the caption thus lands on a named worker at a named work location, and it leads back along the worker and the plate to the picture.

The caption is written from the keyframes of a segment rather than from all of its frames, since a segment holds far more frames than the VLM can look at in one call. Uniform sampling under a fixed camera returns almost identical pictures, while the arc of tack welding appears in the image only for very short moments, which uniform sampling easily misses although they say most about the work. The system therefore uses adaptive keyframe sampling (AKS), *Tang et al. (2025a)*, which scores the frames of a segment against a query describing the work of the site and keeps the few frames that are related to the work and differ from one another. The keyframes are used for writing the caption and are kept in the segment record as well, and the visual inspection of Section 3.4 uses the same sampling method again.

The segment-level captions and the per-second observations are finally placed in one process graph, Fig. 2(c). The graph holds four kinds of node, a segment with its time interval, its caption and its keyframes, a worker as an actor, a plate as a work location and a frame as one second of the recording. The edges are the observations themselves, and a per-second observation connects one worker to one frame, carrying the plate that worker stands at in that second and the work state recognised there. A segment record connects the segment to the frames of its interval and to its keyframes, and also to the workers active in it with the plates they visited and the sentence of the caption belonging to each, while neighbouring segments are connected to each other in time. The segment node also holds a text embedding of its caption and a visual embedding of its keyframes, so that a description given in language is scored against both to find the corresponding segment. Because the records of both levels belong to the same graph and each states the frames it was derived from, a query reads the seconds behind a segment record or the segment a given second belongs to. Any retrieved record therefore leads back to its time interval and to the picture behind it.

3.4. Retrieval and Agentic Use

The memory is opened to the LLM agent through four access paths, and the agent chooses among them according to the question within a reason-and-act loop. The process information each path returns carries its time interval and its frames. The four paths correspond to four ways in which a question enters the memory, that is, describing a situation without naming a time, giving a condition on the recorded fields, tracing forwards or backwards from a given instant, and needing the picture itself to settle the question.

Semantic retrieval maps a natural-language description to segment nodes through their caption and visual embeddings. Each result contains the segment caption and links to the associated workers, plates, keyframes and neighbouring segments, allowing the agent to move from a verbal description to the relevant process record.

Structured graph querying filters the per-second observations by time, worker, plate or work state, and groups the matches when needed. It reports the duration and temporal extent of each group, supporting existence, duration and count questions directly from the recorded observations.

Temporal tracing follows an anchored segment or worker observation forward or backward through neighbouring records. It highlights changes in work location or work state and the appearance or disappearance of workers, providing the local process context needed for order and transition questions.

Visual inspection returns to the source video when the stored observations are incomplete or ambiguous. The agent specifies a segment, time range, instant or set of frames; the system then selects suitable frames and marks the tracked workers and plates. The VLM description and answer remain attached to the frames and timestamps from which they were obtained.

The four paths are driven by a reason-and-act (ReAct) loop, in which the agent chooses the next path from what the previous steps returned. The identifiers of segments, workers, plates and frames that one path returns are the inputs the other paths accept, so the agent moves from a description to the segment behind it and from the seconds a worker spent at a plate to the frames that record them. The output of each path closes with a line of facts about its evidence, the seconds it covers and the gaps in them, the continuity of the tracks involved, and whether the caption and the tracking agree about where a worker was. The state of the evidence is present in this way at every step of the reasoning, and the final answer is returned together with its time interval and its frames.

4. Evaluation through Grounded Process Question Answering

4.1. Question Set

To examine whether the proposed method answers questions about the process correctly, we prepared a small question set from a ground-truth annotation of shop-floor surveillance video. The recording is one hour of fixed-camera video from one stage of a subassembly line, where two workers work on four plates at the positioning of members, chamfering, tack welding and auxiliary tasks. A domain expert watched the recording repeatedly and annotated the timeline of the two workers who appear in it, with the start and end time of every activity of each worker and the work it covered.

From this ground truth we prepared 40 answerable four-option questions and 5 unanswerable ones, Table I. The questions cover time, location, count, activity, order of work and the existence of an entity, the kinds of information production management regularly needs. The three distractors of a question follow the three strategies of STAR, *Wu et al. (2021)*, compositional, frequent and random, and every value they take occurred somewhere in this hour. Each question carries a fifth option E, “Cannot be determined from the recording”, which is the answer to the five unanswerable ones. Before any system saw the set, we checked that each of the 45 questions has exactly one true option on the timeline, and that the wording of a question and its options did not stably give away the answer to the language model under test when it read the set without the video. Each instance also records the interval of the timeline that supports its answer, against which we judge whether a system reports the evidence its answer rests on.

Table I: Question set: types, examples and counts

Type	Example	<i>n</i>
Time	When did tack welding first start at the rear-right plate?	7
Location	At about 13:51, at which plate was the worker who had been tack welding at the front-right plate at 13:31?	7
Count	At 14:00, how many workers were tack welding?	6
Activity	What was the worker at the front-right plate doing at about 13:34?	7
Order	What did the worker at the front-right plate do right after the tack-welding bout that ended at 13:44?	7
Existence	Was anyone tack welding at the front-left plate between 13:38 and 13:42?	6
Unanswerable	At 13:20, how many workers were tack welding?	5
All answerable		4
		0

4.2. Compared Systems and Measures

We put the recording of Section 4.1 to two systems and compare what each of them returns. The VideoAgent-style baseline against which we examine the proposed method inherits the memory and the tools of the official VideoAgent described in Section 2.2, its temporal and object memories, its four tools including the nested agent that writes SQL over the object memory, and its prompt with the adaptations that the question format requires. For computational efficiency, the baseline divides the recording into equal-length segments of 120 frames.

The proposed system is the method of Section 3 in full, so the comparison is between two complete configurations. They differ in the five design dimensions of Table II, the identification of workers, the captioning, the key frames, the segments and the memory, and with the memory also in the tools that reach it and in the prompt that describes them. Everything that does not follow from these choices is

shared. Both systems read the per-frame detections of the same fine-tuned detector, run on the same ReAct executor with a budget of 15 steps, and call the same locally served Qwen3.8-27B language model as the reasoning engine and Qwen3-VL-8B-Instruct as the vision model at the end of the paths.

Table II: Design dimensions on which the two systems differ; everything else is shared (Section 4.2).

	Proposed	VideoAgent-style
Worker identification	Hybrid-SORT tracking with VLM-assisted tracklet fusion	ByteTrack tracking with re-identification by image embeddings from CLIP-B/32 and DINOv2-g/14
Captioning	object-centric captions anchored on the tracked workers and plates	one plain scene caption per segment
Key frames	adaptive keyframe sampling	uniform sampling
Episodes (segments)	change-point segmentation	equal-length segments of 120 frames
Memory	grounded process memory, a process graph of segments, workers, plates and frames	temporal memory and object memory, as in the official VideoAgent

The measures follow the two things an answer has to get right, the option it selects and the evidence it reports. Accuracy (Acc) is the share of the 40 answerable questions whose option letter is correct. The intersection over prediction (IoP) of an answer is the share of its reported interval P that lies inside the ground-truth interval G , $|P \cap G| / |P|$, and mIoP is its mean over the 40 questions. The temporal intersection over union (IoU) is the share of the union of the two intervals that both of them cover, $|P \cap G| / |P \cup G|$, and mIoU is its mean. Unlike IoP, IoU falls when the reported interval runs past the event, and it falls as well when the interval is narrower than the event. The grounded accuracy (Acc@GQA) of NExT-GQA, *Xiao et al. (2024)*, counts an answer only when it is correct and its IoP reaches 0.5, and it therefore measures temporal evidence alignment, so that a correct answer whose interval lies away from the annotated evidence, the evidence misalignment reported by *Qiu et al. (2026)* (Section 2.2), scores nothing. The abstention rate (Abst) is the share of the five unanswerable questions answered E, and the false-abstention rate (FA) is the share of the 40 answerable ones answered E. The token cost is the mean over the question set of the prompt and completion tokens the two models spend on one question, reported separately for the language model and the vision model. An answer that reports no interval scores 0 on IoP and IoU, and an answer naming one instant is scored on the one-second interval of that instant. These measures bound the temporal side of grounding alone. The spatial side enters through the Location questions, whose answers name a plate, and no separate spatial measure is defined here.

4.3. Results

In the small-scale experiment, the proposed system answered 22 of the 40 answerable questions correctly, compared with 13 for the VideoAgent-style baseline. On Acc@GQA, 17 answers met the criterion for the proposed system and 3 for the baseline. The mIoP was 0.802 and 0.411, respectively, while the mIoU was 0.367 and 0.353 (Table III). The proposed configuration therefore achieved higher answer accuracy and Acc@GQA than the VideoAgent-style baseline.

In this run the proposed system answered more questions correctly in four of the six types, Fig.3. Location is the strongest of them, with all seven questions answered correctly against five for the baseline, and Time, Activity and Existence go the same way. Order is level at 2 of 7 for both systems, and Count runs the other way, 1 of 6 against 2 of 6. Acc@GQA shows the same pattern across the six types, and on Count it stands at zero for both systems, neither of them reaching the IoP threshold there.

Both systems abstain on 2 of the 5 unanswerable questions, and on the answerable ones the proposed system abstains falsely once against three times for the baseline. The nine extra correct answers cost

about 6 % more tokens per question, 33.8k against 31.8k, and those tokens are spent on a different model. The proposed system puts 23.8k of them into the language model, because the rows the memory returns enter every prompt that follows. The baseline puts 22.8k into the vision model instead, since it calls that model more than twice as often, 2.31 calls per question against 0.98.

Table III: Results on the question set, one run per system, measures as in Section 4.2. Shares are of the 40 answerable questions, Abst of the 5 unanswerable.

	Proposed	VideoAgent-style
Acc, correct of 40	22 (55.0 %)	13 (32.5 %)
Acc@GQA (correct and IoP $\geq$ 0.5)	17 (42.5 %)	3 (7.5 %)
mIoP	0.802	0.411
mIoU	0.367	0.353
Abst, of 5 / FA, of 40	2 / 1	2 / 3
Tokens per question, LLM + VLM (k)	33.8 = 23.8 + 10.0	31.8 = 9.0 + 22.8

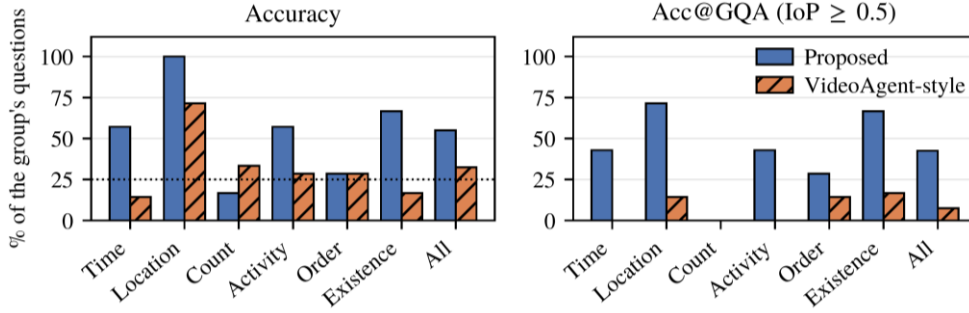


Fig.3: Accuracy (left) and Acc@GQA (right) by question type, one run per system. Four types favour the proposed system, Order is level and Count runs the other way; the dotted line is 25 % guessing.

4.4. Discussion

This evaluation shows that the method as a whole, which combines site perception, memory construction and retrieval, raises in this case the accuracy of process question answering and the share of answers that are correct and meet the requirement on temporal evidence at the same time, at a similar token cost. The improvement on the Location, Time, Activity and Existence questions reflects the value of organising video into queryable process records. The per-second observations keep the location and the visual state of a worker at a specific instant, and the segment-level captions provide the context of a stretch of work, so that the agent can combine information at different granularities to answer a question and return the relevant frames through the source links kept in the records. The structured query was also the retrieval path used most often in this run. For shipyard monitoring, this organisation of the memory supports querying the same recording around different production questions and gives the returned process information a traceable observational basis.

The analysis of the wrong answers further shows that the acquisition of process information is still limited by what the observations contain and by the granularity at which activities are expressed. Of the 18 answerable questions answered wrongly, we associated 12 with the sparse welding record or with the coarse granularity of the action labels. The current welding label denotes a welding arc visible in the frame, whereas tack welding in the questions denotes work that continues over a period, and scattered arc records hardly allow the start, the end and the duration of the whole piece of work to be determined directly. At the same time, the three labels standing, crouching and welding can hardly tell apart specific activities such as fitting members, checking angles and tidying up. A Count question

requires judging how many workers are performing a given task at one instant, and an Order question requires recognising the transition between adjacent activities, so these two types depend especially on an accurate expression of the working state. These errors suggest that the link between the per-second visual state and the semantics of continuous work is the part of the current process memory that still needs to be improved.

Visual inspection gives the agent a way back to the original frames to check the records, but its effect is in turn limited by the interpretive ability of the vision model. In the question-by-question analysis, four wrong answers were related to the reading of the vision model, and on one of them a repeated check produced contradictory judgements. On three wrongly answered Order questions the reported interval lay entirely inside the annotated event, yet the interpretation of the work content was still wrong. The abstention results show that the agent’s judgement of whether the evidence suffices to support an answer is still inadequate. Both systems abstained on only two of the five unanswerable questions, and the remaining answers of the proposed system involved reading a missing record as the absence of an event, accepting an unverified presupposition in the question, and inferring a worker’s identity across a tracking gap. Keeping the source links provides the means to check these answers, while how to use the gaps and inconsistencies in the records to judge whether the evidence is sufficient remains an important factor in the reliability of the answers.

This comparison rests on one hour of video from one work area, with one run of each system, and the results reflect the performance of the two complete configurations in this case. Worker identification, captioning, segmentation and the retrieval paths together make up the difference between the systems, and the 120-frame segments adopted by the baseline also affected the evaluation of temporal evidence. After the prompt-development questions are removed, the results keep the same direction of improvement. The minute-level response time of the current question answering is better suited to retrospective queries over a recorded process. For a future shipyard digital twin, this study provides a way of organising and querying the process observations of the site, and how it connects to the updating of the twin’s state remains to be studied in specific applications.

5. Conclusion

To support the acquisition of process information for shipyard digital twins, this study proposed a grounded process memory for fixed-camera surveillance video. Shipyard-specific perception, including VLM-assisted tracklet fusion, records each worker’s location and visual state over time. Change-point segmentation and object-centric captioning organise these observations into a graph linking per-second records with segment-level descriptions and source frames. An LLM agent uses complementary retrieval tools to answer process questions and identify the corresponding visual evidence.

In the small-scale experiment, the proposed system answered 55.0 % of the 40 answerable questions correctly, compared with 32.5 % for the VideoAgent-style baseline. Acc@GQA, which requires both a correct answer and an evidence interval with IoP of at least 0.5, increased from 7.5 % to 42.5 %, with approximately 6 % more tokens per question. These results support the proposed approach to organising and retrieving process information from shipyard video.

Grounded process question answering serves both as an evaluation task and as an interface for querying recorded work. The remaining errors indicate a need to relate sparse visual observations more closely to continuous activities and to improve the agent’s judgement of evidence sufficiency. By retaining the relationships between workers, activities, work locations and their visual sources, the memory provides a structured representation of site observations. Future work can examine how these observations support the updating of digital-twin process states.

Acknowledgements

The authors thank Japan Marine United Corporation for the funding and for the recording used in this study.

References

- AOYAMA, K.; LIU, J.; HIRO, Y.; MIZUSHIMA, T.; OIZUMI, K. (2016), *Development of Construction Management System based on Cyber-Physical System of Shipbuilding, 1st Report: Development of Construction Process Monitoring and Visualizing System using Video Image Processing Technology*, JSME Manufacturing Systems Division Conf., in Japanese
- CHOI, T.; SEO, Y. (2020), *A Real-Time Physical Progress Measurement Method for Schedule Performance Control Using Vision, an AR Marker and Machine Learning in a Ship Block Assembly Process*, *Sensors* 20/18, 5386
- FAN, Y.; MA, X.; WU, R.; DU, Y.; LI, J.; GAO, Z.; LI, Q. (2024), *VideoAgent: A Memory-augmented Multimodal Agent for Video Understanding*, *Computer Vision – ECCV 2024, Lecture Notes in Computer Science* 15080, Springer, pp.75–92
- GUI, C.; AOYAMA, K. (2026), *Toward Shipyard-Specific Monitoring Video Understanding with Memory-Augmented LLM Agent*, *J. Japan Society of Naval Architects and Ocean Engineers*
- JOCHER, G.; QIU, J.; LIU, M.; LYU, S.; AKYON, F.C.; KALFAOGLU, M.E. (2026), *Ultralytics YOLO26: Unified Real-Time End-to-End Vision Models*, arXiv:2606.03748.
- KILLICK, R.; FEARNHEAD, P.; ECKLEY, I.A. (2012), *Optimal Detection of Changepoints with a Linear Computational Cost*, *J. American Statistical Association* 107/500, pp.1590–1598
- KIM, M.; CHOI, W.; KIM, B.-C.; KIM, H.; SEOL, J.H.; WOO, J.; KO, K.H. (2015), *A Vision-Based System for Monitoring Block Assembly in Shipbuilding*, *Computer-Aided Design* 59, pp.98–108
- KRATSCH, W.; KÖNIG, F.; RÖGLINGER, M. (2022), *Shedding Light on Blind Spots: Developing a Reference Architecture to Leverage Video Data for Process Mining*, *Decision Support Systems* 158, 113794
- QIU, C.; ZHANG, Y.; LUO, X.; SONG, S.; LIU, X. (2026), *VideoSEAL: Mitigating Evidence Misalignment in Agentic Long Video Understanding by Decoupling Answer Authority*, *Int. Conf. Machine Learning (ICML)*
- SHEN, X.; ZHANG, W.; CHEN, J.; ELHOSEINY, M. (2025), *Vgent: Graph-based Retrieval-Reasoning-Augmented Generation for Long Video Understanding*, *Advances in Neural Information Processing Systems* 38 (NeurIPS 2025).
- TANAKA, T.; TANAKA, Y.; SHINODA, T. (2025), *A Study on the Extraction of Construction Information for Productivity Evaluation through Work Monitoring*, *Conf. Japan Society of Naval Architects and Ocean Engineers* 40, paper 2025S-GS13-2, pp.715–717, in Japanese
- TANG, X.; QIU, J.; XIE, L.; TIAN, Y.; JIAO, J.; YE, Q. (2025a), *Adaptive Keyframe Sampling for Long Video Understanding*, *IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR)*, pp.29118–29128
- TANG, Y.; BI, J.; HUANG, C.; LIANG, S.; SHIMADA, D.; HUA, H.; XIAO, Y.; SONG, Y.; LIU, P.; FENG, M.; GUO, J.; LIU, Z.; SONG, L.; VOSOUGHI, A.; HE, J.; HE, L.; ZHANG, Z.; LUO, J.; XU, C. (2025b), *Caption Anything in Video: Fine-grained Object-centric Captioning via Spatiotemporal Multimodal Prompting*, arXiv:2504.05541.
- TAO, F.; ZHANG, M. (2017), *Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing*, *IEEE Access* 5, pp.20418–20427

- TRUONG, C.; OUDRE, L.; VAYATIS, N. (2020), *Selective Review of Offline Change Point Detection Methods*, Signal Processing 167, 107299
- UMEMOTO, D.; IWASHITA, T.; SAKAI, T.; NAKAO, S.; NISHI, Y. (2024), *Research on Monitoring Methods for 4M Information in Manufacturing Sites Using Object Detection Technology*, Conf. Japan Society of Naval Architects and Ocean Engineers 39, paper 2024A-GS15-4, pp.1081–1086, in Japanese
- WÖRDEHOFF, N.; EGGER, A.; KRATSCH, W.; KÖNIG, F.; RÖGLINGER, M. (2025), *Beyond Assumptions: A Reference Architecture to Enable Unsupervised Process Discovery from Video Data*, Decision Support Systems 199, 114544
- WU, B.; YU, S.; CHEN, Z.; TENENBAUM, J.B.; GAN, C. (2021), *STAR: A Benchmark for Situated Reasoning in Real-World Videos*, Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS 2021)
- XIAO, J.; YAO, A.; LI, Y.; CHUA, T.-S. (2024), *Can I Trust Your Answer? Visually Grounded Video Question Answering*, IEEE/CVF Conf. Computer Vision and Pattern Recognition (CVPR), pp.13204–13214
- YANG, M.; HAN, G.; YAN, B.; ZHANG, W.; QI, J.; LU, H.; WANG, D. (2024), *Hybrid-SORT: Weak Cues Matter for Online Multi-Object Tracking*, AAAI Conf. Artificial Intelligence 38/7, pp.6504–6512
- YAO, S.; ZHAO, J.; YU, D.; DU, N.; SHAFRAN, I.; NARASIMHAN, K.; CAO, Y. (2023), *ReAct: Synergizing Reasoning and Acting in Language Models*, Int. Conf. Learning Representations (ICLR)
- ZHANG, X.; JIA, Z.; GUO, Z.; LI, J.; LI, B.; LI, H.; LU, Y. (2025), *Deep Video Discovery: Agentic Search with Tool Use for Long-form Video Understanding*, Advances in Neural Information Processing Systems 38 (NeurIPS 2025)

Implementing CAD Data Standards in Application Software: Challenges and Solutions

Herbert Koelman, SARC, Bussum/Netherlands, H.J.Koelman@sarc.nl

Dennis Korpel, SARC, Bussum/Netherlands, Dennis@sarc.nl

Abstract

Despite ongoing progress, meaningful data exchange between software applications remains challenging. Although many data standards exist, this paper focuses on how application software supports them. Data exchange requires translating between standardized formats and internal data structures, typically through files, APIs, or similar interfaces. While conceptually simple, developing and maintaining these interfaces is often time-consuming, tedious, and error-prone. To address this, we developed a library that generates serialization and deserialization code from internal structures, supporting formats such as XML and JSON. This paper aims to show that interface implementation is manageable, though labor-intensive, and to share our approach and lessons with fellow software developers.

1. Introduction

Although this paper focuses on the practical implementation of data exchange standards in software, it is useful to begin by outlining the broader context. And let us start with economics. A study by *Gallaher et al. (2004)* estimated the annual cost of inadequate interoperability in the U.S. capital facilities industry at US\$ 15.8 billion. With some caution, this can be interpreted as roughly 1.5% of the industry's annual added value. That study is somewhat distant from our own context, both in time and in industrial domain. Closer to home is the conclusion of *Agis and Brett (2024)*, who observe that "a lack of digital architecture for shipbuilding delimits the overall understanding and opportunity search for the full exploitation of digital ecosystems and their virtual interoperability". That said, it is somewhat surprising that literature should have to persuade us of the importance of effective data exchange between the design, engineering, production, and operational phases. Put simply, striving for seamless information exchange across the lifecycle of a vessel is just common sense.

Accordingly, SARC - the developer of the PIAS ship design software and the LOCOPIAS loading and stability software - has devoted considerable effort for decades to technologies and tools for data exchange. While this is often referred to simply as "interfacing", within our domain of ship design and construction the term Product Data Technology (PDT) is more appropriate. At present, SARC is closely involved in two major initiatives in this field. The first is the European Horizon project SEUS, *Gaspar et al. (2023)*, whose primary objective is the integration of CAD, CAE, and PLM systems. The second is a support programme within the Dutch Maritime Master Plan, an ambitious national initiative aimed at the design and construction of several dozen climate-neutral vessels. The digital aspects of this programme are coordinated through the subprogramme Digital Collaboration. Both projects have formulated ambitious objectives, and rightly so. Ultimately, both seek to achieve a fundamental step forward in digital shipbuilding. However, successful implementation ultimately requires us to return to practical reality and address the concrete implementation of data exchange strategies. In other words: programming.

It is this journey - from high-level vision, through architectural design and system engineering, to the mundane practice of software implementation - that forms the flow of this paper. Our first objective is to demonstrate that visionary concepts alone are insufficient; they must ultimately be translated into robust and maintainable software. The second objective is to share a few practical experiences and insights that may prove useful to fellow software developers facing similar interoperability challenges.

2. The status of PDT technology

In discussions about data exchange between computer systems, one frequently encounters confusion over the fact that this issue has not yet been fully resolved. Indeed, some even regard this as a dereliction of duty. This perception is perhaps reinforced by the existence of several so-called “neutral standards”, leading many to assume that these standards are directly and universally applicable. The reality, however, is quite different. Although this paper is not the place to examine the issue in depth - that is covered in slightly more detail in *Koelman et al. (2024)* - its most important aspects will be outlined briefly.

The first aspect is representation variation: the phenomenon that a given object can be described using different mathematical representations. In some cases, these representations are directly convertible on a one-to-one basis. For example, a straight line may be represented either by a point and a direction vector or by two points, with the line implicitly defined as passing through them. In other cases, however, there is no direct one-to-one conversion. Examples include reconstructing a closed, gap-free, geometrically consistent ship hull surface from a point cloud, or transforming a two-dimensional general arrangement plan into a three-dimensional ship arrangement model. The existence of a neutral exchange standard does not remove this problem. For example, although IGES is commonly regarded as a single exchange format, the standard itself defines approximately 150 entity types, many of which have multiple forms, resulting in several hundred distinct entity/form combinations. Consequently, even when two applications both support IGES, successful exchange depends on whether they implement compatible representations for the particular entities involved.

This issue is succinctly captured by *Gielingh (2008)* in the statement that “the neutral model doesn’t really exist”. It is also illustrated in Fig.1, reproduced from *Whitfield (2011)*, which shows that there is no single central representation through which applications can universally understand one another. Instead, interoperability is limited to fragmented overlapping domains, representing those areas in which applications happen to employ the same - or at least mutually convertible - representations.

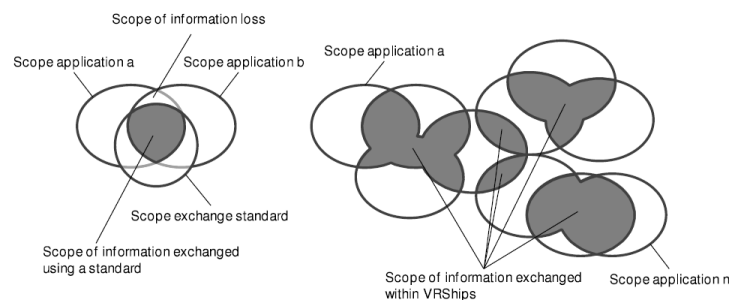


Fig.1: Left the idealized neutral model, right the practice with multiple islands of data sharing

A second, more practical aspect is that conversion between geometric standards does not always work flawlessly. This was already reported by *Gielingh (2008)*, and judging by the findings of *Kurylo et al. (2023)*, the passage of time has not resolved all issues. Based on practical experiments, they concluded that “One of the results of our research and analysis is the need for continuous development of CAD/CAE systems because current neutral formats are not 100% effective”.

A third and final aspect is more fundamental in nature. It is that there are many different categories of information that may need to be exchanged between applications. Later in this paper, for example, IGES and DXF will be discussed. These are two well-known data standards, but they are intended solely for the exchange of graphical information, as reflected in their acronyms: Initial Graphics Exchange Specification and Drawing eXchange Format. In practice, however, there are many other types of information that must be exchanged, including weights and centres of gravity, energy demand, design pressures and temperatures, resistance coefficients of piping components, and coating requirements. And this is only the technical side of the picture, for example financial and logistical must also be taken into account.

The recognition that a product comprises many interrelated aspects, all of which may need to be represented and exchanged, is referred to as multifacetedness. This concept is formalized in an *ISO (2022)* standard, from which Fig.2 is reproduced. The figure distinguishes three complementary aspects: (a) the constructional relationships between components, (b) the functional relationships, and (c) the spatial relationships (e.g., location). Unfortunately, although the standard provides a valuable conceptual framework, it does not offer a directly applicable solution for data exchange.

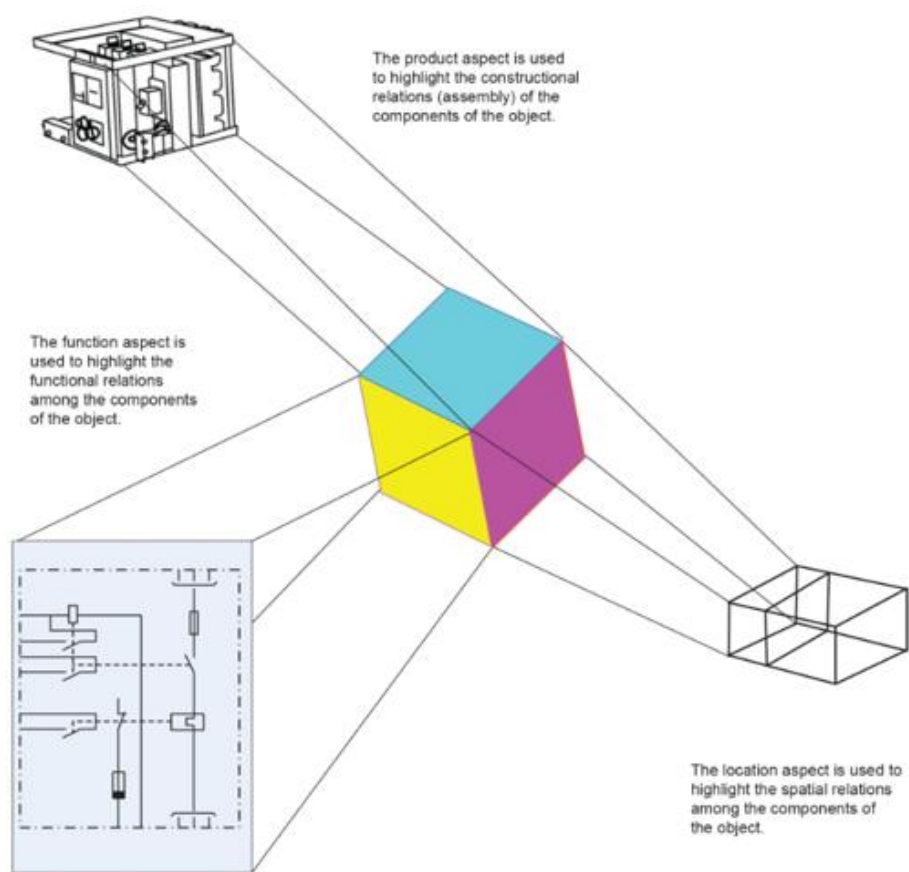


Fig.2: Different aspects (or facets) of an object, from *ISO (2022)*

Finally, a tentative remark on the use of AI as an assistant in data exchange. In this day and age, it is often argued that the long-standing problem of incompatible data standards will eventually disappear because these standards are, in essence, formal languages that can be translated by large language models (LLMs). We cannot predict future developments and therefore prefer to remain cautious. Nevertheless, we believe the following observations are relevant:

- Suppose the objective was to use an LLM to convert a file - or, more generally, a data stream - from one format into another. Where would this AI obtain its training material? Where are the large collections of examples containing identical engineering objects represented in the hundreds of possible combinations of standards, versions, vendor-specific extensions, and modelling conventions?
- Naive desktop experiments with publicly available LLMs demonstrate that automatic format conversion remains far from a solved problem. More advanced approaches have been reported in the literature, such as the work of *Shi et al. (2026)*. Interestingly, however, this approach relies on a specialized representation of STEP rather than the standard format itself. The reason is that the graph structure inherent in conventional STEP files proved too complex for an LLM to process effectively.
- Even if AI eventually becomes capable of reliable format conversion, such translation will most likely remain confined to the same geometric representations - for example, translating

one NURBS representation into another. Truly universal interoperability would also require conversion between fundamentally different representations, such as reconstructing a watertight NURBS model from a point cloud. Or understanding that an unconnected set of surfaces in the real world constitute the boundaries of a watertight compartment. Such tasks require geometric reasoning in addition to language processing. In other words, it calls for Spatial Intelligence, a concept highlighted in the insightful essay of *Li (2025)*.

- Finally, the practical objective is not a standalone tool that can occasionally convert a file. The real requirement is seamless, deterministic, and verifiable data exchange between engineering applications operating within an automated design process. In such a setting, even a small probability of hallucinated or semantically incorrect output is unacceptable.

So far, the picture painted has been rather bleak, and we would gladly leave this entire aspect of PDT aside. However, our software is by no means an isolated system. On the contrary, it invariably forms part of a much broader ecosystem of software applications of many different kinds. Consequently, we have no real choice in the matter: users consistently identify data exchange as one of their primary requirements (where one may, of course, question whether every expressed wish reflects a genuine operational need, but that issue has already been addressed in *Koelman (2023)* and will not be pursued further here). In short, data exchange remains an issue that must be addressed, while keeping a clear focus on the realities of engineering practice. This was also the reason why we were keen to participate in both SEUS and the Maritime Master Plan.

3. Selection and use of standards and their extensions

When selecting data standards, it is useful to distinguish between two categories of ship design data: (a) the hull form itself and (b) everything else.

The hull form is fundamentally about geometry, and geometry cannot be captured by a limited set of parameters that each have an independent physical meaning. Of course, parameters can be derived from a hull form, such as the block coefficient, and the hull may exhibit characteristics that satisfy particular design objectives, such as a required deck area or desired hydrodynamic performance. However, these are not parameters that uniquely determine the final hull shape. At most, they represent design considerations that will have influenced the evolution of the hull form during the design process.

By contrast, many other design entities are defined directly by design parameters which, together with related parameters, fully specify the chosen design. These are often simply numerical values or structured collections of such values. Although they may ultimately give rise to geometry - for example, a container stowage plan based on bays, rows, and tiers, or the shape of a compartment resulting from the intersection of the hull with decks and bulkheads - that geometry is merely the consequence of the design decision rather than the design decision itself. For this reason, these two categories of information are considered separately in the following two subsections.

3.1. Curved Hull shape

The data exchange standards suitable for representing the often doubly curved hull surfaces of ships include DXF/DWG, IGES, ISO 10303/STEP AP216 (Ship Moulded Form), and STEP AP242, in particular the variants that support NURBS curves and surfaces. Based on SARC's practical experience with ship design applications, we have a good understanding of which of these standards are actually used in industry. STEP AP216 is, for all practical purposes, nonexistent, while STEP AP242 is encountered only occasionally, primarily in academic environments. In contrast, IGES, and to a lesser extent DXF, remain the dominant exchange formats. Consequently, the choice of these two standards as the basis for hull-form data exchange required little deliberation. However, selecting one or another exchange standard is only the first step. There are still three practical aspects that require attention and careful consideration: (a) erroneous models, (b) representation variation, and (c) supplementary requirements.

So, the first practical issue is that exchanged models are frequently not directly usable. This is typically because they are incomplete - for example, a set of NURBS surfaces that do not join seamlessly - or inconsistent, such as a set of three-dimensional curves that are intended to intersect but do not share common intersection points. Most CAD systems provide extensive facilities for analysing, repairing or augmenting such models, see e.g. *Koelman and Soede (2002)*, but these procedures often require manual intervention, are time-consuming, and cannot be relied upon to produce a correct result in every case.

The second issue is representation variation, discussed earlier as the phenomenon that a single exchange standard may offer multiple valid representations for the same object. Free-form hull geometry is most commonly represented by IGES entity types 126 and 128, corresponding to NURBS curves and NURBS surfaces, respectively. Since [Fairway](#), the hull-form modeller within PIAS, employs a different internal geometric representation, exporting to IGES requires an actual geometric conversion rather than a simple translation. Developing a robust conversion algorithm proved to be a substantial undertaking, but ultimately resulted in a reliable solution, see *Koelman and Veelo (2018)* for details.

The third and last issue is that receiving applications frequently impose additional requirements that go beyond the exchange standard itself. Historically these often took the form of restrictions on the maximum number of curves or surfaces, although such limitations have largely disappeared. More common today are implementation-specific conventions, such as prescribed naming schemes, entity ordering, or required orientations of surface normal vectors. Such requirements might be understandable because they ease downstream processing, yet they are not part of the data standard and are not always well documented. Consequently, successful interoperability depends not only on complying with the standard, but also on satisfying the expectations of the receiving application. In practice, this further increases the diversity of accepted representations beyond what the standard itself prescribes. Within the SEUS project, the IGES export facilities of PIAS were therefore tailored specifically to the requirements of CADMATIC - a provider of ship engineering and production software, and a constituting partner in SEUS - resulting in a fully automated and seamless transfer of hull-form geometry, as illustrated in Fig.3.

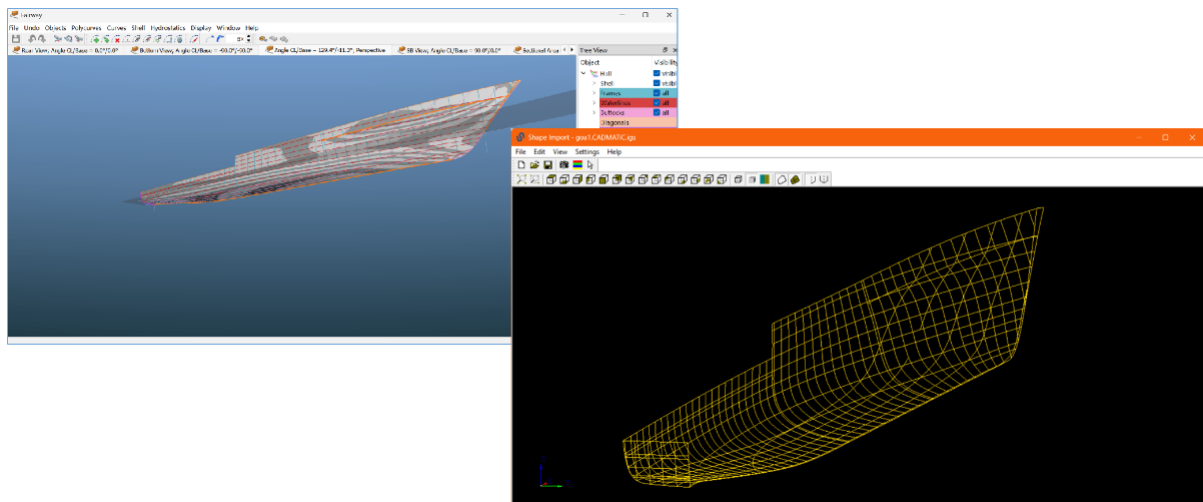


Fig.3: Hull form in PIAS (left) and in CADMATIC (right) after transfer via a tailor-made IGES interface

This example also demonstrates that it is not always possible to choose the most appropriate data exchange standard independently. For the exchange of complex free-form hull geometry, IGES remains the de facto industry standard, and software developers have little choice but to accommodate it. A second example is provided by a set of scripts developed at SARC to drive OpenFOAM for the prediction of calm-water flow and resistance using hull geometry generated by Fairway. The geometric input required by OpenFOAM is essentially a tessellated surface consisting of triangles.

Consequently, recalling the earlier discussion on representation variation, the hull geometry must first be converted from Fairway's internal representation into a large collection of small planar triangles. The file format most commonly used for this purpose is [STL](#). However, STL does not support the explicit definition of surface patches, whereas such information is required in OpenFOAM to apply different mesh refinement levels to different parts of the hull. For that reason, [Wavefront OBJ](#) was selected instead. It is an old, simple, and lightweight format, but one that is well suited to this particular application.

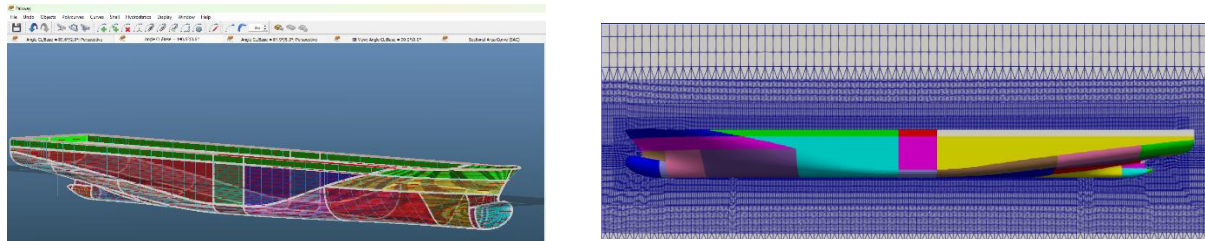


Fig.4: Hull form in PIAS (left) and in OpenFOAM (right) after transfer by Wavefront OBJ

3.2. Ship design data beyond hull shape

In a sense, shape turned out to be the easy part, simply because widely accepted exchange standards exist and are routinely used in practice. The situation is quite different for design information that is not part of the hull geometry. Examples include:

- Individual properties of design entities, such as the design densities of tanks, the engine MCR, the reduction gearbox ratio and weight, or design constraints such as the maximum permissible tonnage and principal dimensions.
- Geometric parameters, such as the positions of decks and bulkheads defining a compartment, the locations of tank level sensors, the reference points of sounding pipes, or the centres of gravity of containers in their design positions.
- Relationships and functional dependencies, for example between speed and required power, draft and deadweight, ambient conditions and HVAC energy consumption, or the Safe Working Load diagram of a crane.
- Non-numerical properties, such as the required classification notations or the desired exterior colour scheme of the vessel.

None of the previously discussed geometry-oriented exchange formats (DXF, DWG, IGES, STEP AP242, STL, or Wavefront OBJ) is intended to represent such information in a structured manner. (Of course, it is possible to include certain annotations in the text accompanying a graphical model, but their meaning or purpose is not covered by the standard at all. For example, we once came across a case where, for shell plate expansions, cutting information had been encoded using colours in a DXF file. Unfortunately, the colour legend had disappeared over time, leaving no one able to interpret the information correctly.) In practice, therefore, there is essentially no generally accepted exchange standard for this type of design data. The only noteworthy exception is STEP AP215 (Ship Arrangement), which addresses many of these concepts (and rightly so, because it's a 974-page ISO document). Nevertheless, from an implementation perspective it has several practical limitations.

- Despite its status as an ISO standard, industrial adoption appears to be extremely limited. In SARC's experience, we have never encountered an AP215 exchange file in practical use.
- The standard defines a large number of individual entities and properties, but many of the relationships between them remain implicit. For example, the definition of the block coefficient as the displaced volume divided by $L \times B \times T$ is described in the specification, but this dependency is not represented explicitly in the exchanged data. (Incidentally, if this relationship were explicitly included in the interface, the block coefficient could be omitted, as any downstream program can evaluate that relationship itself.)
- Numerous application-specific entities are absent. For example, the total length of a sounding

pipe can be included, whereas its complete three-dimensional geometry (particularly when shaped) cannot.

- AP215 is specified in the abstract modelling language [EXPRESS](#), but the standard itself does not prescribe how this information should be encoded in an operational data stream or exchange file.

Nevertheless, there is little practical alternative. Around ten years ago, SARC therefore started equipping the various PIAS modules with an interface that (a) follows the philosophy of STEP AP215, (b) extends it where necessary in the same spirit, and (c) uses XML as the transport layer. The XML representation need not even be stored in a physical file; the same data can equally well be exchanged directly over a network connection.

It has never been our objective to expose every piece of data used internally by PIAS through this interface. On the contrary, its implementation has been entirely demand-driven: new entities and relationships are added only when required in practice. The SEUS project has undoubtedly given this development a significant boost.

Finally, a few words on maintaining this “standard”. It initially consisted of little more than a Microsoft Word document containing the definitions and descriptions of the exchanged entities; the dictionary. Over time, however, it has grown to a size for which Word is no longer an appropriate maintenance environment. Moreover, every definition now exists in at least two places: once in the dictionary and once in the implementation itself, without any explicit link between the two. This duplication has become increasingly difficult to maintain and clearly calls for a more robust solution, which is proposed in Subsection 4.4.

3.3. Primary and Derived Data

So far, we have referred rather generally to data. In practice, however, it is useful to distinguish between primary and derived data. For example, if the hull geometry together with the positions of the watertight bulkheads and decks are considered primary data, then the tank geometry is derived from them, and the tank capacity tables are, in turn, derived from the tank geometry. Similarly, if the hull form, propeller geometry, and propulsion characteristics are regarded as primary, then the corresponding speed–power relationship is a derived result.

In principle, derived data need not be exchanged at all, since they can always be recomputed from the primary data. This assumption, however, presupposes that every receiving application implements the same computational algorithms, which is rarely the case in practice. For this reason, PIAS exports may include derived information (such as tank geometry, tank calibration tables, or stability results) as a service to downstream applications. (Incidentally, there is an alternative to this, namely the implementation of a server (an API) that makes such algorithms available to the outside world. This is an appealing concept that has been explored previously, but a detailed discussion falls outside the scope of this paper.) It should be recognized, however, that these data are derived and therefore informational rather than authoritative. They are not intended to be modified and returned to PIAS with the expectation that the underlying ship design will be updated accordingly. In other words, only primary data should drive the design; derived data merely describe the consequences of that design.

The current PIAS data dictionary occasionally indicates whether a particular entity represents primary or derived information, but this distinction is not applied consistently. Moreover, the interface architecture does not inherently prevent derived information from being reintroduced into the design process. This represents an area for potential improvement.

4. Implementing Robust and Effective Data Exchange Tools

PIAS’ export system is developed in the D Programming Language, <https://dlang.org/>, which will also be used for the following demonstrations. However, the syntax is similar to C, C++, Java and C#,

and might be familiar. The D language not only allows low level control and high performance like C, but also code introspection and generation. The reasons for SARC's choice of this language have been discussed in *Veelo (2018)*, but in particular its advanced introspection capabilities are excellent for automatic solutions to data exchange code, which we explore in this section. We will start with a naive way of export code, identify problems therein, and present our solutions. The following data structure will be used as an example:

```
struct Compartment
{
    string name;
    double weight;
    Position position;
}

struct Position
{
    float l;
    float b;
    float h;
}
```

4.1. Naive import / export code

The straightforward way to export the Compartment struct to an XML file is writing out each field to a file manually, one by one. (Note: This is flawed example code, don't do this in production, use a proper XML library instead.)

```
void exportXml(Compartment compartment)
{
    File output = File("output.xml", "w");
    output.write("<Name>", compartment.name, "</Name>\n");
    output.write("<Weight>", compartment.weight, "</Weight>\n");
    output.write("<Position>\n");
    output.write("  <L>", compartment.position.l, "</L>\n");
    output.write("  <B>", compartment.position.b, "</B>\n");
    output.write("  <H>", compartment.position.h, "</H>\n");
    output.write("</Position>\n");
}
```

For this demonstration, the key observation is that the amount of `write` statements is linearly proportional to the number of fields and the number of types that get exported. Similarly, to import XML back into the internal structure, a linear number of `parse` statements is required. While effective and manageable at first, the maintenance cost increases as the project expands. Consider that whenever a field is added, code needs to be updated in multiple separate locations. It is easy to forget this, resulting in import/export going out of sync, or fields missing from the output entirely. When `N` places in the program do their own I/O this way, the same work is done `N` times, each time slightly differently. This problem gets exacerbated when new output formats are added. For example, when adding JSON export code:

```
void exportJson(Compartment compartment)
{
    File output = File("output.json", "w");
    output.write("\"Name\": \"", compartment.name, "\",\n");
    output.write("\"Weight\": ", compartment.weight, ",\n");
    output.write("\"Position\": {\n");
    output.write("  \"L\": ", compartment.position.l, ",\n");
    output.write("  \"B\": ", compartment.position.b, ",\n");
    output.write("  \"H\": ", compartment.position.h, ",\n");
    output.write("}\n");
}
```

So, S structs, M struct members and F formats, will result in $O((S + M) * F)$ lines of code. This now results in quadratic scaling: every time an export format is added, the amount of code multiplies by (S+M). It may even deteriorate with other variants, if for example instead of output to file, multiple destinations are supported naively (like in-memory buffers or network sockets), the amount of code multiplies further.

4.2. Abstractions

To combat the combinatorial explosion, different variants should be abstracted into a single interface. That means separating concerns: JSON code should only know about the JSON format and know nothing about ‘Compartment’ or XML. Similarly, ‘Compartment’ knows nothing about JSON. But since JSON export functionality needs to know both about Compartment and JSON, there needs to be a ‘bridge’ between the two, which is the interface. This interface needs to be powerful enough to represent all needed formats, but otherwise as simple as possible to prevent the maintenance problems as described in the previous section.

Our solution begins by abstracting the various output formats, recognizing they are all different notations describing a tree. XML, JSON and YAML are structurally the same thing: nested records with named fields holding text, numbers and Booleans, just with different syntax. We create a common tree data structure and write for each output format a single function converting that tree to any output format that is supported. The tree node is a tagged union: either a leaf holding a primitive value, or a record holding child nodes. The ‘union’ part means it holds either of the types inside that union, but only 1 is active at the same time. The ‘tag’ stores which one is currently active. What this looks like in code is as follows: (simplified from PIAS’ actual implementation):

```
// What kind of Node is it?
enum Representation { struct_, string_, int_, float_, bool_, reference }

struct Node
{
    string tag; // XML tag / JSON key
    Representation repr;
    Node*[] children;
    union
    {
        bool asBool;
        double asFloat;
        long asInt;
        string asString;
    }
}
```

The export code for Compartment now builds a tree instead of writing text and no longer knows about any particular format. Each struct starts as an empty node (created with newChild), and gets built out with newChild, newLeaf and newChild calls describing each field to export.

```
Node* toTree(ref Compartment c)
{
    Node* root = newChild(null, "Compartment"); // this is the root, parent = null
    newLeaf(root, "Name", c.name);
    newLeaf(root, "Weight", c.weight);
    Node* pos = newChild(root, "Position");
    newLeaf(pos, "L", c.position.l);
    newLeaf(pos, "B", c.position.b);
    newLeaf(pos, "H", c.position.h);
    return root;
}
```

Each format is then a single function on the tree, written once; each direction is one function as well:

```

string treeToXml(Node* tree);
string treeToJson(Node* tree);

Node* xmlToTree(string xml);
Node* jsonToTree(string json);

```

Where the text ends up is a separate concern again. The format functions return a string, and the caller decides whether that string goes to a file, an in-memory buffer or a network socket.

```

IPCNode* tree = toTree(compartment);
std::file.write("output.xml", treeToXml(tree));
std::file.write("output.json", treeToJson(tree));

```

This creates a pipeline where each stage is completely unaware of the other stages.

[Internal struct] => [Generic tree] => [Text format (xml, json)] => [Output sink (File, socket)]

Because of this, instead of $S \times F \times D$ functions, there are now $S + F + D$: adding a format or a destination is one function, independent of the number of structs.

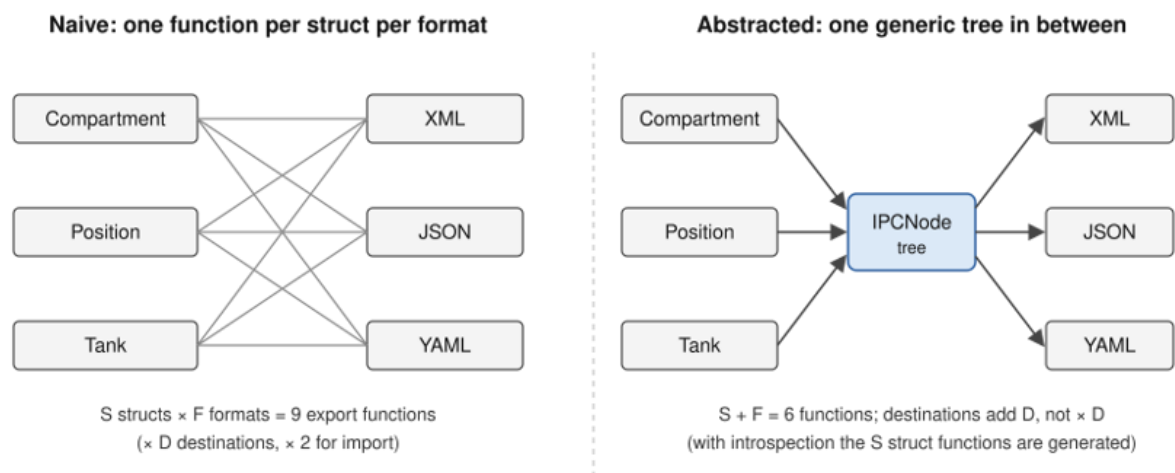


Fig.5: One function per struct per format (left) versus a generic tree in between (right)

4.3. Introspection

While the tree removes the need for copying code for each export format, one layer of redundancy remains: the code to create a generic tree from the internal data structures, enumerating fields a second time. This could be derived from the structure itself instead.

The D programming language has a rich assortment of meta-programming and introspection capabilities, which can be combined to automatically generate import code, export code, and documentation. User Defined Attributes (UDAs) are used to directly attach all the required information to the fields. The `Compartment` example becomes:

```

@IpcTag("Compartment")
struct Compartment
{
    @IpcTag("Name") @IpcDoc("Name of the compartment") string name;
    @IpcTag("Weight") double weight;
    @IpcTag("Position") Position position;
}

@IpcTag("Position")
struct Position
{
    @IpcTag("L") float l;
    @IpcTag("B") float b;
}

```

```

    @IpcTag("H") float h;
}

```

The hand-written `toTree` function from the previous section is now replaced by a single generic one, which works for any annotated struct:

```

IPCNode* root = ipcNewChild(null, "root");
toIpcTree(compartment, root, "Compartment");
string xml = treeToXml(root);

Compartment imported;
fromIpcTree(imported, xmlToTree(xml));

```

Internally, `toIpcTree` iterates over the fields of the struct and reads the attributes to generate the code that was formerly manually written. The D features used for this are ‘static foreach’ and ‘.tupleof’, which roughly looks like this:

```

static foreach (i; 0 .. T.tupleof.length)
{
    visit(t.tupleof[i], memberName, node);
}

```

Where T can be substituted by any type, in our example `Compartment`. Going into detail on how these advanced D features work is out of the scope of this paper, but observe how this is a fixed amount of code that doesn’t need to be changed no matter how many fields are added to ‘Compartment’: it’s the compiler adding iteration to the loop, not the programmer writing new lines for each field. Because there is a single source of truth, import, export and documentation cannot go out of sync when a field is added: adding the field with its attribute is all that is needed.

4.4. Dictionary

While the tree abstraction and introspection solve the code maintenance problem, in practice the semantics of the field also have their own challenges. Different synonyms are in use by different parties (e.g. weight, mass), as well as different units (meter, millimeter), axis conventions (left-handed, right-handed), can all be used to describe the same quality of a model. One party might export a numeric “offset: 10” field, and another party might read that “10” into their own “offset” field, but if both have their own idea of the unit and reference point of that number “10” then the exchange still isn’t successful. Without an integral, maintained description of the words used in the exchanged data, data may be exchanged but still lacking interoperability as there is no consensus on which fields to read and how to interpret them.

Documentation is often written in a separate document, but this has the same maintainability problems as discussed in Subsection 4.1. With introspection however, this can now be annotated directly in code with `@IpcDoc`, and exported to EXPRESS, *ISO (2004)*, a standard data modelling language, from the same structs, for example:

```

ENTITY Compartment;
    guid: String;
    name: String;
    designDensity: Number;
    subcompartment: ARRAY OF SubCompartment;
    closedShell: ClosedShell;
END_ENTITY;

-- Specifies at which side of the ship it is located
TYPE Side = ENUMERATION OF (Sb, Ps, SbAndPs, NoSpecificSide, Undefined)
END_TYPE;

```

EXPRESS is not a widely adopted standard, however, it is neutral in that it doesn't favour a specific file format. Thanks to the automatic generation it can always be swapped out later with something else if desired. Either way, the generated schema creates a data dictionary in a standardized, machine-readable form, with the documentation of each entity and field as comments.

Since this EXPRESS schema is generated, it is always in sync with what the program actually reads and writes. A more structured tool for managing the dictionary itself is currently being considered, but that involves getting agreement with multiple parties on what medium to host this on. While standardizing and understanding data conventions remains largely a communication problem, this approach at least makes the technical side of the problem easier.

4.5. Challenges

The introspection-based code generation works best when everything is consistent and clean, but real-world code is written under real world constraints and often consists of technical debt. Telling developers that their code must be cleaned up to be in a perfect clean state before our tool is usable would block adoption. Instead, pragmatic escape hatches are provided:

- A struct can define its own `toIpc` / `fromIpc` member functions, which take over the conversion for that type only, while the rest of the tree is still generated.
- `@IpcExportFunc` / `@IpcImportFunc` do the same for a single field, for example to convert units or to translate a legacy representation into a clean one.

A further challenge is the conversion of pointers. XML is essentially a tree data structure, but real pointer data structures may contain shared leaf nodes, leading to redundant copies, or cycles, leading to infinite recursion.

During export, a map from object address to tree node is kept. The first time an object is reached through a pointer it is written in full and given an identifier (a [GUID](#)). Any later occurrence becomes a `reference` node that only carries that GUID. Only objects reached through a pointer or class reference get an identity; structs stored by value are copies and are written in place.

During import the process is reversed: as objects are constructed, they are registered by GUID. A `reference` node whose target has already been built is resolved immediately. A reference to a target that appears later in the document (a forward reference) is recorded as a fix-up and patched once the outermost import finishes. Import and export share this bookkeeping with the custom `toIpc` / `fromIpc` hooks, so a reference crossing a hook boundary still resolves.

For cases where a reference per occurrence is still too much, the hooks receive the document root as well as the object's own node. A mesh, for example, can write its vertex pool once at the root and have every face store only indices into it.

4.6. Benefits of D

With PIAS' import/export system, the application programmer doesn't need to know the details of output formats, only how to build the reusable tree. While PIAS' tree is implemented in the D Programming Language, creation and export/import functions are also exposed through a plain C interface. C is considered the 'lingua franca' of programming: most programming language can call C functions, so application code in any language could build a tree with a handful of `ipc_new_*` calls and then ask to export to any supported format.

With most programming languages, it is possible to reap the benefits of abstracting the output formats as described in Subsection 4.2. However, using introspection to create a complete single source of truth like in Subsection 4.3 requires more advanced language features. With most languages it will be

possible to engineer something to do the same, but often in a complicated way, like running non-standard external tools over source code before actually compiling the program. D uniquely integrates this in the compiler in a standard and flexible way.

As discussed in Subsections 4.4 and 4.5, the challenges of naming, defining, and standardizing data remain, but it eliminates a class of bugs and maintenance work on the technical side and makes code future-proof against new output formats. Previously, updating old XML export code to support JSON would cost a lot of programmer time, while now it only requires minimal bridging from the generic export tree to a library for the new format.

5. Application example

From the PIAS suite, with the Layout module bulkheads, decks and compartments can be modelled. XML export and import routines using the naive way (Subsection 4.1) have existed for some time. Recent efforts have been made to update to the new system, and new JSON export functionalities start out using the new tree system. The challenges and experiences described in Section 4 were not merely academic exercises; they were drawn from real-world experience with these applications, which have recently been updated to leverage the new introspection-based tree system. Showcasing the system in action is best done in a live demo, but for reference, Fig.6 shows the internals of a ship modelled in Layout and being imported in CADMATIC. The exchange format used here is XML. Regarding import, we are currently working on integration with the design pipeline as presented in *De León et al. (2026)*.

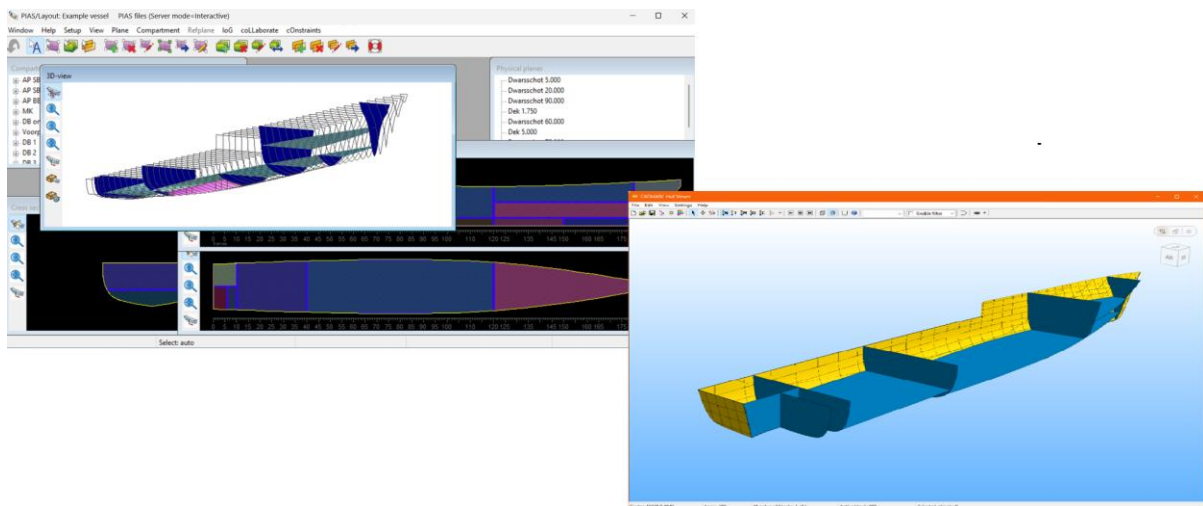


Fig 6: The same ship layout in PIAS (left) and CADMATIC (right)

Acknowledgements

The SEUS project has received funding from the Horizon Europe Framework Programme under grant agreement No 101096224.

References

- AGIS, J.J.G.; BRETT, P.O. (2024), *Digital Shipbuilding – Needs, challenges, and opportunities*, Int. Marine Design Conf., Amsterdam, doi.org/10.59490/imdc.2024.849
- DE LEÓN, D.; KOELMAN, H.J.; OTJENS, S. (2026), *Automatic Adaptation of Ship Subdivision: An Exploration on Methanol Conversion*, 24th COMPIT Conf., Tullamore
- GALLAHER, M. P.; O'CONNOR, A. C.; DETTBARN, J. L.; GILDAY, L. T. (2004), *Cost Analysis of Inadequate Interoperability in the U.S. Capital Facilities Industry (NIST GCR 04-867)*, National

Institute of Standards and Technology, U.S. Dept. Commerce, doi.org/10.6028/NIST.GCR.04-867

GASPAR, H.M.; SEPPÄLÄ, L.; KOELMAN, H.J.; AGIS, J.J.G. (2023), *Can European Shipyards be Smarter? A Proposal from the SEUS Project*, 22nd COMPIT Conf., Drübeck, pp.407-416

GIELINGH, W. (2008), *An assessment of the current state of product data technologies*, Computer-Aided Design, 40(7), pp. 750-759

ISO (2004), *ISO 10303-11: Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual*, Int. Standard Org., Geneva

ISO (2022). *ISO/IEC 81346-1. Industrial Systems, Installations and Equipment and Industrial Products - Structuring Principles and Reference Designations - Part 1: Basic Rules*, Int. Standard Org., Geneva

KOELMAN, H.J.; SOEDE, W.B. (2002), *Construction of a manifold ship hull model, derived from a wireframe mesh*, 4th Int. Symp. on Tools and Methods for Competitive Engineering, Wuhan, https://www.sarc.nl/resources/uploads/2002/04/tmce2002_koelman.pdf

KOELMAN, H.J.; VELO, B.N. (2018), *Fitting Watertight NURBS Patchworks over Irregular Curve Networks with LEANURBS*, 17th COMPIT Conf., Pavone, pp.226-236

KOELMAN, H.J. (2023), *Tell me*, 22nd COMPIT Conf., Drübeck, pp.64-72

KOELMAN, H. J.; VELO, B. N.; SEPPÄLÄ, L.; FILIUS, P. (2024), *Closing the gap between early and detailed ship design models*, IMDC Conf., Amsterdam, doi.org/10.59490/imdc.2024.837

KURYŁO, P.; FRANKOVSKÝ, P.; MALINOWSKI, M.; MACIEJEWSKI, T.; VARGA, J., KOSTKA, J.; ADRIAN, Ł., SZUFA S.; RUSNÁKOVÁ, S. (2023), *Data Exchange with Support for the Neutral Processing of Formats in Computer-Aided Design/Computer-Aided Manufacturing Systems*. Appl. Sci. 13, 9811. doi.org/10.3390/app13179811

LI, F.F. (2025), *From words to worlds: Spatial intelligence is AI's next frontier*, Substack drfeifei.substack.com/p/from-words-to-worlds-spatial-intelligence

SHI, X.; DING, J.; ZHAO, X.; ZHAN, S.; MOHAPATRA, P.; QUISPE, D.; WELBECK, K.; CAO, J.; CHEN, W.; GUO, P.; ZHU, Q (2026), *STEP-LLM: Generating CAD STEP Models from Natural Language with Large Language Models*, arXiv preprint doi.org/10.48550/arXiv.2601.12641

VELO, B. (2018), *How an engineering company chose to migrate to D*, The D Blog, <https://blog.dlang.org/archive/2018/06/20/how-an-engineering-company-chose-to-migrate-to-d/>

WHITFIELD, R.; DUFFY, A.; YORK, P.; VASSALOS, D.; KAKLIS, P. (2011), *Managing the exchange of engineering product data to support through life ship design*, Computer-Aided Design 43(5), pp. 516-532

From Legacy 2D Drawings to OCX: Automated Structural Knowledge Graph Construction for Model-Based Ship Approval

Ahmad Ostovar, TraceArc, Stockholm/Sweden, ahmad@tracearc.com

Shahan Khan, TraceArc, Stockholm/Sweden, shahan@tracearc.com

Vivek Ramaswamy, TraceArc, Stockholm/Sweden, vivek@tracearc.com

Abstract

Model-based class approval still depends on information that exists mainly in 2D drawings. This paper presents an evidence-preserving pipeline that requires no task-specific model training or fine-tuning to recover engineering semantics and partial structural topology from legacy ship drawings and organize them as a knowledge graph aligned with Open Class 3D Exchange (OCX) concepts. Explicit evidence, inference and unresolved items remain distinguishable throughout. Applied to a publicly available NSMV Midship Section drawing, the pipeline reconstructed 73 structural objects and a graph of 1,356 relations, of which 60 objects were assigned OCX concept candidates. A readiness audit reports the missing longitudinal, deck-plan, shell and connection information rather than inferring it.

1. Introduction

Ship structural design is already strongly digital, yet many approval and coordination activities are still conducted through drawings. Even when a drawing originates from a 3D CAD model, the information that reaches another discipline or an external reviewer is often flattened into PDF, annotated, revised and returned as a document. This is not merely a legacy-data problem. Drawings remain convenient contractual and review artefacts, and they carry compact combinations of geometry, scantlings, dimensions, notes and conventions that experienced engineers interpret almost implicitly.

At the same time, the maritime industry is moving toward model-based approval. The Open Class 3D Exchange (OCX) standard was developed specifically to exchange ship structural information between design tools and classification workflows. Recent COMPIT papers have shown both the practical value of OCX and the remaining organizational concerns surrounding model-based approval. *Son et al. (2022)* demonstrated the use of OCX as an exchange layer in a 3D approval workflow; *Gušani et al. (2023)* investigated structural model reuse across design systems; and *Peippo et al. (2025)* argued for an essential structural model that is sufficient for approval without exposing unnecessary production detail. These developments create a clear question for projects in which the authoritative information is still drawing-centric: how much of the engineering meaning can be reconstructed before a native 3D product model is available?

The problem is harder than optical character recognition. A profile designation such as “HP 140x8” is useful only when it can be associated with the correct structural location, while a bracketed thickness such as “[12]” requires the drawing notes to recover its H36 material meaning. An opening annotation such as “400x110 OPNG” is also incomplete until its leader and host geometry are resolved. Deck/level labels provide engineering coordinates, while centreline references provide a second local coordinate system. Repeated graphical motifs may be structural symbols, dimension graphics or simply fragments created by line removal. In other words, the drawing must be interpreted as a system of evidence rather than as an image containing independent detections.

Research on engineering-drawing digitisation has progressed rapidly. Deep-learning approaches can detect symbols and components, *Elyan et al. (2020)*, *Zhang et al. (2023)*, while hybrid neural-symbolic parsers can turn technical drawings into queryable representations, *Van Daele et al. (2021)*. More recent work has emphasized the value of knowledge graphs for preserving relationships rather than only extracted text, *Fagan et al. (2023)*. In the maritime domain, *Ha and Gaspar (2025)* converted General Arrangement drawings toward 3D models, *Weidemann et al. (2025)* explored AI-assisted checking of technical ship drawings, and *Bronson et al. (2025)* used retrieval-augmented language models to

validate information across heterogeneous ship-design documents. The present work is complementary: it focuses on structural drawings, keeps the extraction auditable, and intentionally avoids requiring model fine-tuning or a large language model in the core pipeline.

The work addresses three research questions. First, can a pipeline that requires no project-specific model training recover useful structural semantics and partial 2D topology from dense ship drawings with limited setup? Second, can the same pipeline operate both when a document-internal symbol legend exists and when it does not, without inventing semantic labels? Third, which recovered entities and relationships can be aligned with OCX concepts, and which information remains missing for a defensible 3D exchange model?

The main contribution is an intermediate structural knowledge representation positioned between a 2D drawing and a future OCX model. It retains provenance, distinguishes explicit evidence from inference, and records unresolved information instead of forcing every detected item into the target schema. This distinction is important for class-related workflows, where an apparently complete but weakly supported model is less useful than a partial model whose gaps are explicit.

2. Background and related work

2.1. Drawing-centred ship design and digital information management

Shipbuilding information is distributed across drawings, lists, 3D models, analysis models, specifications and production systems. *Seppälä (2023)* describes this as a data-management problem as much as a design problem, arguing that intent, data and process layers need to be managed together. The issue remains visible even in advanced digital projects: *Bronson et al. (2025)* note that 2D drawings, 3D models and specifications can contain overlapping or conflicting design parameters, while multi-structure product-data approaches are being developed to provide spatial, system-oriented and engineering-oriented views of the same vessel information.

The persistence of drawings is also evident in AI research presented at COMPIT. *Weidemann et al. (2025)* tested commercial multimodal AI for detecting alignment errors in technical ship drawings and found useful assistance potential, but also highlighted the need for more reliable automation. *Ha and Gaspar (2025)* addressed a different drawing type, using segmentation to convert General Arrangement drawings into 3D arrangement models. Both studies reinforce the practical value of extracting more structure from 2D documents rather than assuming that a clean, unified 3D model already exists.

2.2. Engineering drawing understanding and knowledge graphs

Engineering diagrams are unusually sparse images: most pixels are background, while meaning is concentrated in thin lines, small symbols and text whose interpretation depends on location and connectivity. *Jamieson et al. (2024)* review this problem across complex documents and engineering diagrams and show that no single vision architecture solves the full digitisation task. *Elyan et al. (2020)* demonstrated deep-learning-based symbol detection, whereas *Zhang et al. (2023)* vectorized raster drawings and used graph convolution to classify drawing components in context. *Van Daele et al. (2021)* combined neural and symbolic methods to parse technical drawings with limited expert interaction.

A graph representation is particularly attractive when the intended output is engineering knowledge rather than a redrawn image. *Fagan et al. (2023)* show how historic engineering drawings can be converted into knowledge graphs while retaining human oversight. *Luttmer et al. (2024)* similarly demonstrate automatic graph construction from engineering standards, illustrating the wider shift toward machine-actionable engineering knowledge. These studies motivate the separation adopted here between low-level drawing evidence, semantic objects and relationships.

2.3. OCX and model-based approval

OCX provides a domain-specific target for this extracted information. The formally published OCX schema version 3.1.0 dates from July 2025, 3docx.org. The schema includes structural concepts such as plates and stiffeners, catalogued cross-sections, panel geometry, trace lines and connection configurations. A stiffener, for example, is not only a label and section size: OCX represents its geometry and relation to plating, and can carry end-connection and cut information. This is the level of information needed for a true model-based exchange.

Previous COMPIT work has documented the transition. *Son et al. (2022)* showed how an imported OCX model can support rule scantling, direct-strength analysis, review, drafting and translation to downstream formats. *Gušani et al. (2023)* evaluated reuse of OCX structural models between different design systems. *Peippo et al. (2025)* focused on industry concerns around the scope of model-based approval and argued that the exchanged model should contain the information necessary for compliance rather than the entire production model. The present study starts one step earlier: the input is not an OCX-capable 3D model but a set of legacy structural drawings.

3. Proposed methodology

The proposed pipeline is deliberately staged. Each stage creates an auditable intermediate result that can be inspected before the next stage adds interpretation. This makes errors easier to localize and allows stronger learned components to replace individual modules later without changing the information model. Fig.1 summarizes the processing workflow used in the reported case study.

Pipeline from public legacy 2D drawings to OCX-aligned structural knowledge

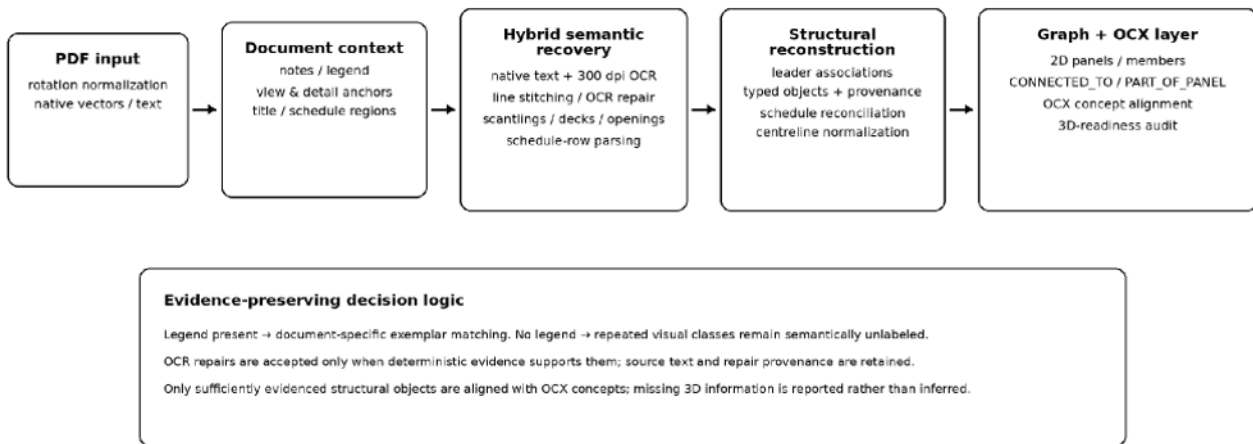


Fig. 1: Processing pipeline used in the public case study, including PDF rotation normalization, hybrid text recovery, schedule reconciliation and conservative topology/OCX alignment

3.1. Input handling and document context

The pipeline is vector-first but not vector-only. Native PDF text and vector primitives are extracted whenever available, and page coordinates are normalized through the PDF rotation transform before text, leaders and geometry are compared. This was necessary for the public case study, whose pages are stored with a 90° rotation. Some engineering lettering is represented as vector outlines rather than searchable text, so a 300 dpi OCR layer supplements native text instead of replacing it. Native and OCR items are geometrically de-duplicated and retain their source type for provenance.

Before local objects are parsed, the system detects document-level context such as Drawing Notes, legends, title blocks, schedule/table regions and drawing-view anchors. The notes can change the interpretation of many local annotations. In the NSMV drawing, for example, they state that major

views are of the starboard side looking forward, that both sides are symmetric unless noted otherwise, that steel is mild steel unless marked, and that a bracketed thickness such as [12] denotes H36 strength with a minimum yield stress of 355 N/mm². The notes also define typical frame spacing and stiffener end-connection codes. These rules are stored as contextual evidence rather than copied silently into every annotation.

The same context stage determines whether a symbol legend is present. If a legend exists, symbol-label pairs can become document-specific examples. If no legend is present, the pipeline switches to legend-free mode and does not assign an engineering name to a graphical motif unless another trusted exemplar exists. Visual repetition and semantic meaning therefore remain separate evidence dimensions.

3.2. View segmentation and ownership of drawing information

Large ship drawings often contain several independent views and detailed sketches on one sheet. The segmentation stage uses view titles, detail references, structural ink, local whitespace and exclusion of the page border to create a core geometry box and an expanded extraction box. The latter is allowed to include owned labels and dimensions without allowing those annotations to distort the geometric segmentation. In the public case, the system distinguished the Typical Ordinary Frame, the Typical Web Frame and Details A and 0-8, while schedule tables were retained as information regions rather than treated as structural views.

Named details are treated as first-class views because they often contain connection semantics that are absent from the principal section. Their text and geometry are parsed independently and then linked back to the document graph. Schedule/table regions are handled separately and excluded from graphical-symbol and structural-line candidate generation, preventing table grids from being misinterpreted as ship structure.

3.3. Hybrid text recovery, normalization and semantic parsing

Native and OCR text items retain page coordinates and source provenance. The parser recognizes plates, HP stiffeners, flat bars, built tees, girders, brackets, openings, decks/levels, reference lines and offsets from centreline, together with property-only callouts. The public drawing required additional normalization of OCR artefacts and reconstruction of multi-line engineering text. Adjacent fragments such as profile dimensions followed by 'OFFSET T', 'PLT DOUBLER' or 'FCPLT' are stitched before semantic parsing when their geometry and syntax support the merge.

Deterministic OCR repair is permitted only when independent evidence supports it. For example, a truncated profile such as 400x180/10x1 can be repaired to 400x180/10x18 only when repeated occurrences of the same profile provide the missing value; the original OCR text, repair rule and reference value remain in the object provenance. The parser also extracts the stiffener schedule as structured rows. Repeated printed schedules are consolidated later at document level rather than counted as distinct engineering definitions.

3.4. Graphical motifs with and without legends

The motif stage is designed for cases in which no labeled symbol dataset is available. Text, long support lines and identified annotation graphics are first suppressed. Residual local components are reconstructed and described geometrically, after which repeated appearance/shape patterns are grouped into visual classes. Repetition is evidence that two marks are graphically related; it is not proof that they represent the same engineering concept.

When a legend is available, legend crops provide direct examples and can attach semantic labels to matching instances. In legend-free mode, the retained records keep neutral visual identities. Strong leaders, dimension graphics and other annotation primitives are rejected from the semantic symbol

inventory but retained in diagnostics. The public NSMV document contains no symbol legend, so the 199 retained graphical motif records remain intentionally unnamed.

3.5. Leader association and structural-object reconstruction

Engineering text is connected to geometry by tracing native PDF vectors and local structural support. The system records the relation type, target point, orientation and confidence. This allows an annotation such as an HP profile, plate thickness or built tee to become evidence for a physical object rather than remain a free-floating text item. The same separation prevents leaders and dimensions from inflating the visual-symbol inventory.

Typed objects are reconstructed from explicit annotations first. Profiles such as HP 140x8, built tees and flat bars create Stiffener/Girder/FlatBar objects when their geometry association is supported; plate and bracket annotations create corresponding objects; and text-supported OPNG callouts create opening hypotheses. Property-only thickness annotations remain PropertyCallout entities until a geometric host can be resolved. When a type is inferred from geometry rather than explicit wording, the object is marked as geometry-inferred and kept distinct from explicitly typed evidence.

3.6. Schedule consolidation, physical 2D topology and coordinate normalization

Repeated schedule rows are first consolidated into document-level engineering definitions while preserving every printed occurrence. Candidate links between schedule definitions and reconstructed objects are created only when profile dimensions and, when available, deck context agree. In the topology layer, collinear/connected vector segments are joined into ContinuousMember2D candidates, long structural lines near deck/level contexts form PanelBoundary2D evidence, and supported endpoint or T-junction geometry can create conservative CONNECTED_TO hypotheses. Closed contours are not promoted as openings solely because they are closed; the evaluated configuration requires explicit opening text or comparably strong supporting evidence.

Each structural object is normalized within its view. When a centreline reference is detected, an object can also receive a signed normalized position relative to that centreline. Metric offsets in millimetres are generated only when enough explicit offset anchors exist to calibrate drawing coordinates; otherwise the graph retains the qualitative/normalized centreline relation without inventing an engineering distance. This distinction is important in the public case, where centreline references are common but a reliable global millimetre calibration is not available.

3.7. Knowledge graph and OCX alignment

The knowledge graph separates drawing evidence, semantic annotations, reconstructed structural objects and physical topology. Relations include DESCRIBES, ASSIGNS_PROPERTIES_TO, MATCHES_SCHEDULE, PART_OF_PANEL, CONNECTED_TO and feature/reference links. Fig.2 shows the public ordinary-frame view before and after structural/topological reconstruction; Fig.3 illustrates how the evidence-preserving graph exposes an OCX-aligned subset.

OCX 3.1.0 is used as a semantic target rather than as the internal data structure. Explicit Plates, Stiffeners and supported section profiles can be aligned with corresponding OCX concepts, while panel contexts, connection hypotheses and opening features remain candidates when the 2D drawing cannot supply the complete 3D schema requirements. The pipeline therefore reports concept coverage and a separate 3D-readiness status, preventing a mapping percentage from being misread as percentage completion of a vessel model.

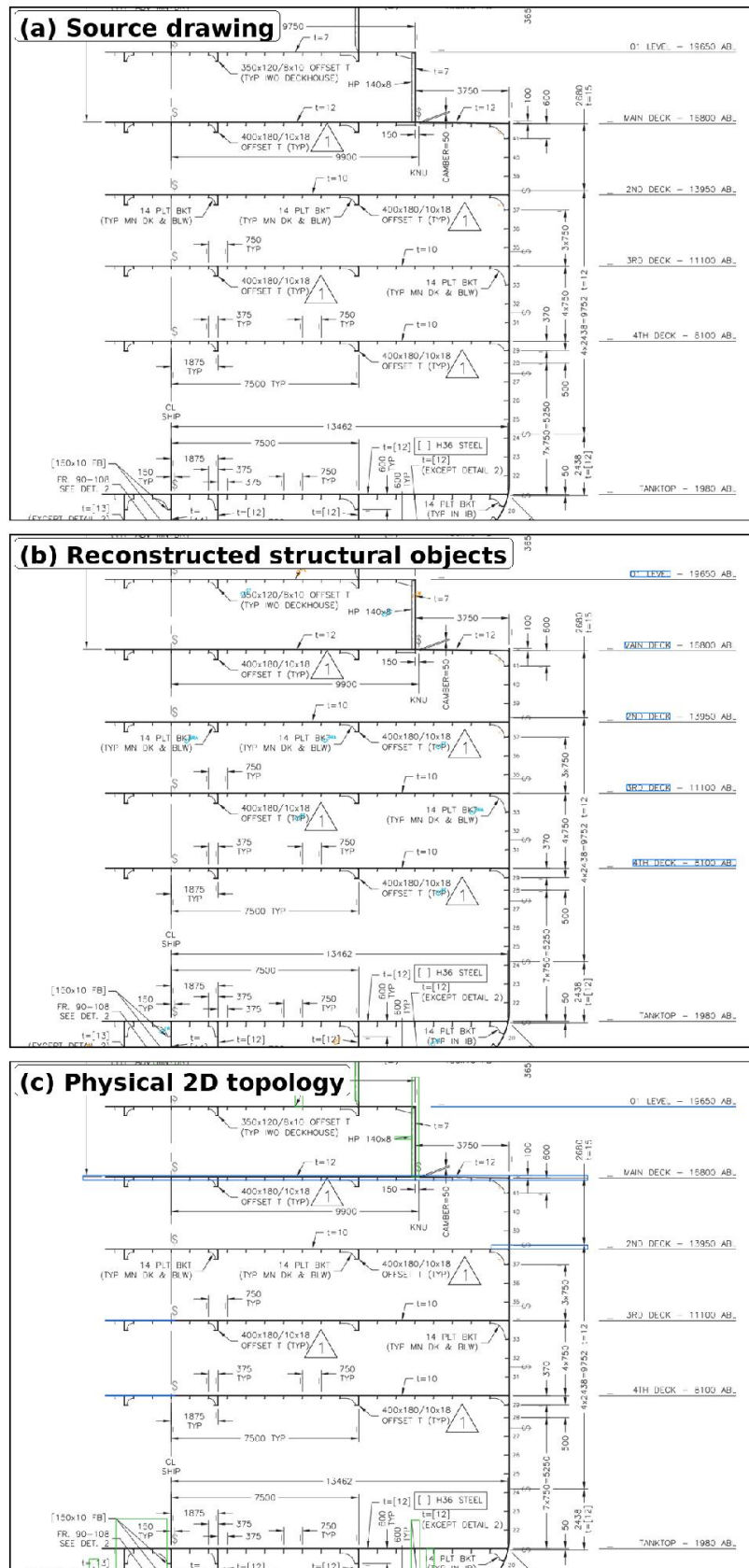
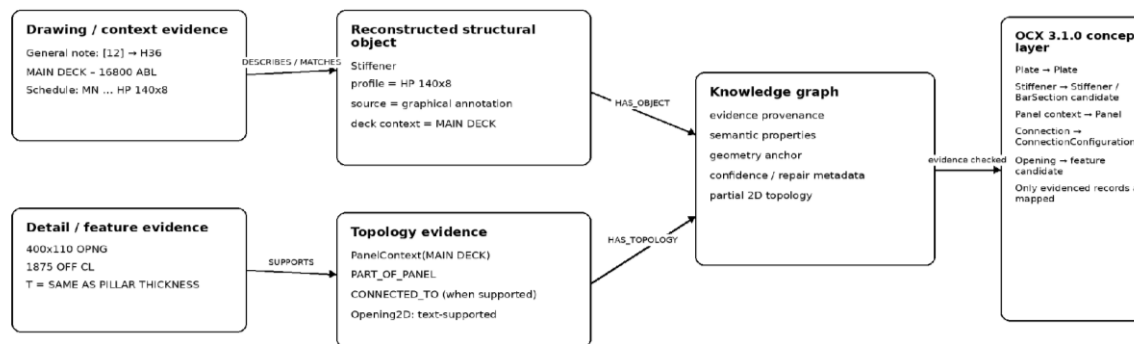


Fig. 2: Public NSMV Phase 3 ordinary-frame excerpt: (a) source drawing, (b) reconstructed structural objects and semantic associations, and (c) physical 2D topology. Source: *U.S. Maritime Administration (2016)*.

Evidence graph and OCX-aligned subset for the public NSMV case study



The graph retains source evidence, deterministic OCR repairs and unresolved items even when they are not eligible for OCX mapping.

Fig.3: Evidence-preserving knowledge graph and OCX-aligned subset for the public NSMV case study. Only records with sufficient supporting evidence are mapped to OCX concepts.

Table 1: Main information layers in the proposed representation

Layer	Examples	Typical output / relation
Drawing evidence	native/OCR text, vector path, drawing note, schedule row	source bbox/page, source type, confidence, repair provenance
Semantic entities	Plate/Stiffener annotation, Deck/Level, Offset, Opening, Girder	parsed properties; DESCRIBES / ASSIGNS_PROPERTIES_TO
Structural objects	Plate, Stiffener, Girder, FlatBar, Bracket	resolved 2D anchor; explicit vs geometry-inferred type
Physical topology	ContinuousMember2D, panel boundary, connection, Opening2D	PART_OF_PANEL; CONNECTED_TO; feature hypotheses
OCX alignment	Plate, Stiffener/BarSection, Panel, connection candidates	concept mapping + readiness + missing-information audit

4. Experimental setup

4.1. Drawing set

The case study uses the publicly available NSMV Phase 3 ‘Midship Section’, Drawing 2015-017-03-11, issued by the *U.S. Maritime Administration (2016)*. The PDF contains three sheets. Sheet 1 contains General Notes, abbreviations, principal characteristics, classification information, a drawing index, relief-cut examples and a built-tee dimensioning example. Sheet 2 contains the Typical Ordinary Frame, a Table of Stiffeners and Details 0–2. Sheet 3 contains the Typical Web Frame, the repeated stiffener schedule and Details 3–8. The document contains no symbol legend.

The drawing is useful as a reproducible robustness case because it mixes information types and drawing conventions on the same sheets. The pages are internally rotated, much of the fine engineering lettering is encoded as vector outlines rather than searchable PDF text, the stiffener schedule is repeated, and connection details contain short multi-line annotations. The General Notes also provide contextual rules for material strength, orientation, symmetry, frame spacing and end connections. Because the drawing is publicly hosted by MARAD, representative source excerpts and system overlays can be reproduced in this paper without exposing proprietary project material.

4.2. Implementation and evaluation protocol

The evaluated implementation is written in Python. PyMuPDF is used for native PDF text/vector extraction and coordinate transforms; Tesseract at 300 dpi provides the supplemental OCR layer;

OpenCV and scikit-image support image morphology; scikit-learn-based clustering supports legend-free motif grouping; and NetworkX stores the graph. The reported configuration includes deterministic text repair, view/detail extraction, schedule consolidation, structural reconstruction, topology generation and OCX mapping. No task-specific object-detector training, large language model (LLM), or vision-language model (VLM) inference is used in the reported extraction path.

All numerical results reported below come from one complete Google Colab Enterprise run of the fixed configuration on the three-sheet public PDF; no post-processing corrections were applied to the reported output. The results are system-coverage and consistency measurements, not object-detection accuracy. A manually labeled benchmark is not yet available, so precision, recall and F1 are intentionally not inferred from the number of extracted objects. The public NSMV document was also used while adapting the parser to rotation, outline text and multi-line annotations; it should therefore be treated as a transparent case study rather than an independent held-out benchmark.

5. Results

5.1. Document understanding and semantic extraction

The three sheets produced 13 extracted view regions: two principal frame views, ten named detail views (Detail A and Details 0–8), and one page-level context region. No significant overlap was found between the final engineering-view boxes. The semantic layer contains 154 entities. Seventy-seven annotations were associated with drawing geometry, and 73 structural objects were reconstructed: 60 explicitly typed from engineering text and 13 geometry-inferred objects. The repeated Table of Stiffeners produced 48 printed schedule occurrences that were consolidated into 20 unique document-level entries. Fourteen conservative schedule-to-object match candidates were generated. The complete graph contains 1,356 relations.

Table II: Extraction and graph-construction results for the public NSMV Midship Section case study

Measure	Count
Pages	3
Extracted view regions	13
Principal frame views	2
Named detail views	10
Significant view overlaps	0
Semantic entities	154
Annotation→geometry associations	77
Reconstructed structural objects	73
Unique schedule entries	20
Knowledge-graph relations	1356

5.2. Legend-free graphical discovery

The graphical pipeline generated 674 raw motif candidates. After reconstruction and filtering, 199 records were retained as repeated-motif candidates, 270 candidates remained unresolved and 205 strong annotation graphics were excluded from the motif inventory. Because the source document contains no legend and the project symbol library was empty, none of the 199 retained motif records received a semantic engineering name. This is deliberate abstention: the system can record repeated visual form without claiming that the form denotes a bracket, penetration or other named feature.

5.3. Structural reconstruction and physical topology

The topology layer contains 20 panel-context nodes and 39 PART_OF_PANEL relations. Forty-one ContinuousMember2D candidates and 20 panel-boundary candidates were reconstructed from vector evidence. Three physical structural-object pairs received conservative CONNECTED_TO evidence,

and one independent member intersection was recorded. Three Opening2D candidates were promoted, all supported by explicit OPNG annotations; no geometry-only closed contour was promoted as an opening. Sixty-one structural objects received a normalized position relative to a detected centreline reference. Metric offset-from-CL calibration was not reported because the views did not contain enough reliable anchors for a defensible drawing-coordinate-to-millimetre transform.

Table III: Physical 2D topology and cross-source reconciliation extracted from the public case study

Physical-topology item	Count	Interpretation
PanelContext nodes	20	deck/level context nodes associated with principal structural views
PART OF PANEL	39	object-to-panel context relations
ContinuousMember2D	41	joined 2D member candidates from vector evidence
PanelBoundary2D	20	2D deck/panel boundary evidence
Physical CONNECTED_TO object pairs	3	conservative endpoint/T-junction/shared-group connections
StructuralIntersection2D	1	independent member intersection
Opening2D	3	all supported by explicit OPNG annotations
Objects with CL-normalized position	61	signed/normalized position relative to detected centreline
Schedule→object match candidates	14	profile match; deck context used when available

5.4. OCX alignment and 3D readiness

At the preferred physical-object layer, 60 of 73 reconstructed structural objects were assigned an OCX structural concept candidate under the implemented mapping rules, corresponding to 82.2% mapping coverage at this layer. The 13 remaining objects are geometry-inferred and are deliberately not treated as equally strong OCX evidence. The semantic-entity layer has lower coverage because it also contains contextual, reference and property records that do not correspond one-to-one with OCX structural classes. The structural-object rate is therefore the more interpretable coverage measure; it is not an accuracy score and does not measure completion of a 3D vessel model.

The readiness audit concludes `partial_2d_semantic_topology_not_full_3d` and sets `full_valid_3docx_ready = false`. The midship-section drawing provides deck elevations, centreline references, scantlings/materials, ordinary/web-frame cross-sections, connection details, schedule information and partial 2D topology. It does not define the complete fore-aft extent of every member, complete deck-plan geometry, shell surfaces and plate boundaries, a full 3D coordinate system, or unambiguous welding/feature semantics for the entire vessel. The audit therefore recommends longitudinal sections, deck plans, shell expansion and welding/connection information as complementary sources before a schema-valid OCX model can be claimed.

Table IV: Information identified by the readiness audit as necessary before a complete 3D model in OCX can be claimed

Missing information for complete 3D reconstruction	Likely complementary source
Longitudinal extents and station coordinates of members	Longitudinal Sections / project reference data
Complete deck-plan member paths and panel boundaries	Deck Plans
Shell surface and shell-plate development	Shell Expansion
Complete plate boundaries, seams and 3D surface definitions	Midship + longitudinal/deck/shell sources
Connection, welding, bracket and feature semantics	Welding / connection drawings and specifications

6. Discussion

6.1. Why an intermediate graph is useful

The main practical result is not the 82.2% concept-coverage value; it is the separation of evidence from interpretation. The public drawing combines facts of different strength. ‘HP 140x8’ is an explicit profile statement. The General Notes make [12] a documented H36 marker. A schedule row such as MAIN DECK ... HP 140x8 provides a second source that can support a graphical object. By contrast, a repeated visual motif without a legend remains only a visual class. Keeping these evidence types separate makes the graph auditable and allows corrections to be traced back to their source.

The same principle applies to OCR repair. The public PDF contains outline text and small annotations that are difficult to recover consistently. The implemented workflow can stitch multi-line engineering text and repair a truncated profile only when repeated deterministic evidence supports the missing value; the source text and repair metadata remain in the graph. This is preferable to silently normalizing an uncertain OCR result, particularly in an approval-oriented workflow where a plausible but untraceable value can be more harmful than an explicit unresolved item.

6.2. Relationship to model-based approval

The result is consistent with recent OCX practice. *Peippo et al. (2025)* emphasize that model-based approval should exchange the structural information necessary for compliance rather than every production detail. The same principle applies in reverse when information is reconstructed from drawings: only evidence that is sufficiently supported should enter the OCX-aligned layer. The 2D graph can already support traceability, search and cross-source reconciliation even when the complete 3D surface and longitudinal definitions are unavailable.

A full 3Docx export from the present midship-section document would therefore be misleading. Cross-sections define local shape and scantlings but not the complete longitudinal trace of every plate and stiffener. Deck plans are needed for plan-view member paths and panel boundaries, shell information is needed for the outer-hull surface and plate development, and welding/connection information is needed to resolve connection semantics consistently. The current graph provides stable deck/level contexts, local structural objects, schedule entries and partial connections that can serve as anchors for future cross-drawing fusion.

6.3. Operation without task-specific model training and the role of AI

The evaluated implementation establishes a baseline that requires no project-specific training or fine-tuning. This does not mean that the workflow contains no learned component: Tesseract is a pre-trained OCR engine, and unsupervised clustering is used for motif grouping. The distinction is that no drawing-family-specific detector or language/vision model is trained for the case study. This is pragmatic for shipyards that may not have labeled data for each drawing family and where text styles and symbol vocabularies vary across projects. More powerful learned detectors or vision-language models can later be introduced at well-defined interfaces, for example to classify unresolved motifs or rank ambiguous type hypotheses, without changing the evidence and provenance model.

This differs from the approach of *Weidemann et al. (2025)*, where a multimodal AI system is directly asked to interpret drawing alignment, and from *Ha and Gaspar (2025)*, where deep segmentation supports 2D-to-3D conversion. It is closer in spirit to hybrid technical-drawing parsers such as *Van Daele et al. (2021)*: the objective is not to maximize the number of AI components, but to preserve engineering provenance and make abstention possible when the evidence is insufficient.

6.4. Limitations and validation requirements

The case study still has important limitations. First, it contains one public structural drawing set from one design and was used during robustness adaptation, so it is not an independent held-out benchmark. Second, the present evaluation is primarily a coverage and system-consistency study; manually verified precision/recall is not yet available. Third, OCR remains a source of variability because fine outline text can be recovered differently across runtime/library versions. Fourth, physical connectivity is intentionally sparse: the run produced three CONNECTED_TO object pairs, no opening-to-member links and no multi-anchor continuous member. The system also did not establish a defensible millimetre-scale centreline calibration.

A qualitative error audit identified several concrete failure cases that should remain visible rather than tuned away for this single drawing. The ordinary-frame view missed the TANKTOP label even though the associated 1980 ABL elevation was recovered; Detail 2 was isolated correctly but did not yield a promoted structural object; and some connection details retain only partial semantics. The next validation stage should therefore use hand-labeled public frame/detail views to measure object precision/recall, field-level property accuracy, annotation-association accuracy and relation F1. A second public drawing should then be processed without rule changes to provide a genuinely held-out generalization test.

Table V: Validation framework. Accuracy metrics requiring hand-labeled ground truth are intentionally separated from the coverage statistics reported in this paper.

Evaluation dimension	Recommended metric	Current paper status
Text / property extraction	exact match, field-level accuracy, OCR error rate	coverage + provenance reported; hand-labeled benchmark next
Structural-object reconstruction	precision, recall, F1 by object type	73 reconstructed objects; accuracy ground truth pending
Annotation association	relation precision/recall; target-point tolerance	77 associations produced; ground truth pending
Topology	PART_OF_PANEL / CONNECTED_TO / opening relation precision-recall	partial topology reported; relation ground truth pending
Schedule reconciliation	unique-row accuracy; object-match precision	20 unique entries; 14 match candidates
Graph consistency	constraint violations, duplicate/conflicting properties	auditable graph and provenance available
OCX alignment	mapped eligible objects / verified objects; schema validation	60/73 structural objects concept-mapped; full 3D validation not claimed
Engineering effort	review time, corrections per drawing	planned user study

7. Conclusions and future work

This paper presented a practical method for converting legacy ship structural drawings into an intermediate engineering knowledge graph aligned with OCX concepts. The method treats drawing understanding as evidence reconstruction rather than as a set of independent detections. Document notes and legends are interpreted first; native text and OCR are reconciled; principal sections and named details define ownership; annotations are associated with geometry; schedule information is consolidated; structural objects are reconstructed; and a separate topology layer records panels, members, conservative connections and text-supported openings. When no legend exists, repeated motifs remain unlabeled rather than receiving guessed engineering names.

The public NSMV Midship Section case study demonstrates that a useful amount of structure can be recovered without task-specific model training or an LLM/VLM in the extraction path. The reported run identified 13 view regions without significant overlap, created 154 semantic entities and 77 annotation-to-geometry associations, reconstructed 73 structural objects and 41 continuous 2D member candidates, consolidated 20 unique schedule entries, and produced a graph with 1,356 relations. The topology includes 39 PART_OF_PANEL relations, three conservative physical CONNECTED_TO object pairs and three text-supported opening candidates. Under the implemented mapping rules, 60 structural objects (82.2% of the reconstructed object layer) were assigned an OCX structural concept candidate.

The same experiment also defines the boundary of the claim. A midship-section document does not contain enough longitudinal, plan-view, shell and connection information for a complete schema-valid OCX vessel model. Future work should first establish a hand-labeled public benchmark and a held-out second drawing, then extend the graph across longitudinal sections, deck plans, shell expansion and welding/connection information. In that form, the approach could provide a practical migration path from drawing-centred engineering archives toward model-based approval while retaining the provenance and uncertainty needed for engineering review.

References

- BRONSON, J.; TEIMOURI, M.; GASPAR, H.; FONSECA, I.; BIERKOWSKA, K.; GINTER, F.; KOELMAN, H. (2025), *A RAG-based LLM Approach for Data Validation and Harmonization in Ship Design*, 23rd COMPIT Conf., pp.171-181
- ELYAN, E.; JAMIESON, L.; ALI-GOMBE, A. (2020), *Deep learning for symbols detection and classification in engineering drawings*, Neural Networks 129, pp.91-102, doi:10.1016/j.neunet.2020.05.025
- FAGAN, A.M.; WEST, G.M.; MCARTHUR, S.D.J. (2023), *Leveraging Knowledge from Historic Engineering Drawings*, 13th Nuclear Plant Instrumentation, Control & Human-Machine Interface Technologies, Knoxville
- GUŠANI, S.; RADONIĆ, M.; PUURULA, J. (2023), *OCX Standard and Structural Model Reuse in the Shipbuilding Design*, 22nd COMPIT Conf., Drübeck
- HA, J.; GASPAR, H. (2025), *Two-Stage Conversion of GA Drawings into 3D Model Using Deep Learning*, 23rd COMPIT Conf., pp.228-237
- JAMIESON, L.; MORENO-GARCÍA, C.F.; ELYAN, E. (2024), *A review of deep learning methods for digitisation of complex documents and engineering diagrams*, Artificial Intelligence Review 57/136, doi:10.1007/s10462-024-10779-2
- LUTTMER, J.; KANDEL, M.; EHRING, D.; NAGARAJAH, A. (2024), *Automatic knowledge graph creation from engineering standards using the example of formulas*, Proc. Design Society 4, pp.423-432, doi:10.1017/pds.2024.45
- OCX CONSORTIUM (2025), *Open Class 3D Exchange (OCX) Schema*, Version 3.1.0, 3docx.org.
- PEIPPO, J.; PUURULA, J.; PUDD, M.; PONKRATOV, D. (2025), *Solution Addressing Industry Concerns of Model-Based Approval*, 23rd COMPIT Conf., pp.245-254
- SEPPÄLÄ, L. (2023), *Integrated Shipbuilding Data Management*, 22nd COMPIT Conf., Drübeck
- SON, M.-J.; SEPPÄLÄ, T.; MERIKANTO, J.; AAE, O.; ASTRUP, O.C. (2022), *Utilization of OCX as Part of 3D Model Based Approval in Ship Design Process*, 21st COMPIT Conf., pp.258-272

U.S. MARITIME ADMINISTRATION (2016), *NSMV Phase 3 - Midship Section*, Drawing 2015-017-03-11, U.S. Department of Transportation,
<https://www.maritime.dot.gov/sites/marad.dot.gov/files/docs/national-defense/office-ship-operations/rrf/2641/midship-section.pdf>.

VAN DAELE, D.; DECLEYRE, N.; DUBOIS, H.; MEERT, W. (2021), *An Automated Engineering Assistant: Learning Parsers for Technical Drawings*, AAAI Conf. on Artificial Intelligence 35(17), pp.15195-15203, doi:10.1609/aaai.v35i17.17783

WEIDEMANN, M.A.T.; GASPAR, H.M.; LIEN, H.V. (2025), *AI-Assisted Misalignment Detection in Technical Ship Drawings*, 23rd COMPIT Conf., pp.190-204

ZHANG, W.; JOSEPH, J.; YIN, Y.; XIE, L.; FURUHATA, T.; YAMAKAWA, S.; SHIMADA, K.; KARA, L.B. (2023), *Component segmentation of engineering drawings using Graph Convolutional Networks*, Computers in Industry 147, 103885

AI-Based CAD Model Comparison for Heterogeneous Environments

Constantin-Alexander Gresens, PROSTEP, Hamburg/Germany, constantin-alexander.gresens@prostep.com
Carsten Zerbst, PROSTEP, Hamburg/Germany, carsten.zerbst@prostep.com

Abstract

Ship models are frequently exchanged between heterogeneous design tools, making reliable validation of model transfers an important challenge in modern shipbuilding. This paper presents an AI-assisted approach for evaluating the quality of CAD model transfers based on vendor-neutral representations. Geometric and structural information exported from different tools is analysed via a Model Context Protocol (MCP) server, enabling a large language model to compare corresponding steel design entities such as panels, plates, stiffeners, brackets, and other structural elements. The proposed workflow automatically identifies inconsistencies, summarises deviations, and provides an interpretable assessment of model equivalence. The approach explores how AI can support efficient validation of digital ship models and improve confidence in data exchange across different design environments.

1. Introduction

Modern, digital shipbuilding depends on a variety of specialised early, detailed, and production design systems, often from a number of different vendors. Achieving a faithful transfer of CAD models from one of these tools to the next is thus an important part of ship design. Regardless of how the transfers are carried out on a technical level, designers and engineers need to be able to trust the faithfulness of these model conversions. A transfer can superficially appear successful while still losing an entity, slightly moving a stiffener trace, changing plate or profile thickness or altering a contour. Such deviations can be difficult and inefficient to review manually, both because of the sheer number of entities, geometries and connections present in steel models for ships, and because two valid and equivalent model representations need not be textually identical. A useful validation process must therefore distinguish harmless representational discrepancies due to data model differences between the design tools from changes that affect the actual steel model.

This paper explores whether a large language model (LLM) can assist that process when the exchanged models are available as ShipXML documents, a vendor-neutral model representation in XML-dialect format. In this approach, the LLM is not treated as a geometry engine, comparing raw model files with all their potential representational differences. Instead, a Model Context Protocol (MCP) server gives controlled access to parsed model data and deterministic comparison functions.

In this context, this proof of concept compares two architectures. In Approach 1, the MCP server performs entity matching, numerical comparison and tolerance evaluation using deterministic comparator methods; the LLM selects (via MCP) from these functions, investigates findings returned by them and writes an interpretable assessment of the discrepancies between the models. In Approach 2, the server exposes broader sets of panel, plate and stiffener data and the LLM performs more of the correspondence and comparison itself; in this latter approach, less of the comparison is done deterministically using the MCP and more of it is left to the LLM directly. This paper both explores the potential of LLM-based model transfer validation itself and compares the abovementioned approaches to ask where language-model flexibility is useful, where it becomes unreliable, and how this boundary affects repeatability and traceability.

On the technical side, we provide an overview of this implementation as a C# architecture, a tool catalogue for both approaches using MCP, and an evaluation design based on unchanged controls and deliberately modified model pairs. We also restrict our exploration on plates and stiffeners, organised in panels, and leave the addition of further structural steel elements such as brackets to future work.

The practical conclusion is that broad retrieval tools are valuable for exploration, whereas repeated detailed validation should rely on deterministic evidence and use the LLM mainly for orchestration and explanation. Based on this comparison, we conclude that AI-assisted CAD model transfer evaluation can increase confidence in faithfulness and quality of steel model transfers in shipbuilding, as well as do so in an efficient manner.

2. Background and Problem Context

2.1. Ship Model Transfer Validation

A CAD steel design model combines geometry, structure and engineering properties and metadata of a ship or vessel. Panels organise plates and stiffeners; plates carry material, thickness, weight and contour information; stiffeners add traces, support relationships and profile sections. In addition, all these entities can also be linked via topological information or other metadata. Transfer validation must thus preserve both individual values, as well as relationships and connections to other entities. Adding to the challenges, exports can contain the same number of elements while describing different geometry, or have a different number of entities that, in fact, represent the same geometries and topologies; this can happen due to data model and architecture discrepancies between different design tools. Additionally, ordering, identifiers, numeric formatting and redundant representations can change without changing engineering meaning, so any high-level direct comparison between model representations can prove unfeasible.

The validation target in this study is semantic equivalence within stated tolerances. For a matched plate, this includes material, thickness, weight, centre of gravity and contour geometry. For a matched stiffener, it includes material, profile section, and trace geometry. At model level, it includes qualified counts of matched, missing and additional entities. The result of this should be seen and used as evidence for engineering review rather than an automatic acceptance workflow.

2.2. Characteristics of the Example Models

The examples are supplied as ShipXML documents originally exported from a NAPA Designer training model. This proof of concept focuses on panels, plates and stiffeners, but any productive model transfer would also include information about holes, cut-outs, notches, brackets, as well as any number of default or custom metadata.

Several values can also appear in more than one representation. Attributes such as thicknesses or weights of the same entities can appear in multiple places of the model; geometry may be provided through tool-specific representations, generic arc segment, NURBS curves, etc. Any model transfer validation thus requires explicit precedence and consistency rules: the parsers, functions and tools provided to the MCP should therefore establish some sort of authoritativeness of these representations as well. However, in the limited scope of this paper, we have made some simplifying assumptions as outlined in section 3. For example, we mostly rely on comparing arc segment representations of contours and traces.

2.3. Model Context Protocol and Tool Use

A naïve approach to AI-based model comparison could look something like the following: we prompt LLMs to compare two steel model files directly and provide a prosaic assessment of model equivalence. This approach was dismissed immediately - or, more truthfully, not even really considered - for a multitude of reasons. One of the more important ones is context consumption. Steel models for ships are very large, even when restricted to, say, block or (ship) section level, and would thus consume vast amounts of context of the LLM from the very beginning of the comparison process. Additionally, and equally as important, evidence-based traceability of the findings the LLM reports become virtually impossible if the context for any comparison is two entire ship model files. Human confirmation of such findings would be extremely difficult and the LLM itself would also be

much more prone to errors and hallucinations. In contrast, MCP defines how a client (LLM) discovers and calls tools relevant to ship model comparison exposed by a server. Each tool (or function, or endpoint) has a name, description and JSON Schema input definition, and a declared output schema. This type of structured content can therefore carry typed comparison evidence instead of prose assembled inside the server [Model Context Protocol Contributors (2025)]. Intuitively, we can view the MCP server as providing the LLM with reasonably sized chunks of information about a collection of or individual steel elements of a ship model. The LLM can then repeatedly query the MCP tools to get finer information as it needs it and incrementally build its context as appropriate for the comparison task it is currently undertaking (for example, comparing the same stiffener from two different model files).

The official C# SDK that was used in this experiment supports MCP clients and servers in .NET and provides hosting extensions; its main package is suitable for a local standard-input-and-output server as utilised in this paper, while the ASP.NET Core package supports HTTP scenarios [Model Context Protocol Contributors and Microsoft (2026)] that might be more suitable for a productive implementation.

We also wanted at least the possibility of ‘plugging in’ different AI models into our architecture, potentially from different AI providers, and leave the rest of the implementation unchanged. Most, if not all, candidate LLM-model providers support tool calling through their APIs (see, for example, OpenAI (2026); Anthropic (2026)). However, their request and response types differ, so provider-specific objects should remain outside the comparator and MCP domain model. Therefore, for this exploration, a small LLM adapter is implemented to translate the common application request into each provider's API format and convert tool calls back into a neutral representation. Note that for the limited scope of this paper, the adapter was only verified for OpenAI and Anthropic models, and we mostly used OpenAI models in the actual experiments.

3. Validation Requirements

For this exploration we made a number of assumptions and scope limitations and postulated some validation requirements that will be outlined in the following. We also want to be able to understand and confirm the LLM’s findings and assessments, so some requirements are needed when it comes to how the comparison process stores its findings and how it links them to evidence.

3.1. Comparison Scope

The PoC accepts two local ShipXML files labelled Model A and Model B. Model A is treated as the reference export and Model B as the transferred or re-exported model. The initial scope covers panel membership, plates and stiffeners. The main discrepancy classes are missing or additional entities, changed plate contours, changed stiffener traces, changed thickness or weight, displaced centres of gravity, and changed stiffener profile sections.

Table 1: Comparison scope for the proof of concept

Level	Compared information	Example finding
Model	Entity counts and unmatched entities	Two stiffeners missing from Model B
Panel	Identity, category and child membership	Plate moved to another panel
Plate	Material, thickness, weight, centre of gravity and contour	Thickness changed from 8 to 10 mm
Stiffener	Material, profile, support and trace	Trace displaced beyond tolerance
Geometry	Endpoints, sampled curve distance and derived measures	Contour differs locally by 6.2 mm

3.2. Equivalence and Tolerances

Some assumptions must be made when it comes to determining when two entities across different model files are considered equivalent (in geometry) and when a discrepancy is reported. Exact floating-point equality is unsuitable for exchanged CAD geometry in this regard due to different tolerances, rounding margins, etc. across different design tools. The comparator therefore uses configurable tolerances rather than relying on strict mathematical equivalence. A practical, if arbitrary, initial configuration is 1.0 mm for point and centre-of-gravity distance, 0.1 per cent for weight and area, and 0.1 degrees for direction. These are PoC defaults, and we do not claim these would serve as acceptance limits for production. Each result records both the observed deviation and the tolerance that produced its status.

Additionally, we must align curve representations so that we can compare them efficiently. Curves are canonicalised before comparison; their segments are converted into a common three-dimensional representation, based on arc segments; direction is normalised so that a reversed but otherwise equal contour does not fail; closed contours are aligned to a common starting point, then compared by endpoints, angles, and sampled distance, again based on arc segments. This is mostly used for Approach 1. It is, once more, to be noted that these are merely decision made for this exploratory proof of concept and should be made broader and more flexible in any productive implementation.

3.3. Entity Correspondence

Another high-level challenge that arises quite quickly in this exploration is the question of how to match entities between the separate models. With data model and naming differences between design tools, how do we identify two, say, stiffeners between the two model files that describe the same steel object in the actual ship. In this respect, the initial experiments use mostly persistent identifiers so that comparison behaviour can be separated from this - potentially harder - entity-resolution problem. Productive exchanges cannot assume this. In such an environment, matching pipelines could, for example, reserve multiple ordered stages: exact persistent identifiers, utilising mapping and configuration files used in the model transfer to match equivalent entities, deterministic similarity matching, and manual or LLM-assisted review of ambiguous candidates.

To elaborate a little further on this, mapping and/or configuration file contents could establish equivalence in the deterministic stage by applying custom name or identifier mapping to entities in Model A and thus identifying names and identifiers in Model B. Similarity matching can, for instance, combine parent-panel correspondence, material and thickness, bounding-box overlap, centre-of-gravity distance and geometric similarity to determine equivalence. The LLM may also explain why candidates are ambiguous, but it should not silently create authoritative mappings without evidence. Hence, future work can expose proposed matches for human confirmation and persist accepted mappings in a configuration file for human oversight and validation.

4. Proof of Concept Architecture

In the following we will outline the architecture of this experiment and provide some insight into what each stage does and why. This overview can seem a little technical, but (hopefully) intuitive explanations and descriptions are also provided so it is clear why this architecture was chosen and makes practical sense.

4.1. Components and Data Flow

The implementation is organised into a parser and normaliser, a comparison domain layer, an MCP server, an LLM adapter, and an evidence store. This separation permits deterministic tests without an LLM and AI-model experiments without changing comparator logic. Local file access is restricted to configured directories, and the AI-model receives structured extracts rather than arbitrary file-system access (mostly for security reasons).

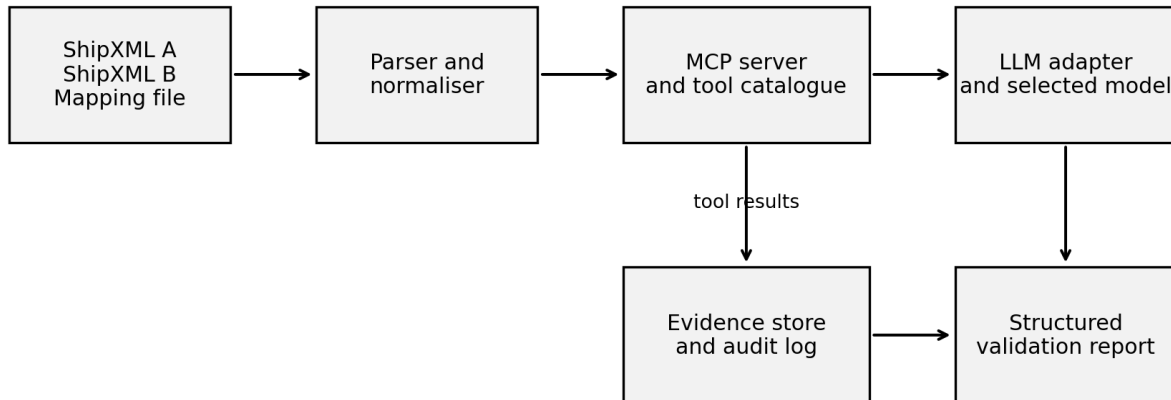


Fig.1: Logical architecture of the proof of concept

Original XML paths, identifiers and source values remain attached as provenance so every reported deviation can be traced back to both files.

4.2. C# Implementation

The proposed implementation targets a current supported .NET release and uses the official ModelContextProtocol package. The local PoC runs the MCP server over standard input and output so that it avoids network configuration and keeps model data on the workstation. An HTTP transport can be added later without changing the comparison services but has not been explored here. C# system libraries provide parsing for the XML documents and serialise typed tool results in JSON.

The comparison report contains matched and unmatched entities, property deviations, geometry deviations, severity (partly as defined in the comparator, partly as perceived by the LLM), tolerance, and evidence references. MCP tool methods call these services and return data-transfer records; they do not contain comparison logics themselves.

4.3. Normalisation and Evidence

Normalisation converts coordinate strings into numeric vectors, resolves declared units, removes insignificant numeric formatting and selects canonical values. The normalised representation also retains the original value and XML location.

Every discrepancy receives an evidence identifier. For example, a plate-thickness deviation records the entity pair, both values, unit, difference, tolerance and source locations. An LLM assessment must cite those evidence identifiers in its structured response, so that all reported issues can be checked and verified independently by a human. This is doubly important since even purported discrepancies listed in the report may not be actual - AI hallucinations will always exist to some extent. Requiring a high level of evidence detail thus constrains the model to available findings and makes review of potential errors and hallucinations practical.

4.4. LLM Adapter

The adapter offers - or rather can offer - a common application-level contract for model selection, messages, available tools, tool results, output schema, temperature (randomness/creativity of the respective AI model), as well as performance and budgetary information such as context consumption and token budget. Provider implementations translate this contract into API requests and the application stores prompts and generation settings with each run. Even though we mostly focused on OpenAI LLM models for this experiment, it could potentially use one model initially and add another later without modifying the MCP server or comparison logic. This allows for the comparison of

different AI-models from potentially different providers to see which might be more suited for (certain parts of) CAD ship model transfer validation.

The model output follows a schema containing an overall status, confidence, summary, deviations, evidence identifiers, unresolved questions and recommended review actions. Whereas there is practical sense behind these choices, the schema can – and most likely should – also be extended or amended when moving this experiment to a productive implementation. Also note that a schema-constrained output like this reduces formatting failures made by the LLM, but it does not guarantee validity by any means and result confirmation remains necessary, as outlined in 4.3.

5. Two MCP Tool Strategies

We shall now outline the two approaches in a little more detail and provide an overview of what functions the MCP server has available to present the model data to the LLM.

5.1. Approach 1: Deterministic Comparison Tools

Approach 1 exposes fine-grained tools backed by a deterministic comparator. Deterministic in this sense means that they have to be geometrically sound, hard-coded logic implementations that stand in contrast to any probabilistic comparisons LLMs would make. In this approach, the LLM begins with a model summary, inspects unmatched entities and deviation statistics, and requests detailed evidence via MCP only where needed. The server performs all tolerance decisions and geometric comparisons. The model's role is to plan the investigation, connect related findings and communicate their likely importance.

Table II: Fine grained tool catalogue for Approach 1

Tool	Purpose	Typical output
get_model_summary	Read counts, model metadata and parse warnings	Counts by entity type and warning list
match_entities	Run identifier and configured mapping stages	Matched, unmatched and ambiguous sets
compare_entity_counts	Compare counts by type and panel	Absolute and relative differences
compare_hierarchy	Check panel to plate and stiffener relationships	Moved or differently grouped entities
compare_plate	Compare properties and contour of one plate pair	Typed deviations and evidence identifiers
compare_stiffener	Compare properties, profile and trace of one stiffener pair	Typed deviations and evidence identifiers
list_unmatched_entities	Retrieve missing and additional entities	Compact entity descriptors
get_deviation_statistics	Aggregate findings by type and severity	Counts, maxima and distributions
get_comparison_evidence	Resolve selected evidence identifiers	Source values and XML locations

This strategy limits context size because the LLM sees differences rather than whole entity populations. This makes the approach very repeatable, and its main cost is engineering effort: each comparison capability must be implemented (deterministically) and tested before the model can use it.

5.2. Approach 2: Broad Retrieval Tools

Approach 2 exposes fewer and broader tools. A model can request panel, plate or stiffener records from both files, filter them by parent panel or identifier, and compare the returned values itself. The MCP server still parses and normalises XML because raw document fragments may contain

redundant representations or irrelevant information, which - among other considerations - consume unnecessary context.

Table III: Broad retrieval tool catalogue for Approach 2

Tool	Purpose	Typical output
get_entities	Return paged normalised entities by type and model	Unmatched sets of entities (panels, plates, stiffeners) from both model files
get_entity_subtree	Return one entity with selected children	Entity information and geometry including children
get_xml_fragment	Return a narrow source fragment when required	Semi-raw XML for selected entities

With this approach, the model receives enough data to potentially notice patterns that were not anticipated by a dedicated deterministic comparator. It can also adapt its investigation without adding a new endpoint for every question. The drawbacks are equally direct: matching and arithmetic become probabilistic, repeated runs may differ, and large entity collections compete with instructions and prior tool results for context. Paging is essential; for example, returning all stiffeners from the sample in one response would be a poor default which could quickly become inefficient and uneconomic.

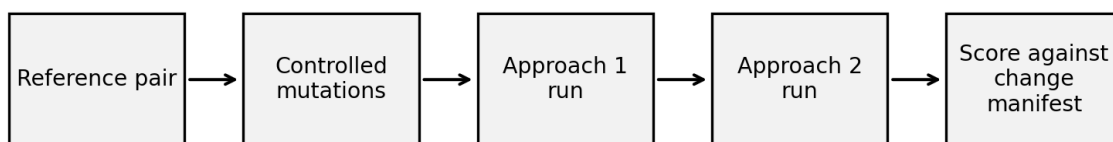
5.3. Intuitive Difference between Approaches

On a less technical level, the two approaches can be viewed as follows. Approach 1 requires considerable engineering effort in implementing the comparator functions that will be made available to the LLM via MCP. It trades this effort for a very high level of transparency and traceability in validation results, reduced context consumption and makes human confirmation of any findings in a structured way practical. Approach 2 on the other hand involves minimal engineering effort and leaves virtually all comparing up to the LLM. This can also have additional benefits - such as finding deviations between models that were not covered/considered in the deterministic comparator logic from Approach 1 - but trades them for vastly increased context consumption and token usage, as well as poor repeatability and traceability of reported findings.

6. Experimental Method

6.1. Test Corpus

The evaluation uses model pairs from several export paths, corresponding to ShipXML representations of the same ship steel model from different design tools. The corpus contains three classes. First, we simply compare identical files as a control group for the experiment, to ensure the LLM does not invent discrepancies. Second, repeat exports of unchanged models from different systems test whether tool-specific ordering or regenerated identifiers affect results. Third, controlled-change pairs contain known modifications recorded in a change manifest before the LLM starts the comparison.



Repeat each configuration and record detection, false positives, tool calls, tokens, latency and unsupported claims

Fig.2: Experimental workflow

For each base model, we prepared one unchanged control and three modified variants. Changes include deletion of a plate or stiffener, plate-thickness modification, contour-point displacement, stiffener-trace displacement, weight or centre-of-gravity modification, and profile-section substitution. Each variant contains between three and eight changes, producing enough combinations to test detection without making manual (i.e. confirmed by humans) ground truth impractical.

6.2. Procedure

Both approaches receive the same model pair, tolerance configuration and system-level task or prompt. Runs start without conversation history or non-system context. Approach 1 can call the complete deterministic tool catalogue, whereas Approach 2 receives only the broad retrieval tools as outlined in section 5. Each configuration is repeated five times to reveal run-to-run variation. The LLM adapter records model identifier, provider, generation settings, tool calls, elapsed time and the final structured assessment.

The controlled-change manifest is established beforehand and not provided to the model. After each run, reported deviations are matched to those manifest entries. A deviation is correct when it identifies the affected entity and property or geometry. A false positive is a reported change with no corresponding manifest entry: since we (humans) manipulated the model files ourselves and documented them in the manifest, we can easily detect invented discrepancies. Additionally, the LLM will also provide some ‘narrative’ reasoning for the discrepancies it found. These were also logged - because a report may reproduce all known changes yet add an explanation that the evidence does not support - but have not been analysed in the scope of this paper.

6.3. Measures

For this experiment, we measured the performance of each Approach using a set list of measure. These are explained in the following. Detection recall is the proportion of injected discrepancies reported. Precision is the proportion of reported discrepancies supported by the manifest or an independently verified export difference. Repeatability is measured as agreement between repeated runs on the set of reported discrepancy keys.

Again, note that the study is exploratory and the corpus is not intended to represent every ship type, modelling convention or design tool. Quantitative results should therefore support architectural observations rather than broad claims about industrial performance.

7. Results

7.1. Detection and Repeatability

On the unchanged controls, Approach 1 produced no false deviation reports in 30 repeated runs. Approach 2 produced a small number of inconsistent warnings concerning reordered entities and numeric formatting. These warnings did not correspond to engineering changes, illustrating the cost of asking the model to infer equivalence from broad records.

Across 96 injected discrepancies, Approach 1 detected 94 and Approach 2 detected 82. Both approaches found missing entities and simple thickness changes reliably. The largest difference occurred for geometric deviations: deterministic curve sampling detected local contour and trace displacement consistently, whereas the LLM sometimes compared only endpoints or summary properties. Approach 2 occasionally identified a changed weight but failed to connect it to the associated thickness change in its narrative reasoning.

Table IV: Discrepancy detection results

Measure	Approach 1	Approach 2
Injected discrepancies	96	96
True positive reports	94	82
False positive reports	2	13
Recall	97.9%	85.4%
Precision	97.9%	86.3%
Mean repeat-run agreement	98.1%	84.7%

7.2. Tool Use and Resource Consumption

Approach 1 used more tool calls because the model moved from summaries to selected detail. Nevertheless, each response was compact. Approach 2 used fewer calls but transferred substantially more structured data. Its context use grew with the number of entities (mostly stiffeners as those are more numerous than plates) and became sensitive to pagination choices.

Most unsupported claims in Approach 2 were plausible interpretations of geometric or hierarchy data rather than invented entities. They remain problematic because the final reader cannot distinguish a plausible inference from a computed result. Requiring evidence identifiers reduced this behaviour but did not remove comparison omissions.

7.3. Report Quality

Both approaches produced summaries that were a lot easier to review for a human than raw comparator output. Approach 1 reports were more consistent and could rank deviations using deterministic severity data. Approach 2 sometimes supplied useful exploratory observations, including clusters of changes within one panel, but its wording varied more between runs.

8. Discussion

8.1. Suitability of AI Assistance

The PoC supports a limited but useful role for AI in transfer validation. The LLM can decide which evidence to inspect, relate findings across entity types and convert structured deviations into language accessible to engineers and project stakeholders. MCP provides a clean boundary because the same model can discover tool responses without receiving unrestricted access to local files.

The comparison also shows why the LLM should not replace geometry algorithms completely. Model transfers require reproducible tolerance decisions, unit handling, curve comparison and stable correspondence. These tasks are straightforward to test in code and difficult to audit when performed implicitly inside a language model.

8.2. Interpretation of the Two Approaches

Approach 1 is the better foundation for repeated and detailed validation. It concentrates real-world engineering knowledge in explicit functions, controls context size and gives each conclusion a traceable source. The LLM adds value without becoming the authority for numerical equality.

Approach 2 remains useful as an exploratory mode. Broad retrieval tools allow the model to inspect properties not yet covered by a dedicated comparator and can help developers decide which deterministic checks to implement next. It is also a useful baseline that reveals what is lost when comparison logic is delegated to the model. The two modes can coexist in one server: validated tools for routine use and bounded retrieval tools for investigation.

8.3. Practical Safeguards

A later operational system should treat tool outputs and model text differently. Deterministic results should be signed or hashed with their input files and configuration. The LLM report should retain the model identifier, prompt version, evidence identifiers and tool log. Acceptance decisions should remain with a qualified user. The MCP specification similarly recommends human oversight, validation of tool inputs and results, timeouts and audit logging, *Model Context Protocol Contributors (2025)*.

Data governance also matters, as ship models usually are commercially sensitive. The local parser and comparator can run without transmitting complete files to a model provider. Only the structured evidence needed for assessment should leave the local process, and an on-premises model adapter should remain possible.

8.4. Limitations and Threats to Validity

The current scope emphasises planar steel structure and does not establish coverage for every construct present in ship steel model representations such as ShipXML. Exporter-specific interpretations, curved shell geometry, openings, brackets, seams and assemblies require further profiles and test cases. The sample's repeated representations also mean that a parser built for one CAD tool may not generalise without explicit schema profiles.

The use of persistent identifiers in much of the initial corpus also understates correspondence difficulty. Mapping files address known identifier changes, but automatic similarity matching introduces thresholds and ambiguous cases that need their own evaluation, not done in this PoC. Additionally, model APIs and tool-use behaviour change over time. The adapter reduces integration effort but does not guarantee equal behaviour across providers or model versions.

Finally, there may be errors or oversights in the engineering work done for the deterministic comparator logic of Approach 1. These could lead to some errors seen in the experiment, and a cleaner and broader coverage may increase recall, precision and repeatability, mainly for Approach 1.

9. Future Work

The next implementation step could be to complete the deterministic plate and stiffener comparators and generate more complex controlled model variants. Geometry tests should include reversed contours, different segmentations of the same curve, local offsets, split stiffeners, transformed coordinate frames, etc.

The matching stage should then consume one or more versioned mapping files and expose ambiguous candidates for confirmation. Further work can extend the schema profile to brackets, holes, cut-outs and seams, and can compare an on-premises AI-model with a hosted model through the same adapter.

10. Conclusion

AI can make CAD ship model transfer validation easier to investigate and communicate, provided that it operates over controlled engineering evidence. The proof-of-concept architecture presented in this paper separates parsing, normalisation, entity matching and tolerance-aware geometry from LLM orchestration and reporting. This division preserves reproducibility while allowing the model to guide an investigation and explain findings in accessible language.

The comparison of tool strategies suggests that broad data access is useful for exploration, but deterministic MCP tools are preferable for recurring validation. The practical value of the LLM lies in selecting, connecting and summarising evidence rather than deciding geometric equivalence on its own.

References

ANTHROPIC (2026), *Tool use with Claude*, Claude Platform Documentation, <https://platform.claude.com/docs/en/agents-and-tools/tool-use/overview>

MODEL CONTEXT PROTOCOL CONTRIBUTORS (2025), *Model Context Protocol specification: Tools*, revision 18 June 2025, <https://modelcontextprotocol.io/specification/2025-06-18/server/tools>

MODEL CONTEXT PROTOCOL CONTRIBUTORS AND MICROSOFT (2026), *MCP C# SDK*, <https://github.com/modelcontextprotocol/csharp-sdk>

OPENAI (2026), *Function calling*, OpenAI API Documentation, <https://developers.openai.com/api/docs/guides/function-calling>

Algorithmic Optimization of Space Allocation in Initial Design of Ships under Assigned Functional, Performance and Regulatory Requirements

Daniel van den Berg, Damen Research, Gorinchem/Netherlands, daniel.van.den.berg@damen.com

Piero Favero, Damen Research, Gorinchem/Netherlands, piero.favero@damen.com

Abstract

This paper presents an algorithmic solution for optimizing the space subdivision in initial design stage of vessels with assigned functional, performance and regulatory requirements. While optimization methods in naval architecture are commonly applied to hull form design, this study extends their use to the arrangement of internal spaces. The proposed solution combines design space exploration to generate design candidates with varying principal dimensions and a gradient descent algorithm to determine optimal bulkheads and deck locations for each candidate. The result is a method that identifies optimal subdivision layouts while simultaneously highlighting vessel dimensions that best satisfy the specified requirements.

1. Introduction

Design of modern vessels is a highly complex process where a large number of requirements need to be evaluated in parallel, *Andrews (2018)*. The decisions taken in the very initial stage, when least is known about the design, determine most of a ship's performance and cost, *Mavris and DeLaurentis (2000)*, *Andrews (2012)*, *Kana et al. (2016)*: principal dimensions are chosen, a space subdivision is drawn, and structure and systems are engineered on the basis of those choices. However, due to the complex nature of the problem, the quality of the chosen solutions can only be assessed once the design has matured in sufficient details: by that time, the consequences of early misjudgements have already propagated into structure, systems, layout and performance of the vessel, and correcting them may require prohibitively expensive rework. The cost for implementing significant changes has increased precisely as the knowledge about the problem was becoming available.

In the last decades, computational design space exploration became an increasingly adopted solution to tackle those limitations, *Papanikolaou et al. (2024)*, at least for the disciplines that have been increasingly automated. Methods to explore hull forms and to execute resistance, propulsion, weight and stability calculations multiplied over the years, and while specific implementations may differ significantly, they all rely on a design space exploration approach that is fundamentally well consolidated.

Thousands of design solutions can now be evaluated in the time it takes an engineer to draw one hull form. The physics of ship performance made this step toward automation possible: total weight is largely driven by hull dimensions and deadweight, and hydrodynamic resistance, intact stability and seakeeping only depend on the outer shape of the hull, together with its mass properties.

For a large part of the commercial and naval fleet, however, volume, deck area and the regulatory rules governing space subdivision matter at least as much as pure performance. These vessels are dimensioned by what has to fit inside the hull and the superstructure, and by how much it weights, therefore leaving the space arrangement out of the design space exploration will unavoidably optimize the ship against criteria which are not the ones deciding it.

Moreover, although mass properties for disciplines such as intact stability and seakeeping can be estimated, their distribution is ultimately determined by the internal arrangement: what the vessel carries, and where it is located. Solving space allocation for each design candidate therefore allows performance to be assessed from the resulting mass distribution rather than from external assumptions.

While typical disciplines of naval architecture have enjoyed substantial innovation, approaches to the synthesis of general arrangements have lagged behind, *Shields (2018)*. In the last two decades, academic research has increasingly tackled algorithmic approaches to internal subdivision, *Carlson (1987)*, *Parsons et al. (2008)*, *Andrews (2011)*, *Kim et al. (2025)*; however, the field has largely resisted the same treatment in industry: designers still approach the early stage general arrangement with the same tools they had available decades ago: a set of similar ships and regression formulae, now just wrapped into faster computer-aided toolkits.

There are multiple reasons for this slower innovation:

- A general arrangement involves a high number of variables, including functional and logical requirements of spaces, dimensions, volumes, areas, system arrangements, routing and onboard operations, which an experienced designer solves implicitly using a mix of intuition, know-how and talent.
- Vessel speed performance has traditionally dominated the optimization process, while space allocation would follow, adapting to a hull geometry defined on the basis of speed-performance requirements. Attention has since shifted towards hull optimization that still accommodates the general arrangement inside, but the two disciplines have rarely been solved in parallel.

It is out of the scope of this paper to discuss the comprehensive multi-objective optimization approach ultimately solving the holistic ship design optimization problem. Instead, this work addresses the more specific, and less explored, problem of space allocation, presenting a two-stage method to execute parametric optimization of the space allocation of a new vessel design.

Automated ship arrangement in early-stage design has largely developed around two recurring computational formulations: packing and assignment:

- Packing: objects representing the hull, subdivision, spaces and systems are positioned according to packing rules, and the ship configuration emerges from their spatial interaction. The Design Building Block approach of David Andrews and collaborators established a concept stage three-dimensional, architecturally explicit representation of the ship, where functional building blocks could be manipulated. Building on this tradition, Bart van Oers formulated arrangement generation explicitly as a packing problem. *Van Oers et al. (2010)* developed packing into a generative method producing large numbers of feasible configurations, while WARGEAR, *Le Poole et al. (2022a,b)*, addressed the related problem of rapidly detailing functional arrangements into space-level general arrangement plans.
- Assignment: the hull geometry and principal subdivisions are defined beforehand, creating a set of candidate regions such as zone-decks or compartments. The optimization assigns functional spaces to these regions and subsequently generates their detailed arrangements. The Intelligent Ship Arrangements work of *Parsons et al. (2008)* provides a comprehensive implementation of this formulation, evaluating candidate assignments through preference and utility functions.

Both approaches address a combinatorial problem, determining which space goes where, that grows quickly with the number of spaces. That is the appropriate formulation when the ship architecture is still unknown. The approach presented in this paper addresses a different problem: which space goes where is known, because the product architecture, including space topology, system allocation and key interfaces, is already available through reuse of a parent design or direct user definition.

The problem is therefore not where spaces should be located, but how large they need to be and where their boundaries should fall within a given design candidate. For each candidate in a design space containing thousands of alternatives, the algorithm answers a single question: is sufficient space available to realise the intended architecture? By framing the problem in this way, the combinatorial problem is replaced by a bounded positioning problem, that can be solved inside a design space

exploration loop. This reformulation is realized in two key steps presented in this work:

1. A framework based on a model-based system engineering approach to capture the requirements driving the space allocation and,
2. A mathematical framework that can be used to solve the space arrangement for a large set of different designs embedding the space allocation in overall vessel design trade-off.

2. Premises and assumptions

2.1. The quality of a subdivision can be measured by one single KPI

The overall premise on which this work is based is that, once the topology of the general arrangement is fixed, the fit between what a functional space needs and what the compartment hosting it provides can be reduced to a number.

- Every functional space carries requirements expressed on a set of geometric metrics.
- A solution to the space allocation problem (set of positions of the space subdivisions) determines the KPIs of each compartment.

The mismatch between the two, accumulated over every space in the ship, is a scalar measure of how good a solution to the space allocation problem is, and minimizing it is the overall objective of the space allocation function.

The residual value of the cost function at the end of the optimization, represents a “handicap” each candidate carries along and on which it will be evaluated.

2.2. The requirement set is a property of the design candidate

The requirements a compartment needs to satisfy are functions of the candidate being evaluated.

The volume required for fuel follows from the energy calculated in a route simulation, which follows from the resistance, which follow from the hull shape. The length required for the machinery space follows from the installed power and therefore from the performance of the hull. The distance a tank boundary keeps from the outer shell, the height of a double bottom structure or the position of a collision bulkheads are all examples of rule requirements written in terms of the dimensions of the ship.

Two candidates with different principal dimensions therefore differ in the space they make available and in the space their systems require.

2.3. The topology is fixed

Every candidate is an instance of the same architecture, inheriting the same number of space subdivisions and set of compartments. Candidates differ in where their boundaries lie, while the count of those boundaries and what they bound stays common to all.

The mapping of function to compartment belongs to the architecture and stays outside the set of design variables. The engine room stays where the architecture says the engine room is; under optimization it becomes larger or smaller while remaining an engine room in the same relative position in the ship.

While this boundary represents a constrain to the freedom of the optimization algorithm, the decision to adopt it follows from the sensible advantages it provides:

- Reuse of basic engineering - This is the industrial argument. An architecture that instantiates into many ships means that everything depending on topology is engineered once and inherited

by every member of the family: the systems and routing concept, the structural arrangement and its standard details, the arrangement principles agreed with class, the production sequence.

- Feasibility - by limiting the topology of the general arrangement to what has been proposed in the initial stage design, it is ensured that each design candidate produced in the optimization will maintain the same interfaces between adjacent spaces and the systems they host.
- Solvability - Since the logics of the spaces and their interfaces are scrutinized by designers and secured in the underlying architecture, the optimization algorithm is relieved from having to deal with all the interfaces and mutual constraints that combinatorial algorithm bring along.
- Computational efficiency - The combinatorial part of the arrangement problem is removed by construction, leaving to the optimizer the positioning of a modest number of boundaries. That makes a per-candidate arrangement affordable inside a population of thousands of candidates.
- Comparability - Because every candidate shares a topology, the residual mismatch means the same thing for all of them. Differences between candidates originate in their dimensions and their requirements alone, which turns the residual into a field mapped over the design space.

3. A model-based approach to space allocation

3.1. A two stages concept evaluation approach

This work presents a two-stage method to execute concept definition and concept evaluation with regard to the space allocation of a new vessel design.

- the first stage follows a model-based approach that captures the logical connections between the functional and performance requirements of the vessel, its systems requirements, the dimensions, weight and the cost of the functional spaces that each system will occupy.
- the second stage plugs that logic into an automatic design space exploration algorithm, which generates the variations within a constrained design space and evaluates each design candidate in terms of its KPIs.

The fundamental premise that defines this approach is that the vessel architecture is settled in the first stage by the user: this includes the number of space subdivisions and the logic of the space allocation, meaning which functional space occupies which compartment. The model needs to be built once at the beginning of a new product development, and it forms the basis for the cloud of design candidates generated in next phase.

Every design candidate is an instance of the same architecture, differing from its siblings in dimensions, capacity and performance while sharing the same arrangement topology.

The design space exploration algorithm will solve the physical implementation per each candidate, meaning how large each functional space needs to be and where its boundaries fall.

The optimization problem can then be formulated in the following way: “given a list of space subdivisions, compartments and functional spaces, where:

- each compartment dimensions (KPIs) depend on the placement of the subdivisions, and
- each function requirements depend on a set of high-level requirements and on the predicted performance of the candidate,

find the most optimal set of boundaries positions to ensure that the KPIs of each compartment meet or exceed the requirements of the function assigned to it”.

What emerges from the solution of the problem are the implicit connections and the trade-offs typical of highly constrained design spaces, which is what the method delivers to the designer.

3.2. The RFLP Structure

The first stage of the proposed methodology requires developing the architecture model according to the RFLP structure familiar from model based systems engineering. The RFLP structure consists of the following four abstraction layers which are approached in order:

- Requirements - The high level requirements are captured as formal statements: they emerge from the stakeholder needs, as the vessel operational profile, payload, complement, speed and endurance, class notations and the applicable statutory rules. Each statement is a traceable object in the system model.
- Functional - The functions the vessel performs follow from those requirements: accommodate crew and passengers store cargo and consumables, propel the ship, generate and distribute power, etc. Functions are allocated to spaces: a space exists in the model because a function needs it.
- Logical - The logical layer holds the architecture together: the set of spaces, the relations between them, and the rules linking each requirement, for each space to a driving quantity of the design. It states that the fuel tanks hold the energy demanded by the design route; while the logic stays the same for every member of the family, how many cubic metres of fuel tank that law translates into is particular per each design candidate, depending on its predicted fuel consumptions.
- Physical - The physical layer is the instance: a hull of given hull dimensions and coefficients and boundaries position. It is generated automatically by the design space exploration tool, one instance per design candidate.

During the design phase, the first three layers are constructed by the designers and MBSE specialists, once, at the beginning of the project:

- The requirement structure is inherited by the high-level requirements and the applicable rules and regulation. While the specific high-level requirements may differ per every project, the underlying requirement structure remains mostly unchanged across projects, and statutory requirements often apply to several projects unchanged. This ensures that an extensive database of requirements structures can be created to be reused across different projects, minimizing the implementation cost per project.
- The functional layer is derived from the requirements and generated by assembling a set of pre-built functional blocks, each representing a functional space of the ship.
- The logics are embedded in the functional blocks, as the basic governing equations of these building blocks are largely stable across different vessels.

The physical layer is produced by the solver, thousands of times. A requirement written once propagates to every candidate in the population, and a changed rule (a new class notation, a revised in-house standard) or a modification in the arrangement can be edited in one place, after which the whole design space can be re-evaluated against it.

3.3. The design space exploration phase

The second stage plugs the model into a design space exploration engine. Each design candidate is described by the baseline architecture plus a set of macro-parameters (principal dimensions and main coefficients), each given a range that the design team considers admissible. The engine samples candidates within the given design space, and a typical exploration involves between few hundreds and few thousands of them.

Among the other disciplines, the space allocation is solved once per candidate. It returns a converged solution of the subdivision problem (the position of every bulkhead and deck) together with a scalar cost value representing how well the obtained general arrangement of that candidate fits the assigned

high-level requirements. It is this cost value that enters the screening criterion alongside the other disciplines; this way, the arrangement becomes one of the criteria that select the hull.

4. The space allocation Model

This section introduces the space allocation part of the architecture model. At its core, the model consists of three layers: boundaries, compartments and spaces. Boundaries represent the physical subdivisions of the vessel and together define the compartments. Each compartment is then allocated to a space, which carries a function required for the vessel to meet its requirements. The three layers are explained in more detail below.

4.1. Boundaries

A boundary is a space subdivision separating two compartments. It is identified by its type (transverse bulkhead, longitudinal bulkhead or deck) and are characterised by a value representing its position in the space and a grid step value representing the minimum applicable discrete change to its position.

- A transverse bulkhead carries an X-position and can move along the length of the ship in discrete steps defined by the local frame spacing. Its position is bounded by the adjacent aft and forward bulkheads.
- A longitudinal bulkhead is identified by its Y position, moves in discrete steps across the breadth of the vessel and is restricted by the neighbour longitudinal bulkheads and the hull.
- A deck is an horizontal subdivision identified by its Z position which can move vertically in discrete steps while being constrained by the neighbouring decks.

All the boundaries are to be intended as an underlying grid on which subdivisions between compartments lie, and the array of their positions is the array of the design variables. A minimum of six extreme boundaries are always defined:

- the aft boundary, typically the transom, and the forward boundary, represented by fixed fictitious subdivisions located at the aft and fore ends of the vessel.
- the hull outer shell defines the two outermost longitudinal boundaries
- the bottom and the main deck define the lower and upper boundaries of the vessel main hull.
- Fig.1 shows an example of boundary edges overlayed on the reference vessel used in this work, the DAMEN VSX5209.



Fig.1: Example of boundary edges overlayed on a DAMEN VSX5209

4.2. Compartments

A compartment is the geometrical volume uniquely determined by six boundaries: two transverse, two longitudinal, two horizontal, with their positions and carries 5 geometrical metrics: the length, breadth, height, area and volume. These metrics are the key performance indicators (KPIs) of the compartment: all of them can be derived from the hull geometry and the six boundaries enclosing it by either estimates or direct solution of the compartment's hydrostatics.

The KPIs are used in the space allocation algorithm to evaluate how well a compartment can accommodate its allocated space. Fig.2 shows the same VSX5209 as in Fig.1, but now with the compartment metrics displayed.

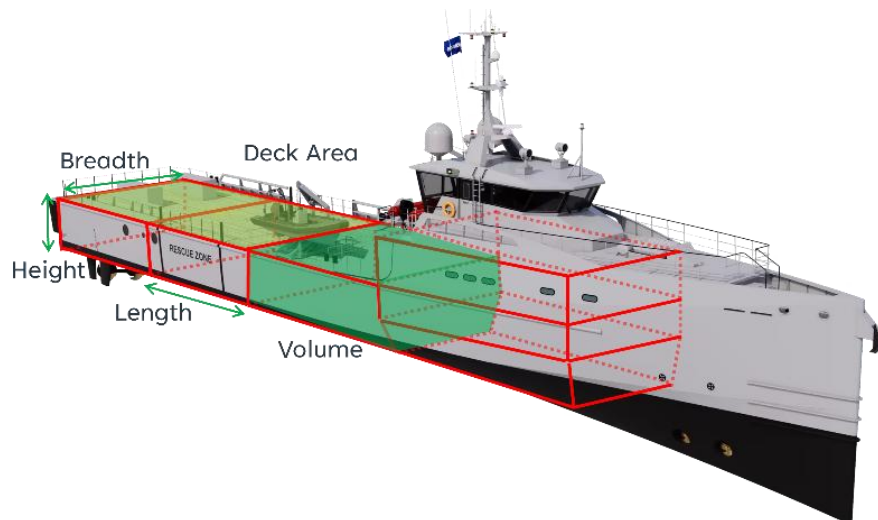


Fig.2: Overlay of compartment and metrics on the DAMEN VSX5209

4.3. Functional spaces

A space is an object carrying a function that needs to be accommodated: a tank, a machinery space, a cargo hold, an accommodation etc. A space is hosted by one or more compartments, and a compartment contributes to one and only one space. Each functional space object carries methods to determine the requirements it needs to fulfil in order to carry out its function, and the logics to evaluate how well the host compartment fit those requirements:

- the space requirements are expressed in terms of high-level requirements for the vessel and the performance of the specific candidate being evaluated.
- The measurement of fit evaluates how well the host compartment can accommodate the desired function.



Figure 3. Examples of spaces assigned to their compartments overlaid on the VSX5209.

By writing the requirements for a space in terms of the properties of the design candidate that is being evaluated, the space allocation becomes a constitutive part of the design space exploration. Figure 3 shows the VSX5209 example with spaces assigned to its compartments. Vessels that perform well on, e.g., resistance will require smaller machinery space and smaller fuel tanks, leaving more room for other spaces. Because every vessel in the cloud of design candidates will differ along its design parameters, every vessel will have a unique optimal space allocation.

Finally, the same space can also be allocated to different compartments. This can be the case of, for example, the fuel tanks that are spread over the vessel. When a space spans multiple compartments, the KPIs available to it are aggregated by a rule that follows the physical meaning of each metric, see Table 1. Extension metrics (deck area and volume) are summed, because they accumulate across compartments. Restrictive metrics (length, breadth and height) are taken as the minimum over the host compartments, because a clearance must be observed everywhere the space extends: a crew accommodation needs the required headroom achieved in every area of the space.

Table 1: The metrics on which requirements and KPIs are expressed, and how they are aggregated when one space is hosted by several compartments

Metric	What it constrains	Aggregation
Length	Fore-to-aft extent available to the space	minimum
Breadth	Athwartship extent, the longitudinal boundaries	minimum
Height	Headroom between the bounding decks	minimum
Deck area	Usable floor area, from the hull at the lower bounding deck	sum
Volume	Capacity enclosed by the six boundaries, from the hull	sum

4.4. Space requirements structure

Each space carries a set of rules expressing the requirement for one metric as a function of one or more driving quantity of the candidate. Rules are written per each of the five compartment KPIs, and per each of the functional spaces of the ship.

Let $\mathbf{R}$ be an array of n requirements $R_{i,d}$, with:

- $i=\{1..n\}$ the list of n applicable requirements for the dimension d of one functional space
- $d = \{\text{length, breadth, height, deck area, volume}\}$

For each space and each metric, an arbitrary number n of rules can be written based each on different driving quantities in order to capture requirements of different origin (functional, regulatory, contractual, company standards...). After they are all evaluated, the target value for the space dimension d emerges from the comparison of the applicable requirements as the most demanding one.

$$T_d = \max_{i=1..n}(R_{d,i}) \quad (1)$$

This paragraph will focus on calculating the requirement for one of the five dimensions of one functional space of the vessel architecture. The subscript d will therefore be omitted from here on to simplify the treatment.

The rules are written in the form of a linear function that expresses the required dimension as function of one driving quantity of the candidate:

Let $\mathbf{X}$ be the array of k driving quantities defined in the architecture model, upon which the requirements can be evaluated, and $\mathbf{C}$, $\mathbf{U}$, $\mathbf{Q}$ and $\mathbf{M}$ be the arrays of coefficients of determination that express how the driving quantities $\mathbf{X}$ affect the requirements $\mathbf{R}$ for a given dimension of one space.

$$\mathbf{R} = f(\mathbf{X}, \mathbf{C}, \mathbf{U}, \mathbf{Q}, \mathbf{M}) \quad (2)$$

- The function f is a linear function and it's constant across any type of functional space and architecture model, allowing the methodology and the design space exploration code to treat any type of space and architecture model.
- The array of driving quantities $\mathbf{X}$ characterize the new design, each candidate and its

performance: they come partly from the design inputs (complement, number of passengers, etc.), partly from the parameters of the design candidate being evaluated (principal dimensions) and partly from the its assessments (deadweight, propulsion power...).

- The arrays C , U , Q , M of coefficients of determination are the architecture model: they capture the logics of how the requirements R for a given function scale with the driving quantities X .

Each of the $i = 1 \dots n$ applicable requirements R_i is derived from one of the $j = 1 \dots k$ driving quantity X_j using the same linear function f:

$$R_i = \min(X_j \cdot C_{i,j} \cdot U_{i,j} + Q_{i,j}, M_i). \quad (3)$$

In equation (3):

- X_i $i = \{0..n\}$ is a generic driving quantity selected from a list X of k available quantities.
- The matrix C of dimension $n \times k$ is the matrix of the conversion coefficient: each of them expresses how much requirement R_i is affected by the variable X_j .

The conversion coefficient can be derived from regressions based on previously built vessels or they can directly convert X into the requirement based on basic physics laws.

For example, members of the matrix will express the length of the machinery space [m] as function of the installed propulsion power [kW] using regressions from already built vessels, or a fuel tank volume requirement [m³] based on the energy required to sail a given route [kWh], through the lower heating value of a chemical fuel [kW/kg] and it's density [kg/m³] (direct law) or through the energy density of a battery energy storage system (semi-empirical data).

- U is a matrix of dimension $n \times k$ of optional user coefficients used to create extra margin for the target value. For example, fuel provision may be increased by a percentage of the calculated fuel consumption on a route simulation, allowing for both design and operational margins.
- Q is a matrix of dimension $n \times k$ containing the offsets applied in the linear formula. It can be used to express logics such as “the engine room length needs to be C times the installed power plus a bias $Q = 2m$, as a reserve of space to ensure inspectability and serviceability of the equipment”. In other cases, the offset value is used when a space needs to meet a minimum constant size constraints: accessibility of voids, minimum values from statutory regulations are examples of this use, where $C_{i,j} = 0$ and $Q_{i,j}$ is set to the minimum requested value.
- M is an array of n maximum values, one per each of the n applicable requirements, that caps what is calculated by the linear formula. It is used to model logics where the linear formula is only applicable within a certain range: a typical example of this is statutory regulations, which often prescribe a linearly calculated minimum combined with a “but not more than” limit.

This same logic is then applied to each of the 5 dimensions of each design space, allowing us to obtain a target value for each of the dimensions of the compartment that hosts each space. The obtained target value will be used in the following paragraph to optimize the position of compartment boundaries.

4.5. Example of requirements definition capturing a statutory requirement

An example of how the requirements definition is used to capture statutory requirement is the application of the following MARPOL Annex I, Regulation 19, paragraph , concerning the width of wing tanks for oil tankers of 5000 tonnes in deadweight and above:

“Wing tanks or spaces shall extend either for the full depth of the ship's side or from the top of the double bottom to the uppermost deck, disregarding a rounded gunwale where fitted. They shall be arranged such that the cargo tanks are located inboard of the moulded line of the side shell plating nowhere less than the distance w , which, as shown in figure 1, is measured at any cross-section at right angles to the side shell, as specified below:

- $w = 0.5 + \text{DWT}/20,000$ (m) or
- $w = 2.0$ m, whichever is the lesser

The minimum value of $w = 1.0$ m”.

The rule will be applied to the breadth requirement for the space “double side” and, through the logics implemented by the algorithm it will directly affect the position of the longitudinal bulkhead delimiting the compartment to which this space is mapped.

To reproduce the complete logic, we will split it into two requirements:

- 1) The first requirement will set the overall minimum dimension of one metre:

$$\left\{ \begin{array}{l} C_{0,j} = 0 \text{ [m/ton dwt]} \\ U_{0,j} = 0.0 \text{ [m]} \\ Q_{1,j} = 1.0 \text{ [m]} \\ M_{1,j} = 1.0 \text{ [m]} \end{array} \right. \quad \text{for each } j \text{ in (list of driving quantities)}$$

- 2) The second requirement will model the linear formulation and the cap value:

$$\left\{ \begin{array}{l} C_{1,j} = 1/20000 \text{ [m/ton dwt]} \\ U_{1,j} = 1.0 \text{ [m]} \\ Q_{1,j} = 0.0 \text{ [m]} \\ M_{1,j} = 2.0 \text{ [m]} \end{array} \right.$$

for j = ordinal number of the DWT in the list of the driving quantities. $C, U, Q, M = 0$ elsewhere.

The target compartment breadth value is obtained by taking the maximum among the calculated values:

$$T = \max(R_0, R_1)$$

4.6. Measuring space requirements vs compartments KPIs fit

For each space and each metric, a fit function Φ compares the 5 main dimensions (KPIs) of the hosting compartment against what the space requires and applies weights for the two opposite conditions:

- the compartment supplies less than the space requires
- the compartment supplies more than the space requires

A compartment smaller than its space requirement means that it will be unable to fully accommodate its function. An extra space beyond what a space requires still has to be built, outfitted and transported for the life of the ship, therefore it will also receive a small penalization.

Around the requirement there is a range within which a space counts as comfortably served, and the fit function treats that range as the target. A bonus can be applied within that range to prize extra provision. The way the different logics of each space can be captured in the architecture and consumed by a universal algorithm is explained in the next paragraph.

4.7. The Cost Function

At the core of the optimization algorithm is a set of five linear functions that measure how each of the five compartment KPIs (length, breadth, height, area, volume) fits the target values of the hosted space. The coefficients of the linear functions (weights) are embedded in the definition of each space, and they capture the logics of how strongly the compartment shall match the space requirements.

The weights are assigned in the architecture to each space and each dimension: a double bottom height imposed by a statutory requirement is weighted differently from a store deck area, whose exact size is negotiable. The cost functions are built to ensure that, when a compartment provides exactly what a space requires, the resulting cost is zero. Hence, a design solution where all the requirements, per every space and every dimension are exactly met will score zero points.

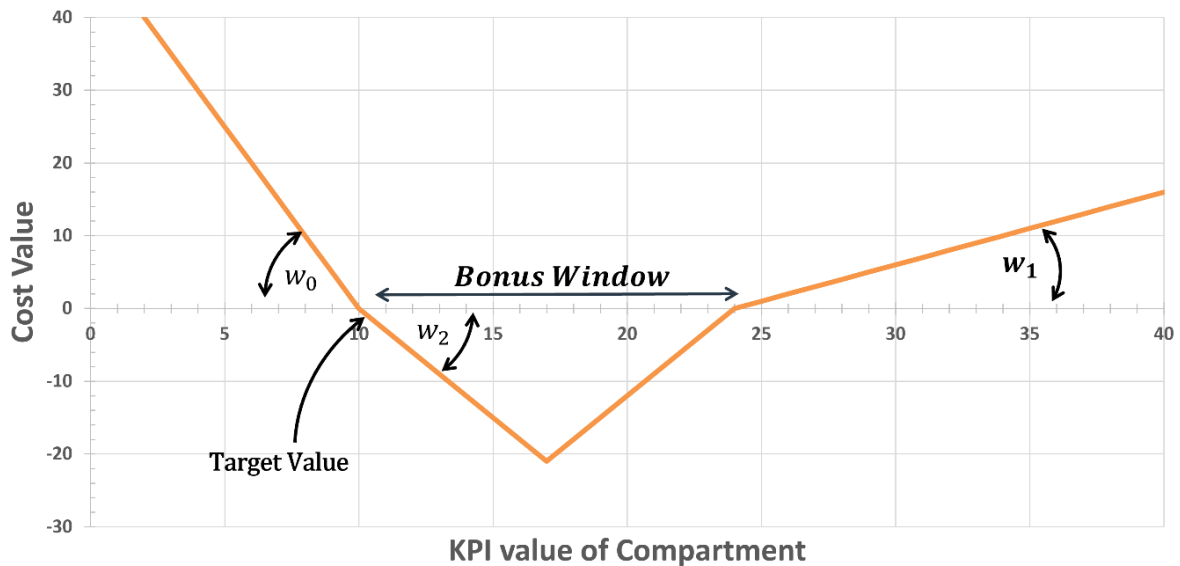


Figure 4. Example of a cost function for a single metric and a single functional space

In total a cost function requires four different user inputs: weights w_0 , w_1 , w_2 , and the bonus window b_w . The three weights control the slope of the cost function at different positions.

Weight w_0 influences the slope of the cost function before the target value. The higher the value of w_0 , the larger the penalty a sample will incur when the compartment dimension is too small compared to the corresponding space requirement. Weight w_1 determines the slope of the function for the dimension higher than the target value. It is used to penalize compartments that are overly large compared to what is required.

The bonus window (b_w) is used to model cases where an extra provision is considered a bonus and is a percentage of the target value. Examples of this are the fuel tanks or the accommodation spaces. Being able to carry more fuel than strictly necessary or having more space per passenger can both be beneficial compared to vessels that don't have this leeway. If a room requires at least 10 metres of length, and $b_w = 0.5$, the space will be rewarded with a negative cost when its dimension falls in the range of 10 to 15 metres.

Weight w_2 determines the slope of the part of the function within the bonus window, both when going down as well as back up to zero. Parameters b_w and w_2 can be leveraged by the designer to be able to reward vessels that exceed certain requirements. When b_w and w_2 are set to zero, the cost function will linearly descend, with slope w_0 , to the target value and then directly increase, with slope w_1 , away from the target value. When an architecture contains a space with a bonus window it is possible that a sample will receive a negative cost.

4.8. Example of Space Definition

An example problem involving a simplified space architecture layout with 4 compartments is presented in Figure 5,

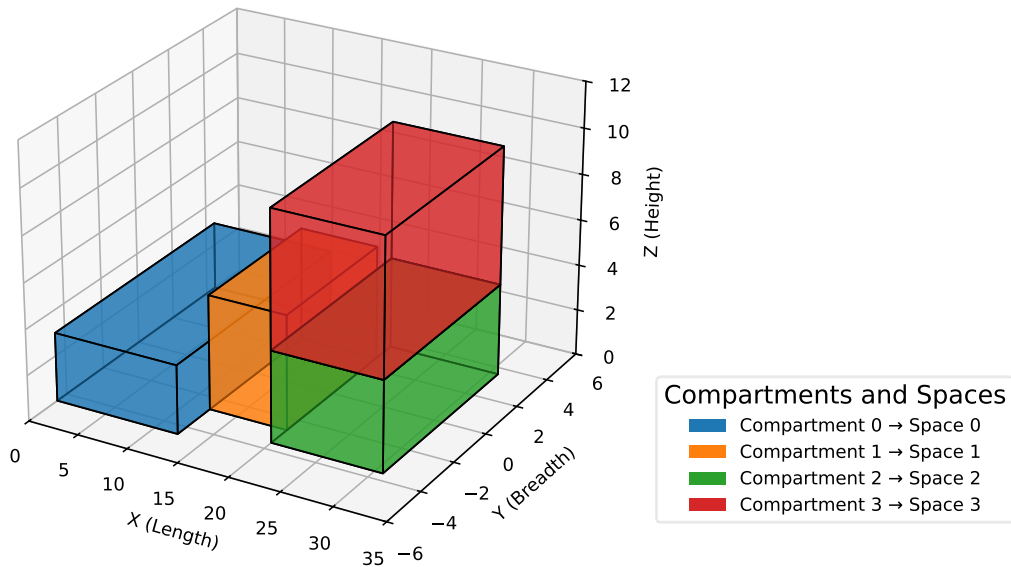


Figure 5. The 4 compartments enclosed within the boundaries used in the example case at the start of the optimization. The legend shows which space is linked to which compartment.

Table 2 provides a typical input for the space definition of these compartments: a space with fixed target dimensions of 7 meters in length and breadth and 10 meters in height is defined. Note that the height of this space has the highest weights for the cost function. The length has asymmetric weighting implying that if this space is too short it is penalized more heavily than being too long.

Table 2. Example of a space requirements inputs

Space ID:	0	Name:	Aft Peak	Host Comp:	0		
Inputs:		Length	Breadth	Height	Area	Volume	
Coefficients	$[C, U, Q, M]$	$[0, 0, 7, 1e9]$	$[0, 0, 7, 1e9]$	$[0, 0, 10, 1e9]$	$[0, 0, 0, 1e9]$	$[0, 0, 0, 1e9]$	
Weights:	$[w_0, w_1, w_2, E]$	$[3, 1, 0, 0]$	$[5, 5, 0, 0]$	$[10, 10, 0, 0]$	$[0, 0, 0, 0]$	$[0, 0, 0, 0]$	

5. The Space Allocation Algorithm

Section 0 introduced the structure of the architecture model used as input to the optimization process. This section will introduce the space allocation algorithm, built on that model.

A candidate is instantiated from the baseline architecture by mapping the baseline boundary positions onto its principal dimensions:

- Fixed boundaries, whose position is dictated by the candidate dimensions (transom, outer shell, bottom, main deck, the forward extremity), are pinned to those dimensions;
- the remainder are assigned an initial position proportional to their position on the baseline design and to the main dimensions of the candidate and become the free variables of the optimization.
- After positioning the boundaries, the compartment geometrical properties are calculated and the initial cost of for all spaces combined is calculated. This information is then brought to the optimization loop.

5.1. Variables and constraints

The independent variables are the positions of the free boundaries, one scalar each, on a discrete grid. The constraints to the range within which each boundary can be moved are imposed at the moment a move is proposed based on the position their neighbouring boundaries reached on the iteration before. Constraints enforce that bulkheads or decks cannot switch position compenetrating each other.

5.2. The coordinate descent loop

At every iteration, the optimization loop evaluates, one by one, the move of each free boundary by one grid step in each of its two directions. For each evaluation, the KPIs of the compartments that are impacted by the move and the cost of the spaces that are assigned to those compartments are recalculated and the resulting change in the fit function is evaluated. A schematic overview of the optimization process is shown in Figure 6.

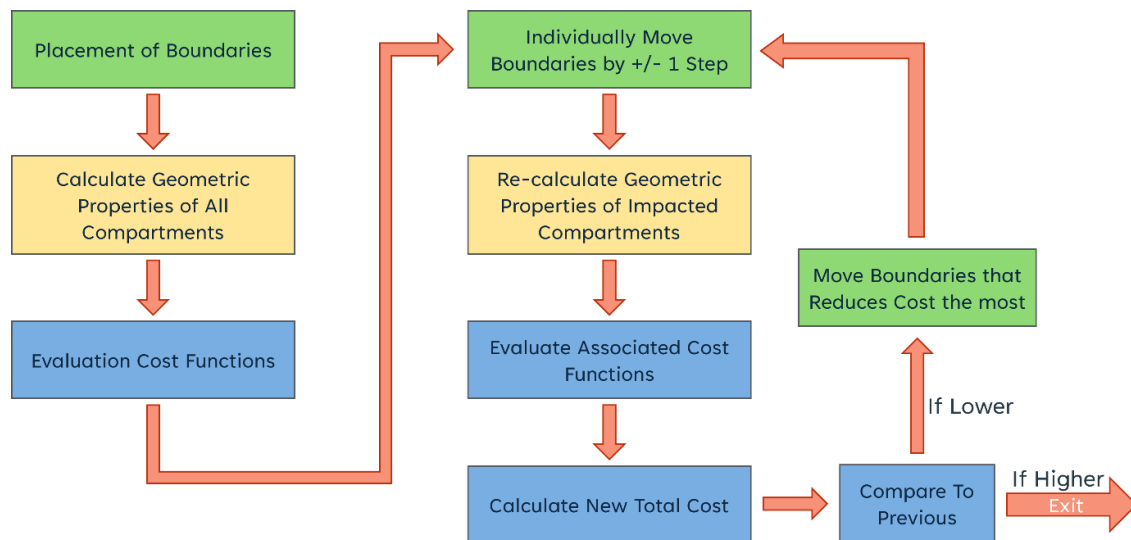


Figure 6. Optimization process for the space allocation algorithm.

The algorithm implements a discrete step coordinate descent optimization: the partial derivatives of the cost function are evaluated and a move along the steepest partial derivative is committed. This choice comes from the discrete nature of the space allocation problem, where the free variables (positions of the free boundaries) can only vary in discrete grid steps and the cost function is defined in segments. The optimization stops after the change in cost is lower than a user-defined minimum threshold.

5.3. Comparing different candidates

The residual value of the fit function at the end of the optimization will be carried along by each design candidate in the multi-objective optimization loop, representing how well that candidate accommodates the required functions inside its compartments.

- $\Phi = 0$: correspond to the case where, for the candidate being examined, every functional space is perfectly served, meaning that sufficient space is accommodated for the function carried by the space and no unused space is left.
- $\Phi > 0$: the sample carries an “handicap” signalling that one or more of the compartments did not achieve sufficient space to fulfil the requirements of its hosted functions, or, symmetrically, too much unused space is left in some compartments.
- $\Phi < 0$: the sample has a bonus, meaning that compartments are delivering some extra space that is exploited by the bonus windows: this corresponds to the case where the extra space onboard this design candidate is profitably exploited to transport more consumables or payload, or increase the comfort of the crew or passengers onboard.

6. Example of Space Optimization

In this section an example of a space allocation optimization is given for a simplified architecture consisting of 4 compartment and 4 spaces. For this simplified setup there is no vessel providing a hull shape to adhere to: it purely demonstrates the optimization capabilities to interpret and optimize a space architecture. The boundaries of the architecture are given in Table 3. The longitudinal bulkheads for this specific example are mirrored over the centre axis and therefor denoted as a pair. Some of the boundaries are fixed boundaries, meaning that they will not be moved during the optimization.

Table 3. Overview of the boundaries used in the example case.

Type	Boundary ID	Location [m]	Fixed
Transversal Bulkhead	[0, 1, 2, 3]	[0.0, 8.0, 13.0, 20.0]	[0, 3]
Longitudinal Bulkhead	[0/1, 2/3, 4/5]	[±5.0, ±3.0, ±4.0]	None
Deck	[0, 1, 2, 3, 4]	[0.0, 3.0, 5.0, 4.0, 10.0]	[0]

The boundaries in Table 3 as configured in their initial state. Since there is no actual hull shape to refer to, all the longitudinal bulkheads are allowed to move, giving more freedom to the algorithm than there would be in a real case. Typically the compartments that have the hull as one of, or both, their longitudinal bulkhead are constrained by those boundaries. The same holds for the decks apart from the bottom deck. Moreover, the compartments are allowed to become asymmetric in shape, even though they start symmetric over the centre axis.

The four spaces have varying target metrics: space 2 will rely on a fictitious driving quantity to calculate its target length and will employ a bonus window, allowing it to become longer than strictly necessary as that is considered a bonus in this example. The weights are applied in such a way that certain spaces are heavily bound on their height (space 2) or have strong penalties on being too small in terms of length or breadth. This way, the different dimensions are all tested within the optimization.

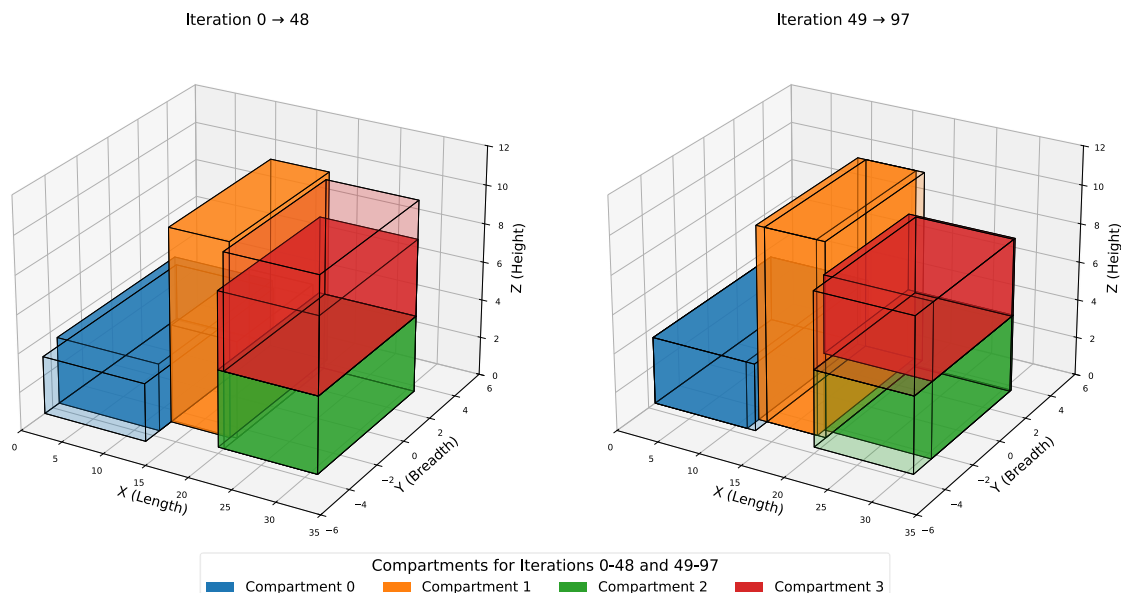


Figure 7. Change in compartments size over the first half (left) and second half of the iterations (right).

Figure 7 shows the change in compartment at two intermediate steps of the optimization. On the left hand side it shows the first half of the optimization process. The right hand side figure the second part of the optimization process. In both figures the sizes of the compartments at the start of the optimization iteration are shown shaded in the same figure.

The first 13 iterations consists of moving the top deck boundary of compartment 1 repeatedly, from its starting position directly close to its target value. It then briefly transitions to changing the width of compartment 0 after which it moves back to fine tune the height of compartment 1 again.

From iteration 0 to 48, compartment 1 has increased its height significantly. This was the most heavily weighted cost function in this example. Although compartment 2 was given a large weight on its length, to achieve that it had to move a boundary that influences other compartments resulting in lower cost reduction. The second change to notice is the reduction in size of compartment 3. This is achieved by lowering the deck 4, a move that only impacts compartment 3. The final change happens on compartment 0. This compartment is initialized being too wide and its width is reduced by moving one of the longitudinal boundaries.

From iterations 49 to 97, smaller changes are applied: while in the initial iterations large changes occurred on boundaries that are not shared, now it has transitioned to fine tuning the position of boundaries that are shared. One example of this are the changing of the bulkheads bounding compartment 1, which has minor adjustments on either side of it. Compartments 2 and 3 share the same longitudinal bulkhead as a boundary. For this specific example the weights were chosen such that compartment 3 is more heavily penalized for a discrepancy in its width than compartment 2. As a results the boundary is placed such that compartment 3 reaches its ideal width, at the cost of compartment 2.

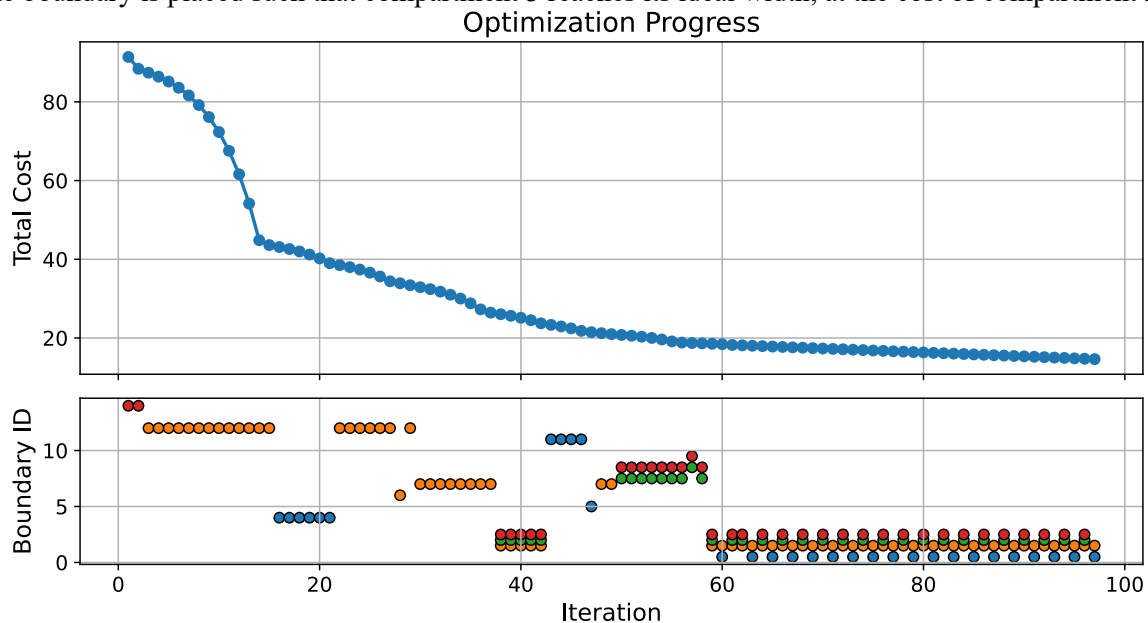


Figure 8. The top figure shows the evolution of cost over the iterations. The bottom figure denotes which boundary is moved. The dots are coloured by which space they affect. Multiple dots per iteration indicate more than one space is affected.

Figure 8 shows the evolution of the total cost over all the iterations and an overview of which boundary is moved at a given iteration. The dots are coloured according to which space the move influences. Multiple dots are drawn if a boundary move influences more than one space.

For this specific example the optimization is terminated whilst still having a non-zero cost. This is a result of the compartments sharing boundaries and the optimization having to trade off the ideal location that results in the lowest total aggregate cost.

Table 4 shows an overview of the residual for each metric for each space after the final iteration of the optimization process. There are a few interesting things to note. The spaces that have boundaries that are fixed, in this example these are all spaces except number 1, have relatively large residual in their length. This is a consequence of the constraint of having to fill the entire vessel with spaces, something which is mimicked here by fixing the first and last bulkhead.

Table 4. Overview of the size of spaces after the final iteration.

Space	Final Cost	Δ Length	Δ Breadth	Δ Height
0	+5.44	+5.30	-0.02	-0.02
1	+0.16	+0.02	+0.01	+0.01
2	-3.08	+2.47	-1.48	-1.70
3	+15.10	+7.48	+0.02	+0.00

Space 1, which is bounded between the other spaces and is not constrained on one of its boundaries is brought to all its target dimensions. Finally, it can be seen that space 2 is lengthened compared to its target value being brought to its ideal bonus value. As a result, space 2 also carries a negative cost. Placing this result in the bigger picture of a design space exploration, a smaller vessel would likely achieve an overall lower score. This would be due to the fact that the fixed fore and aft bulkhead would be closer to each other, reducing the length of compartments 0, 2 and 3. As can be seen in Table 4, almost all of the cost is accrued by not meeting the length target of the space.

7. Discussion and Conclusion

The final space allocation and residual value of the fit function depend on the weights assigned to the underlying cost function for each functional space. Changing the relative values of the w_0 , w_1 , w_2 for the different compartments produces different optimum solution and different trade-offs between functional spaces needs. This highlights one of the larger challenges when using this method to optimize the vessel internal space: calibrating the weights per each space and each dimension is crucial when setting up a new architecture and becomes increasingly demanding as the number of spaces grows. This challenge can be mitigated by creating of the aforementioned spaces database, which will capture the information from previously used spaces and architectures.

Tracing how requirements propagate through the space allocation process and contribute to the final score is essential in understanding the effect of the weights as well as essential for building this library and to create the type of knowledge-based decision making that this whole approach aims to generate. The coordinate descent optimization loop is a simple solution to the problem of searching, for each design candidate, an appropriate set of compartments boundaries positions. However, the limits of coordinate descent algorithms are known: i.e a more optimal solution cannot be identified when a negative gradient can only be obtained by moving two boundaries at the same time. The possibility of implementing a gradient descent in presence of cost functions with discontinued derivatives and that are only defined in discrete steps will be examined.

This work demonstrates how the problem of space allocation can be cast into unified topology using an RFLP framework and employed in a larger design space exploration framework. The proposed method uses a mathematical structure to translate different requirements driving vessel design into measurable metrics. Each metric that is compared for a given space is weighted using a cost function. The total cost accumulated by the vessel as well as the cost per space serves as a measure how well this space architecture will fit within the vessel. The functionality is demonstrated using an example case that shows how the proposed method is capable of minimizing the score for a small space architecture consisting of 4 separate spaces.

References

- ANDREWS, D.J. (1998), *A comprehensive methodology for the design of ships (and other complex systems)*, Proceedings: Mathematical, Physical and Engineering Sciences, pp.187-211
- ANDREWS, D.J. (2012), *Art and science in the design of physically large and complex systems*, Proceedings: Mathematical, Physical and Engineering Sciences, pp.891-912
- ANDREWS, D.J. (2018), *The sophistication of early stage design for complex vessels*, Int. J. Maritime Engineering 160
- CARLSON, C.M. (1987), *General arrangement design computer system and methodology*, Naval Eng. J. 99(3), pp.261-273
- KIM, K.S.; ROH, M.I. (2025), *Review of ship arrangement design using optimization methods*, J. Computational Design and Engineering, 12(1), pp.100-121
- LE POOLE, J.; DUCHATEAU, E.; HOPMAN, H.; KANA, A. (2022a), *Interactive Multi-Constrained System-To-Compartment Allocation To Support Real-Time Collaborative Complex Ship Layout Design Decision-Making*, Conf. Proc. INEC
- LE POOLE, J.; DUCHATEAU, E.; VAN OERS, B.; HOPMAN, H.; KANA, A.A. (2022b). *WARGEAR: 'Real time' generation of detailed layout plans of surface warships during early stage design*, Ocean Eng. 250, 110815
- PAPANIKOLAOU, A., BOULOUGOURIS, E.; ERIKSTAD, S.O.; HARRIES, S. (2024), *Ship Design in the Era of Digital Transition: A State-of-the-Art Report*, 15th IMDC Conf., Amsterdam
- PARSONS, M.G.; CHUNG, H.; NICK, E.; DANIELS, A.; LIU, S.; PATEL, J. (2008), *Intelligent ship arrangements: a new approach to general arrangement*, Naval Eng. J. 120(3), pp.51-65
- SHIELDS, C.P.; SYPNIEWSKI, M.J.; SINGER, D.J. (2018), *Characterizing general arrangements and distributed system configurations in early-stage ship design*, Ocean Eng. 163, pp.107-114
- VAN OERS, B.J.; STAPERSMA, D.; HOPMAN, J.J. (2010), *A 3D packing approach for the early stage configuration design of ships*, 9th COMPIT Conf., Gubbio, pp.367-381

Simulation Driven Approval

Jussi Puurula, RSD Software, jussi.puurula@rsd.fi

Juha Peippo, Rauma Marine Constructions, juha.peippo@rmcfinland.fi

Michael Pudd, Rapid Structural Design, michael.pudd@rsd.fi

Hans Van den Wijngaert, Siemens, hans.van_den_wijngaert@siemens.com

Abstract

Structural approval requires a sufficiently defined structure and a demonstration that it meets the applicable requirements. Drawing submissions convey this information through views, annotations, standard details and supporting calculations. Their visible level of detail, however, is not in itself a measure of the completeness of the approval evidence. This paper proposes simulation driven approval, in which structural objects, materials, connections, design loads, verification methods and results are explicitly connected. The same evidence supports the development of the design and the approval judgement. The central proposition is that this approach can improve the completeness and traceability of structural approval without drawing every production detail. Details needed for approval must still be defined, and any geometric simplification or use of a standard detail must be justified. A framework is proposed for evaluating information coverage, traceability, review effort and change handling. A focused deck framing case examines how global beam-element results can be combined with approval model information to derive flange stress and web shear stress accounting for openings.

1. Introduction

The transition from drawing-based to model-based structural approval, supported by the OCX Consortium and the Open Class 3D Exchange (OCX) standard, creates an opportunity to improve how compliance is demonstrated. OCX provides a basis for exchanging structural information, which can be connected to applicable requirements and calculation evidence to make the basis for acceptance clear and verifiable. This paper proposes simulation driven approval as a way to realise that opportunity, with the aim of improving the completeness of approval evidence and making it easier to review.

1.1. Shipyard requirements for design process efficiency

From the shipyard's perspective, design efficiency means delivering reliable design information in time for production planning, procurement and block fabrication. Pressure on project lead times increases the need for design material to be available earlier, sufficiently complete and supported by evidence of compliance. Short iteration cycles support design progress when each cycle provides reliable information for the next decision. Unresolved structural questions and inconsistencies between design representations can lead to changes after production planning, procurement or fabrication has begun, causing disruption and rework.

Meeting this need requires coordination between disciplines and a shared design basis that is kept up to date as the design develops. The shipyard seeks to reuse information across design, calculations and review, with clear identification of the applicable revision and the matters still to be resolved. Information exchange with owners, classification societies and suppliers should support this common understanding. The wider objective is continuity of information from concept design through production and lifecycle activities, with the level of detail developing according to the decisions and tasks it must support.

Computational verification should therefore form part of normal design work. Model-based approval, automated checks and digital traceability can support the intended process by connecting review questions and verification evidence to the developing design. This work should produce approval-

ready material, with structural definition, design basis and supporting checks available for review. Earlier resolution of design questions, less repeated preparation of information and more predictable progression towards production are the benefits sought by the shipyard.



Fig.1: Shipyard objectives for design process efficiency and the capabilities supporting them

1.2. Focus of the present work

A structural approval decision depends on knowing what has been designed, the conditions for which it has been designed and the evidence supporting its compliance. In a drawing-based approval process, these answers are assembled from structural plans, annotations, detail references, design data and calculations. The reviewer must be able to determine which calculations support each part of the submitted structure and whether they apply to the design revision and loading conditions under review. Without systematic connections between these materials, the reviewer cannot readily establish the coverage and applicability of the verification evidence.

Building on the connected analysis and model-based approval approaches presented in our earlier work, *Puurula et al. (2024)*, *Peippo et al. (2025)*, we propose simulation driven approval as a process that connects OCX-based structural information, applicable requirements and calculations to provide a traceable basis for structural approval. The present paper focuses on how this connected information meets structural approval needs and makes the basis for acceptance clear and verifiable. For each structural member, the reviewer can examine what is required, how it has been checked and the basis for accepting the result. Our proposition is that making these relationships explicit can improve the completeness of approval evidence and make it easier to review.

The paper focuses on structural strength and dimensioning within design approval. This includes the structural arrangement, load response, member capacity and detail requirements relevant to structural performance. It does not address the full scope of classification or production documentation.

In this paper, the approval model means the structural definition linked to its design basis, requirements and verification evidence. An analysis model is the representation used for a particular calculation and may simplify the geometry while retaining the behaviour relevant to that check.

The paper first identifies the information needed for structural approval and distinguishes it from the detailed geometry required for production. It then describes how simulation driven approval connects this information to verification. Global structural analysis and connected review workflows are

developed with reference to our earlier work, *Puurula et al. (2024)*, *Peippo et al. (2025)*. The new deck framing case extends this approach to detail verification by combining global beam-element forces and moments with approval model information to support structural compliance checks. The case illustrates how the information needed for verification can remain available without reproducing the full production geometry in the analysis model.

2. What structural approval requires

Structural approval requires evidence that the vessel's structure meets the applicable strength and integrity requirements. We reviewed the classification rules and guidance referenced below to identify the information required for this assessment. The review covered *DNV (2024)*, DNV-CG-0550, Section 5, H050–H080; *Bureau Veritas (2026)*, NR467, Part B, Chapter 1, Section 4; and *Lloyd's Register (2026)*, LR-RU-001, Part 3, Chapter 1, Sections 2 and 5. The review also included *DNV's (2021)* guidance on finite element analysis and *Lloyd's Register's (2021)* guidance on primary structures of naval ships under Category NS2.

Table I groups the findings into five information needs covering the structural definition, design basis and evidence of compliance. This grouping provides the baseline used in this paper to assess how a model-based approach can meet approval needs. It does not replace the submission requirements of individual classification societies.

Table I: Five information needs forming the baseline for structural approval

Need	What must be identifiable and checkable
1 Structural arrangement	The location, geometry and arrangement of the structural members, including the extent of the structure submitted for review.
2 Scantlings and materials	Plate thicknesses, member sections, spacing and material specifications, associated with the correct structural members.
3 Connections and details	How members connect; relevant welds, openings, reinforcements and discontinuities; and the locations to which referenced standard details apply.
4 Design basis	Applicable design conditions, class notations, loads and load cases, together with the assumptions needed to interpret the structural checks.
5 Evidence of compliance	Required rule checks and calculations, including the definition, assumptions, loads and boundary conditions of direct analyses and the criteria used to judge their results.

The five information needs differ in how they are typically communicated. Structural arrangement, scantlings and materials (1–2) are readily presented in structural plans or models. Connections and details (3) are often defined through typical arrangements and standard-detail references. Drawing or modelling every instance of a detail is neither practical nor necessary, provided that it is clear which detail applies at each location.

The design basis (4) establishes what the structure is designed to withstand; the verification evidence (5) demonstrates that it meets the applicable requirements. In a fragmented drawing-based submission, these are documented in separate reports and calculations without systematic links to the structural items under review. Such a submission does not comprehensively demonstrate which requirements have been checked, for which members and under which conditions. The calculations may exist, but their coverage and applicability remain insufficiently demonstrated to the reviewer.

A detailed drawing or model can therefore communicate the structure clearly while leaving the basis for its acceptance only partly demonstrated. Where supporting calculations are not submitted or their

applicability is unclear, the reviewer cannot readily verify the basis for acceptance and may need further information.

As illustrated in Fig.2, the five information needs must be connected to the same structure, design revision and design conditions so that the reviewer can establish the basis for acceptance.

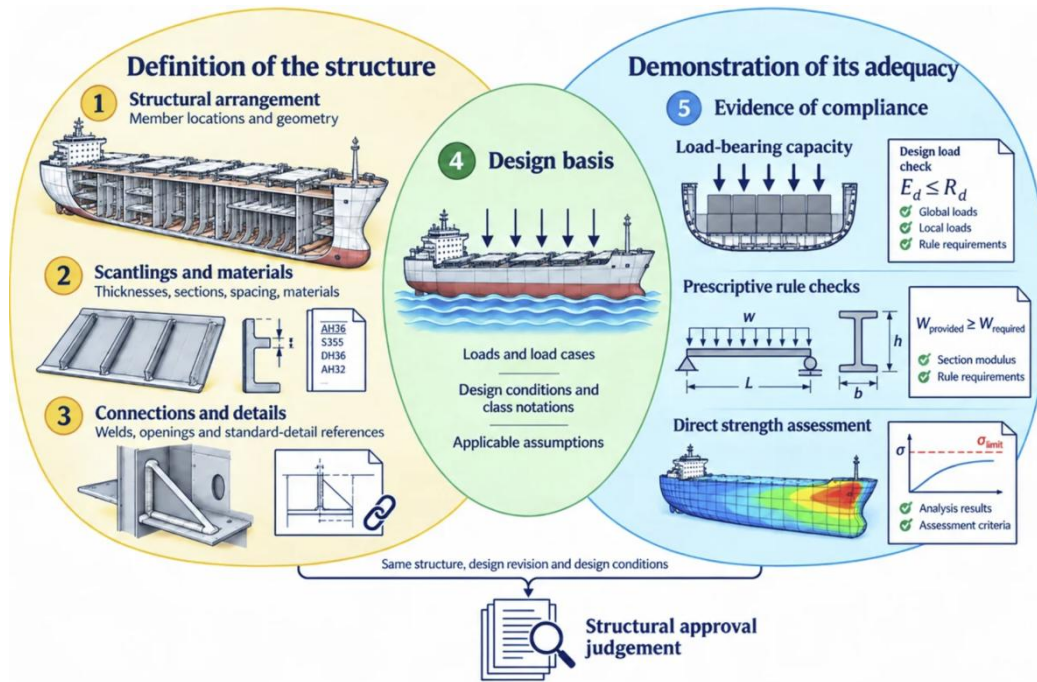


Fig.2: Five information needs for structural approval

2.1. Approval detail and production detail

We consider that structural approval should not require every detail to be explicitly represented in a drawing or model, provided that sufficient information is available to verify the structural solution. A detail can be defined by its governing dimensions, properties and connection principles, with calculations demonstrating that it meets the applicable requirements. This distinction builds *Peippo et al. (2025)*, who proposed an essential structural model supported by calculations as the basis for the Strength Gate, before completion of production detailing.

A common practice is to use a typical-details drawing that defines connection principles and indicates where different detail types apply. Each occurrence need not be drawn separately when the applicable detail and its governing parameters are clear. *DNV (2024)* recognises standard details identified on structural drawings in DNV-CG-0550 H070. This illustrates how the information needed for verification can be provided without reproducing the full geometry at every location.

Detailed geometric modelling serves the shipyard's fabrication and assembly needs. *Okumoto et al. (2009)* distinguish the approval-oriented key plan from the yard plan, which adds information such as assembly details, welding schemes, fitting details and piece codes. These production needs do not, by themselves, make detailed production geometry a prerequisite for approval.

The distinction depends on the role of the information. Dimensions, weld specifications and other properties that govern structural performance must be sufficiently defined for verification, even where their full geometry is not represented. The subsequent production design must implement the verified solution within the parameters and assumptions on which its acceptance is based. Changes outside that basis require reassessment of the affected checks.

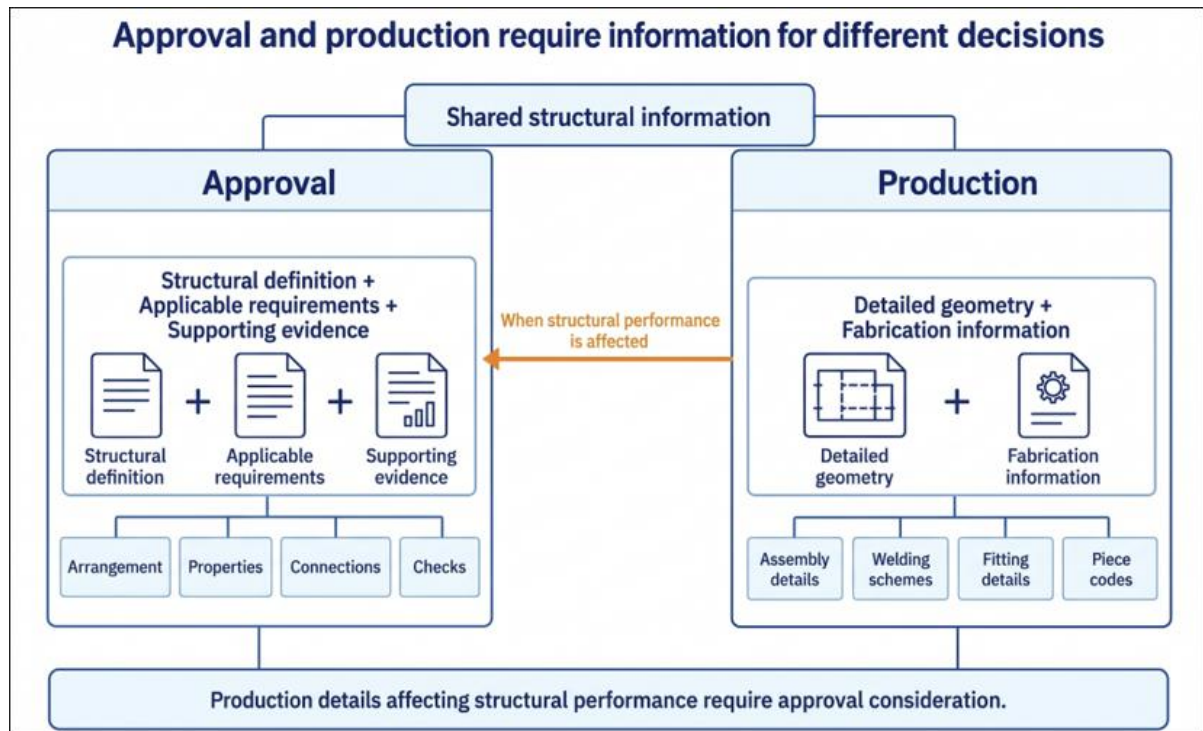


Fig.3: Approval requires verification evidence; production requires fabrication detail

3. Simulation driven approval

Simulation driven approval connects the structural information needed for approval with the requirements and calculations used to verify it. The same connected information supports design development and approval review. As the structure develops, the designer can assess its adequacy; during review, the basis for that assessment remains available for examination.

3.1. Connecting structure, requirements and verification

In the proposed approach, OCX-based structural information is connected to the applicable requirements, design conditions and verification calculations. Each structural member has a calculation trace: a linked record of the member being checked, the requirement, input data, assumptions, load case, calculation method and result assessed against the acceptance criteria. Missing inputs and unresolved checks are visible alongside completed checks.

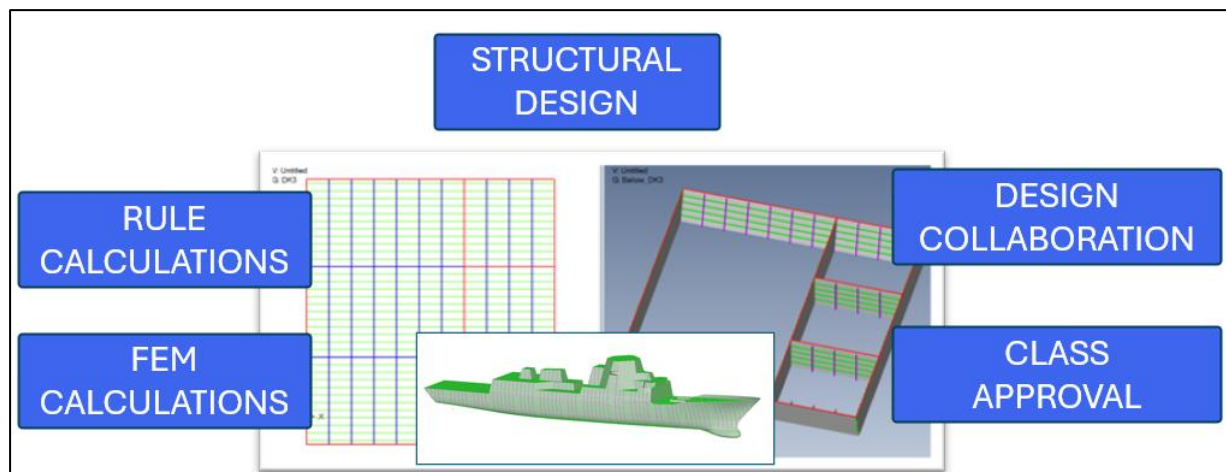


Fig.4: A shared structural model supporting design, calculations and approval, *Peippo et al. (2025)*

This connection addresses the gap identified in Section 2. The reviewer can follow the calculation trace from a structural member to the evidence supporting its acceptance. The trace also makes it possible to examine verification coverage: which requirements have been addressed, for which members and under which conditions. The relationships established during design remain available throughout approval.

An analysis may use a simplified representation, while the structural properties and detail parameters needed for verification remain available through the approval model. The adequacy of that representation depends on the checks it supports.

The information does not need to be stored in one file. Standard details, load definitions and calculation results may remain in separate resources, provided that their revisions and relationships to the relevant structural members are explicit and accessible. Together, they provide a clear and verifiable basis for design and approval.

3.2. Global structural analysis and dimensioning

Global structural analysis provides the basis for dimensioning structural members under the relevant load cases. It captures how loads are transferred through the vessel and how the response of each member depends on the surrounding structure. Connecting this analysis to the structural definition allows member dimensions to be assessed as part of the load-carrying system.

In simulation driven approval, the structural arrangement, member properties and design conditions provide the inputs for the global analysis. The analysis representation is selected to capture the behaviour needed for the intended checks. Plates, beams and equivalent properties can be used where their assumptions are justified. The objective is to obtain reliable structural responses for dimensioning; reproducing every production feature is not a prerequisite.

The global results are then connected back to the corresponding structural members. Member forces, moments and stresses are combined with section properties, material data and applicable requirements to evaluate structural adequacy. Each check retains its load case, calculation method, assumptions and acceptance criterion. This creates a calculation trace from the defined structure and loading to the result used to justify its dimensions. Required checks that are not covered by the global analysis remain identifiable.

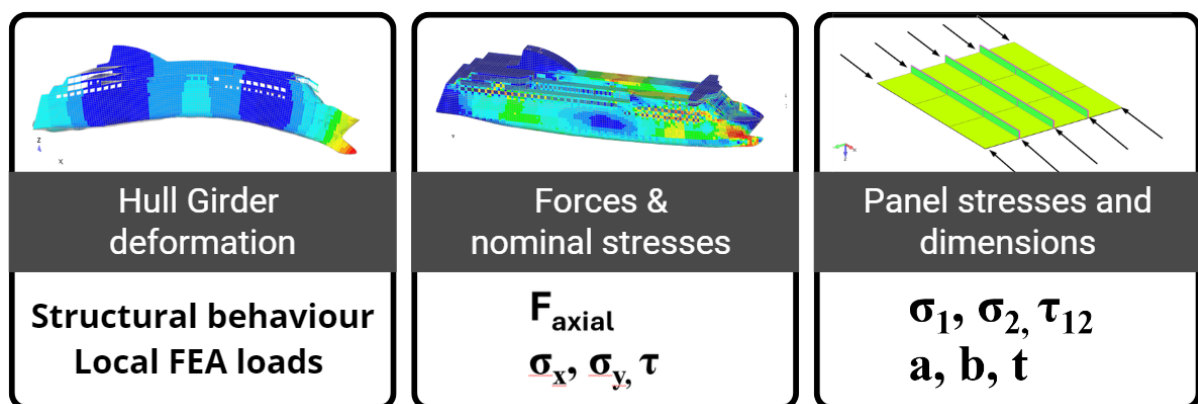


Fig.5: Global analysis results supporting structural verification, *Puurula et al. (2024)*

The same calculation trace serves both design and approval. During design, it helps identify governing load cases, insufficient capacity and opportunities to adjust scantlings. During approval, the reviewer can examine the basis for those choices and the coverage of the supporting checks. When the structure changes, the calculation trace helps identify which analyses and checks need to be updated.

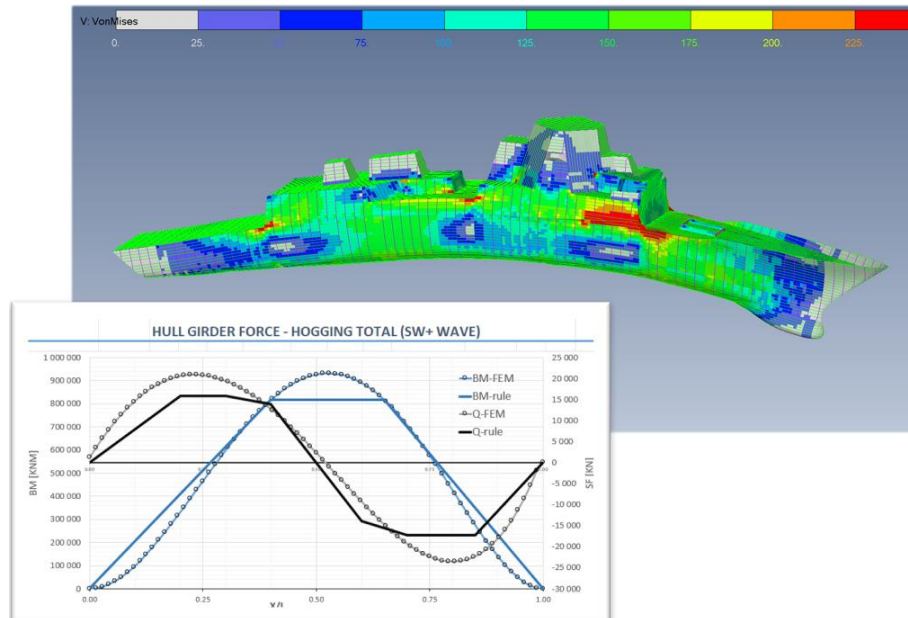


Fig.6: Global FE response and comparison with rule-based hull-girder loads, *Peippo et al. (2025)*

Our earlier publications provide the foundation for this global analysis and connected review workflow, *Puurula et al. (2024)*, *Peippo et al. (2025)*. The present paper builds on that work by explaining how the connected information addresses the approval needs identified in Section 2. The new case extends the calculation trace to detail verification: global member results are combined with the additional structural information needed to assess details that are not explicitly represented in the global analysis model.

3.3. Extending verification to details

Global analysis provides structural responses that can also support detail verification. The additional information needed for a particular check can be obtained from the connected structural definition. An opening, for example, may be described by its dimensions and position, while a connection may be defined by its type, member properties and weld specification. These parameters can support a calculation without requiring the full detail geometry to be drawn or modelled.

The calculation trace connects the global response, the relevant detail parameters and the applicable requirement. For a member containing an opening, this may involve combining the member force with the remaining effective section to assess compliance with the applicable acceptance criterion. For a connection, the transferred forces can be assessed against the capacity of the defined connection. In each case, the reviewer can examine how the detail has been accounted for and which assumptions govern the result.

The verification method must address the behaviour relevant to the detail. Depending on the question, this may be a prescriptive rule check, an engineering calculation, an applicable standard solution or a local finite-element analysis. Explicit geometric modelling is needed where the selected assessment requires it. Where a justified calculation can account for the relevant effects through parameters, a detailed geometric representation is not required for that check.

Two aspects of the calculation chain must therefore be established: the global model must provide suitable member responses, and the subsequent detail calculation must adequately represent the effects being assessed. If a detail changes the stiffness or load path sufficiently to affect the global response, that influence must also be included in the analysis. Effects outside the scope of the selected calculation remain subject to additional checks.

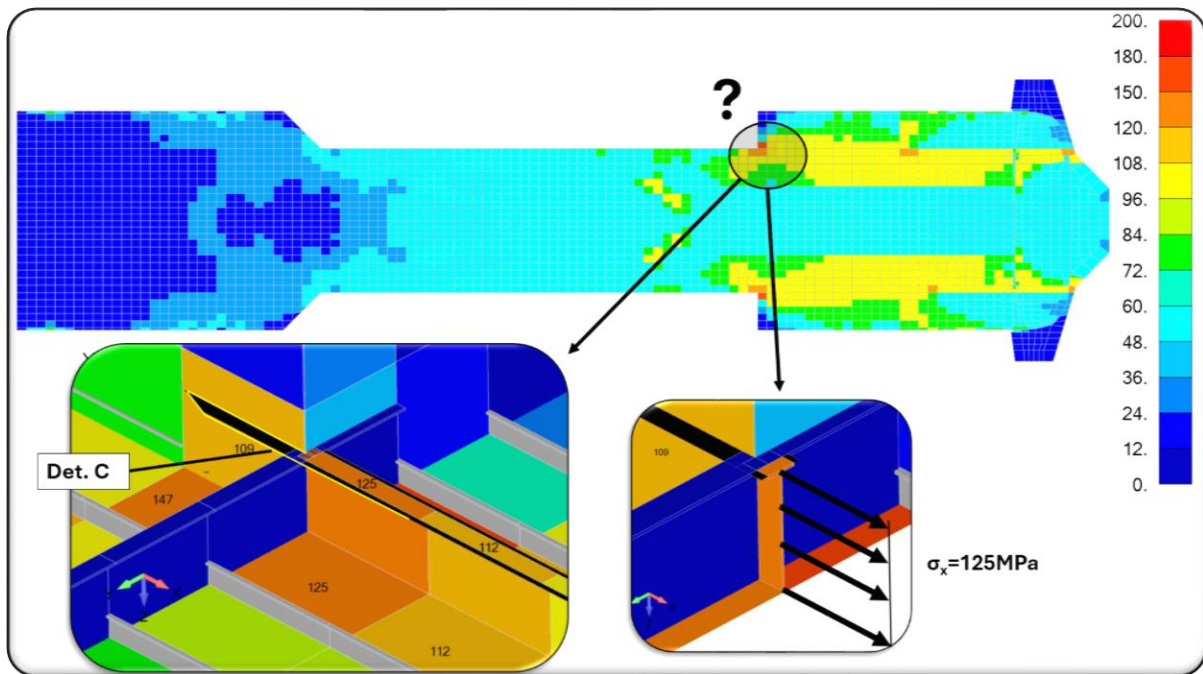


Fig.7: Global stresses linked to structural details for further assessment, *Puurula et al. (2024)*

The deck framing case in Section 4.3 demonstrates this extension using global beam-element results together with section and opening information. The case examines selected flange and web stress checks and compares them with a detailed finite-element representation. Its purpose is to show how a traceable calculation can account for structural details without reproducing their full geometry in the global model.

3.4. Review and change handling

The connected calculation trace provides a direct starting point for approval review. From a structural member, the reviewer can access the applicable requirements, design conditions, calculation assumptions and results. The reviewer can examine both the outcome of an individual check and the extent to which the required verification has been completed. Missing information, unresolved checks and results requiring revision remain visible.

This changes how clarification can be handled. A review question can refer to the specific member, requirement or calculation step concerned. The designer can address that question within the same trace, preserving the connection between the issue, the supporting information and its resolution. This can reduce repeated exchanges of supplementary reports whose relevance the reviewer must otherwise reconstruct. The evidence must remain tied to the design revision it represents. A change in scantlings, material, loading or connection definition may invalidate earlier results. The affected checks are identified through their dependencies, repeated where necessary and associated with the revised structure. Changes affecting stiffness or load transfer may also require the global analysis to be updated, including checks on members beyond the modified location.

Production detailing follows the same principle. Adding production details does not invalidate the verification provided they remain within the verified parameters and assumptions. A change that alters the load path, effective section, connection capacity or another verification assumption requires reassessment of the affected checks. Approval therefore refers to an identifiable structural definition and its supporting evidence. The connected information makes the basis for the decision available for review and records how review questions and design changes have been addressed. The classification society retains responsibility for the approval judgement.

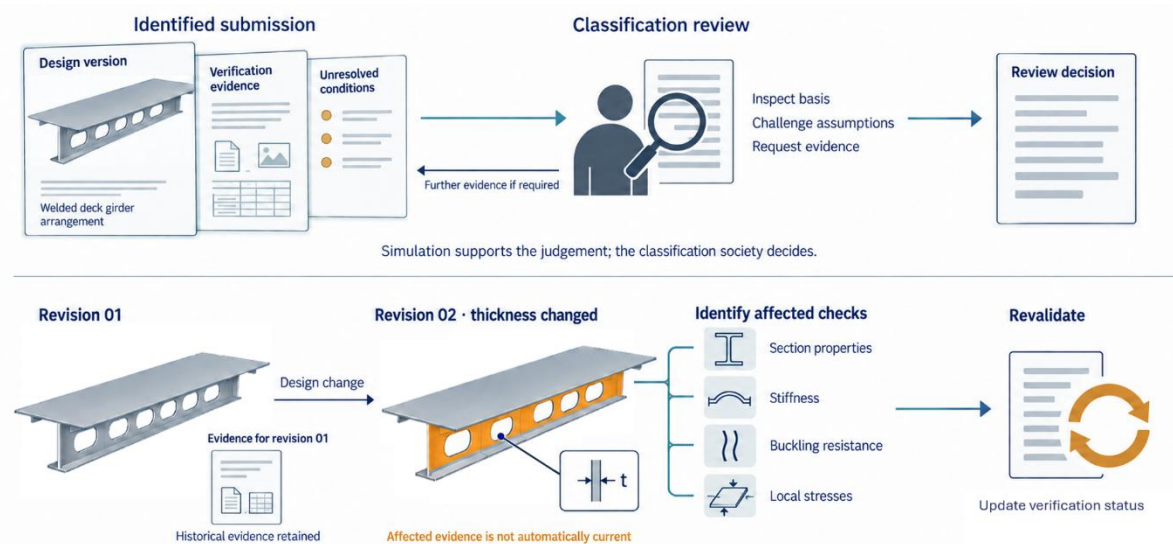


Fig.8: Connected review and revalidation following a design change

4. Demonstration and evaluation

The earlier studies provide practical examples of global structural analysis and connected verification workflows, *Puurula et al. (2024)*, *Peippo et al. (2025)*. Here, they are considered against the approval information needs identified in Section 2. They establish the foundation for the new deck framing case, which extends the approach to calculations that combine global member results with detail information. The evaluation framework distinguishes what these examples demonstrate from the broader benefits to be assessed in approval practice.

4.1. Connected structural design, analysis and review

Puurula et al. (2024) used a fictitious 100 m RoPax vessel to demonstrate model-based FEA reporting. The integration maintained the relationship between structural panels in the OCX model and their corresponding FE entities. Differences deliberately introduced between the models could be detected, stresses could be viewed in their structural context, and the information required for buckling assessment could be extracted. The reviewer could therefore examine analysis evidence in the context of the structure to which it applied.

Peippo et al. (2025) extended this work towards a unified model supporting structural design, calculations and approval collaboration. A fictitious 135 m naval destroyer was defined through an OCX model containing plate and stiffener locations, scantlings and spans. The model was reused in DNV Nauticus Hull for prescriptive calculations and as the basis for automated FE meshing and global analysis. This demonstrated how the same structural information could support both design development and verification. The study described design progression from continuous load paths to compliant member scantlings and structural details. Global and local load effects are to be assessed in combination against the applicable strength criteria. The proposed Strength Gate addressed the structural arrangement and main scantlings, followed by a Detail Gate aligned with the production schedule. Global assessment thus provided a basis for developing the design before all details had been completed. *Peippo et al. (2025)* focused on the early stages of structural design, using a shared model to support structural layout, scantling calculations and global strength assessment. The present paper builds on that foundation by proposing an approval approach in which structural parameters and connected calculation evidence can support verification without a fully detailed geometric model. It organises approval information into five needs and examines how their connections make verification coverage and the basis for acceptance clear and verifiable.

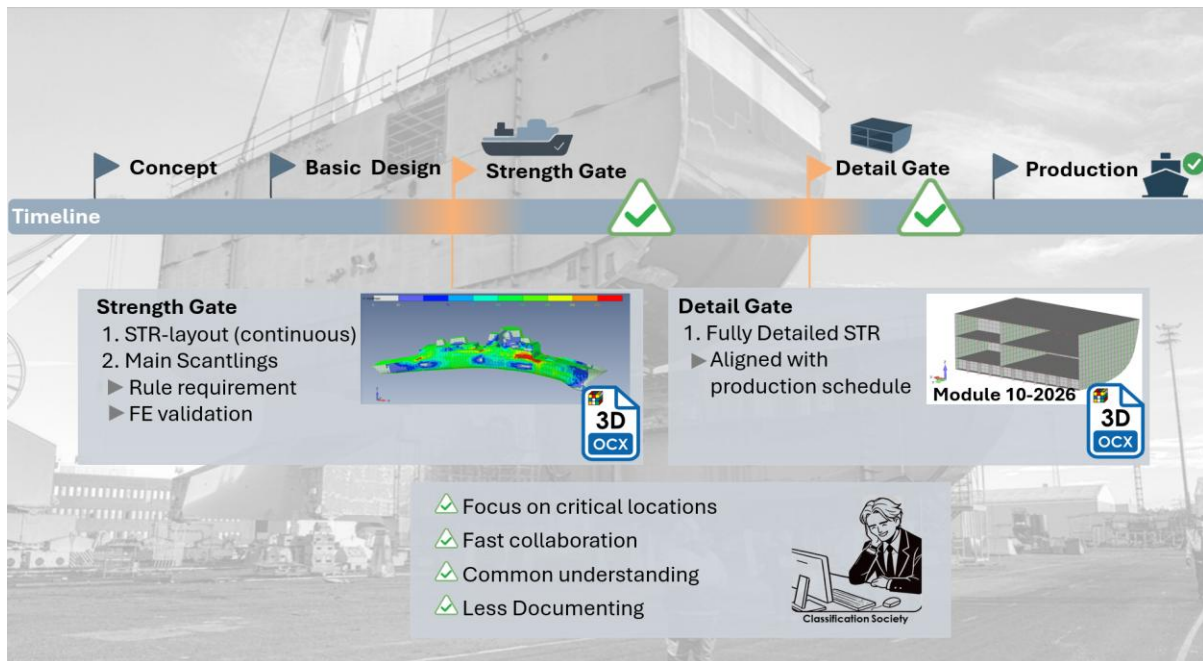


Fig.9: The staged approval concept presented in *Peippo et al. (2025)*

Together, the studies provide the foundation for the connected approach described in Section 3. The new case in Section 4.3 develops the detail-verification part of this approach: global beam-element results are combined with structural parameters to assess a detail without explicitly reproducing its geometry in the global model.

4.2. Basis for evaluation

The central proposition concerns the completeness of approval evidence and how readily it can be reviewed. The evaluation therefore considers whether the information needed for a check is available, whether the calculation applies to the structure being assessed, and whether the basis for the result can be followed. The five information needs in Table I provide the baseline.

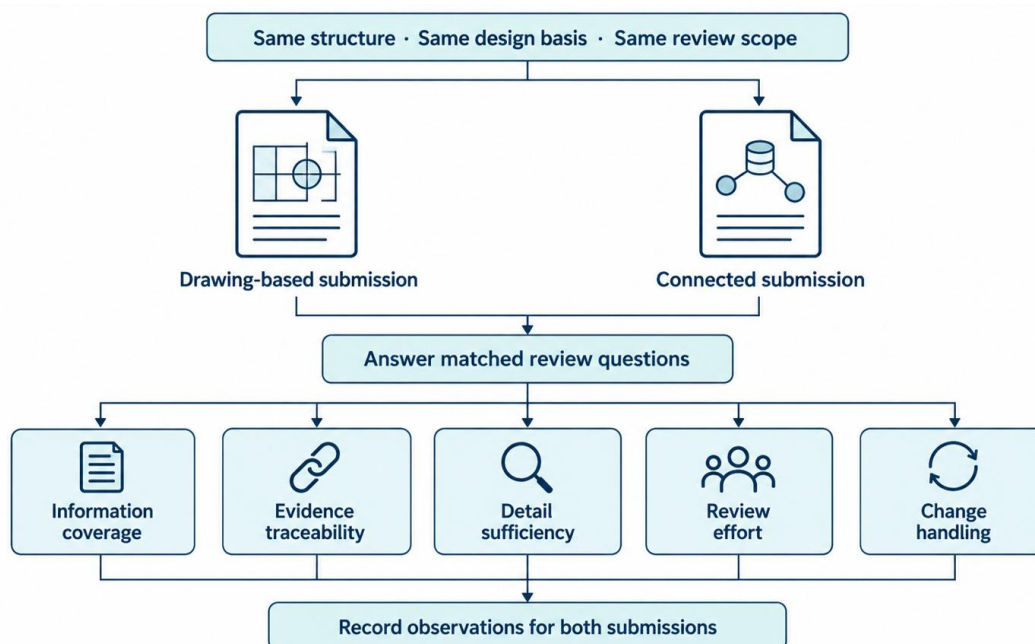


Fig.10: Proposed framework for a future comparison of drawing-based and connected approval submissions

Table II.: What to assess

Criterion	Observation to record in both submissions
Information coverage	Whether every approval-relevant item in the agreed scope is present, identifiable and sufficiently defined; record omissions and unresolved assumptions.
Evidence traceability	Whether the reviewer can trace a selected result to the correct structure, properties, load case, assumptions, criterion and revision.
Detail sufficiency	Whether a connection can be evaluated without showing each occurrence in detail drawing; record missing parameters, applicability questions and additional evidence requested.
Review effort	Time and actions needed to answer matched review questions, including cross-document searches and clarification requests.
Change handling	Whether the workflow identifies all checks requiring reassessment after a design change, compared against an independently prepared reference list, and clearly marks previous results that have not been revalidated for the revised design.

The earlier studies provide practical examples of connected structural design, analysis and review. The deck framing case in Section 4.3 examines how global analysis results and structural information can be combined to assess a detail. The comparison with a detailed finite-element representation supports our assessment of the calculation's adequacy for the checks presented. The case provides a practical example for evaluation rather than a comparative assessment of approval workflows. The proposed framework offers a basis for designers, shipyards and classification societies to evaluate the approach together.

4.3. Deck-framing case: verification using a simplified analysis model

The case is based on the assessment of primary members supporting decks described in SDA for Primary Structure of Passenger Ships, Chapter 5, by *Lloyd's Register (2019)*. Section 1 identifies the need for independent strength analysis where, among other conditions, web openings exceed 25% of the web depth. Section 2 specifies a detailed FE representation with explicitly modelled openings. Using this assessment framework, the case compares flange and nominal web shear stresses from a detailed shell model with those derived from global beam-element results and connected structural information under the same deck load.

The approval model contains structural arrangement, scantlings and materials, connection and detail information, and the design basis. These correspond to the first four approval information needs identified in Section 2. Connecting this information to analysis results and acceptance criteria provides the basis for the fifth need: evidence of compliance.

In the proposed approach, member forces and moments are obtained from the whole-vessel global model. For this comparison, the selected deck region was extracted into a beam-based submodel corresponding to the detailed local shell model, with the surrounding structure represented by boundary conditions. This allowed the two representations to be compared under equivalent loading and boundary conditions.

FE analyses using a beam-based submodel and a detailed shell model were performed for the same deck load case. Both models were prepared and post-processed in Simcenter Femap, with the analyses performed using Simcenter Nastran. Femap was selected for its customisation capabilities, which supported adapting the modelling and results-processing workflow needed for this study.

V: Untitled
G: COMPIT-CASE

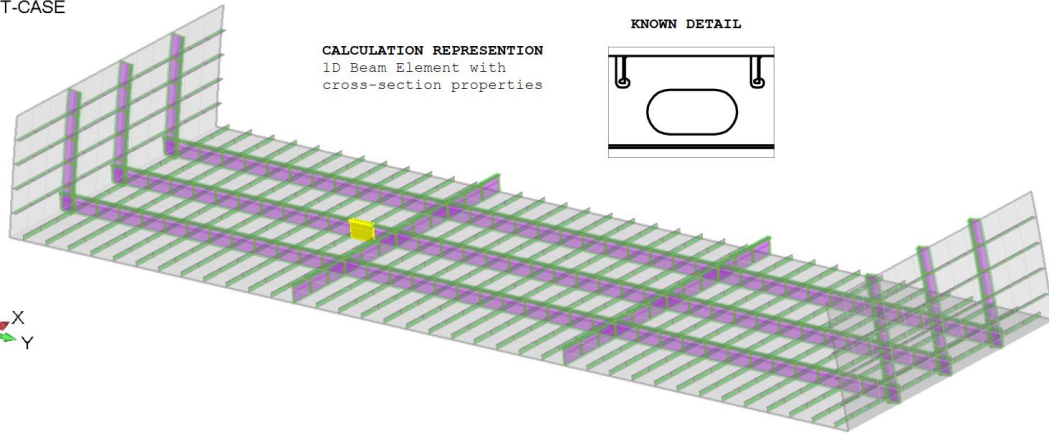


Fig.11: Deck region extracted from the global beam-based FE model

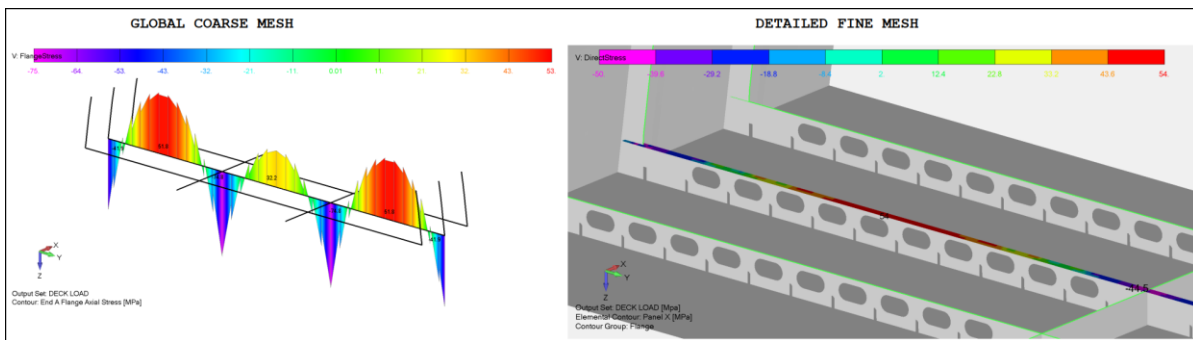


Fig.12: Flange stresses under the deck load: global beam model (left) and detailed shell model (right).

The global beam model gives a shear-force magnitude of 31067 N at the selected location. The opening leaves a total remaining web height of 200 mm; with a web thickness of 7 mm, the remaining web area is 1400 mm². Dividing the shear force by this area gives a nominal shear stress of 22.2 MPa.

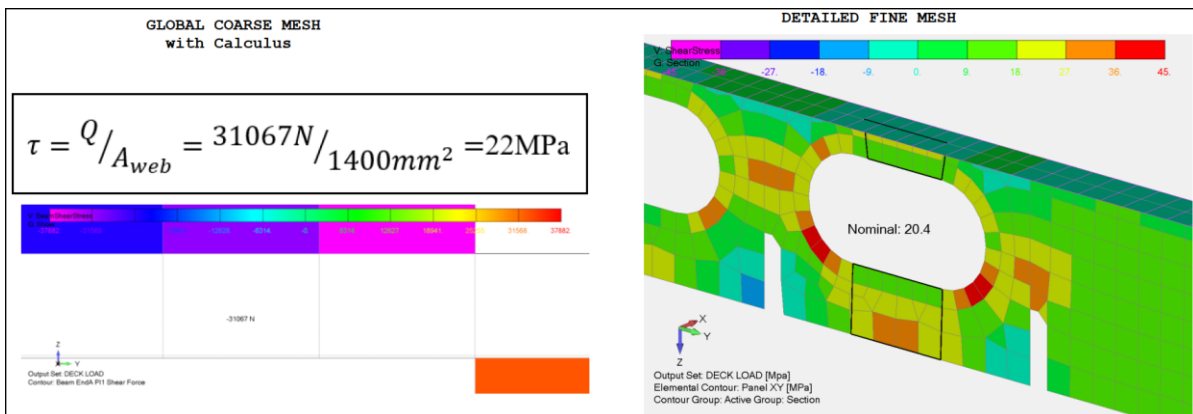


Fig.13: Nominal web shear assessment at a girder opening: calculation from global beam results (left) and detailed shell results (right)

For Grade A steel ($\sigma_0 = 235$ MPa), rule stress criteria are $0.52\sigma_0$ for average direct stress in the face plate and $0.35\sigma_0$ for average shear stress and $0.75\sigma_0$ for average von Mises stress in the web plating, *Lloyd's Register (2019)*, Chapter 5, Section 4, Table 5.4.1.

All reported stresses in Table III are below the selected limits. The beam-based calculations show close agreement with the shell results for mid-span flange stress and nominal web shear stress, while giving a higher compressive stress magnitude at the intersection. The comparison supports the proposed calculation approach for the responses and load case examined.

Table III: Stress comparison and utilisation against the selected criteria

Comparison	Beam-based result (MPa)	Shell result (MPa)	Allowable (MPa)	Beam-based utilisation	Shell utilisation	Interpretation
Flange stress at mid-span	51.8	54	122	42 %	44 %	Close agreement
Flange stress at the girder intersection	-74.8	-44.5	122	61 %	36 %	Higher compressive stress magnitude in the beam model
Nominal web shear stress	22.2	20.4	82	27 %	25 %	Close agreement
Web von Mises stress	-	49	176	-	28 %	Similar utilisation to the beam-based nominal shear check (28% versus

In the proposed approval submission, the OCX-based structural definition would be linked to the design basis and member-specific calculation traces. For the girder opening considered here, the trace would identify the member and design revision, the applicable requirement and load case, the global shear force, the opening dimensions and remaining web area, and the calculation leading to the assessed stress. The acceptance criterion and resulting utilisation would complete the calculation trace. The underlying analysis model and its assumptions would remain available for examination.

5. Discussion

The central proposition is that explicit connections between structural information, requirements and calculations can improve the completeness of approval evidence and make it easier to review. A drawing or model can communicate the structure clearly while leaving the coverage and applicability of its verification difficult to establish. Simulation driven approval addresses this gap by retaining the relationship between each structural member, the conditions for which it is assessed and the evidence supporting its acceptance.

Our earlier studies provide the foundation for this approach. The present paper develops their connected workflows into a clear description of the information needed for approval. The deck framing case contributes a technical example: the information required for a selected assessment can be assembled from global analysis results and structural parameters without reproducing every detail in the analysis geometry.

5.1. Completeness of evidence and effectiveness of review

In conventional FE reporting, engineering judgement is used to select the governing or otherwise representative locations for detailed presentation. Their results, together with the rationale for their selection, support the assessment of the remaining structure. Although the analysis may produce results throughout the model, the submitted report presents a selection of that evidence.

The proposed approach would make results and calculation traces available as data for every member within the assessment scope. The reviewer could see the verification status across the structure and select individual calculations for spot-checking or closer examination. Selection would determine which results are examined in detail, while the evidence for all assessed members would remain available. This provides a lightweight but comprehensive form of reporting within the defined scope. Each result is connected to the relevant member, design revision, requirement, load case and calculation assumptions. The reviewer can examine the basis for a satisfactory result and identify missing, unresolved or outdated checks. Providing this coverage as structured data avoids preparing a separate written account of every calculation.

The connection also gives clarification requests a precise subject. Instead of requesting additional report extracts to establish the basis for a result, the reviewer can refer directly to the member and calculation concerned. The designer can address the question within the same trace, preserving the relationship between the issue, the evidence and its resolution.

The expected benefits are more complete approval evidence, fewer clarification cycles and less effort spent compiling and maintaining reports. The same information supports design decisions and can be updated as the structure changes. This serves the shipyard objectives in Section 1.1 by making dependable verification available earlier and keeping it accessible throughout design and approval.

5.2. Strength-based verification and structural detailing

This paper approaches structural approval primarily through strength and dimensioning. From this perspective, the central question is whether the arrangement, members and connections can withstand the applicable loads and satisfy the relevant structural requirements. The proposed emphasis is on demonstrating that adequacy through a clear connection between the structural definition, requirements and verification evidence.

Requirements for opening clearances, structural continuity, transitions and other aspects of good shipbuilding practice remain part of that assessment where they affect structural performance. They are commonly communicated through detailed drawings. Their verification can also be organised around explicit requirements applied to identifiable structural members and connections, using the dimensions, relationships and properties needed for each check.

The proposed submission would therefore combine a model of the principal structure and its response with detail requirements and the evidence supporting their fulfilment. Depending on the requirement, this evidence may consist of a strength calculation, a geometric or relational check, or confirmation that a standard detail applies. The necessary parameters must be available, but every occurrence need not be represented as complete production geometry.

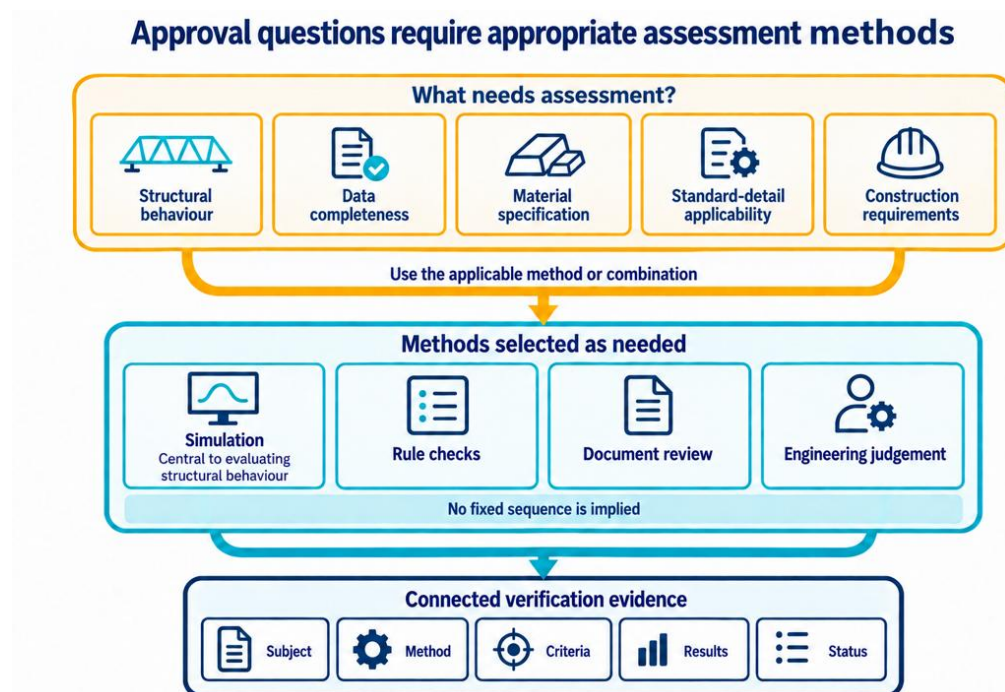


Fig.14: Assessment methods connected to requirements and verification evidence

The deck framing case illustrates one part of this approach. Global member responses are combined with section properties and opening dimensions to obtain the quantities needed for the selected assessment. The relevant detail information enters the calculation even though the opening is not explicitly represented in the global FE mesh.

Making detail requirements and their supporting data explicit also creates a basis for further automation. Algorithms could check defined geometric and structural conditions, while AI agents

could help identify applicable checks, retrieve inputs and coordinate verification. Each outcome would remain traceable to the requirement, design revision, method and evidence used. Developing and validating these capabilities is a future direction beyond the present case.

5.3. Further evaluation

The deck framing case provides a technical example of the proposed calculation approach. We assess its adequacy for the responses and load case presented; it has not undergone an independent approval review. Its use as an alternative to the detailed assessment procedure specified by LR would require agreement on the scope and technical justification.

The next step is to evaluate the connected submission with designers, shipyards and classification societies using the framework in Section 4.2. The key question is whether providing member-specific results and calculation traces across the assessed structure gives the reviewer a more complete basis for approval with less reporting and clarification effort.

Such an evaluation should examine verification coverage, the ability to follow selected calculations, and the handling of design changes. It should also measure the work required to prepare, review and maintain the evidence. We invite industry partners to undertake this evaluation together, building on the connected workflows and calculation example presented here.

6. Conclusions

Simulation driven approval connects structural information, applicable requirements and calculations into a traceable basis for acceptance. It makes explicit what has been assessed, under which conditions and with what result. The central proposition is that these connections can improve the completeness of approval evidence and make it easier to review.

The proposed approach makes results and calculation traces available as structured data for every member within the assessment scope. Reviewers can examine selected calculations in depth while retaining visibility of verification coverage across the structure. This offers a practical route to fewer clarification cycles, less report preparation and more effective review. The same evidence supports dimensioning during design and identifies the checks requiring reassessment when the structure changes.

Approval can therefore be built around the structural information and evidence needed to demonstrate compliance, without requiring every production detail to be geometrically represented. The earlier studies and the present calculation case provide a technical foundation for this approach. Evaluation with shipyards and classification societies is the next step towards demonstrating its benefits in approval practice.

The opportunity is to make the basis for acceptance available throughout design and approval, with each result linked to the relevant structure and its supporting calculations available for review.

References

BUREAU VERITAS (2026), *NR467 Rules for the Classification of Steel Ships*, July 2026, Part A, Chapter 1, Section 1; Part A, Chapter 2, Section 1; Part B, Chapter 1, Section 4.

DNV (2021), *DNV-CG-0127 Finite element analysis*, August 2021, Section 1, 2 and 4.

DNV (2024), *DNV-CG-0550 Maritime services – terms and systematics*, December 2024, Section 5, documentation types H050, H052, H060, H061, H062, H070 and H080.

LLOYD'S REGISTER (2019), *SDA for Primary Structure of Passenger Ships*, August 2019, Chapter

5, Sections 1–4

LLOYD’S REGISTER (2021), *Primary Structure of Naval Ships under Category NS2*, June 2021, Chapter 2, 3.1.10; Chapter 3, Sections 3, 4 and 5

LLOYD’S REGISTER (2026), *LR-RU-001 Rules and Regulations for the Classification of Ships*, July 2026, Classification explanatory note; Part 3, Chapter 1, Sections 2 and 5

OKUMOTO, Y.; TAKEDA, Y.; MANO, M.; OKADA, T. (2009), *Design of Ship Hull Structures — A Practical Guide for Engineers*, Ch.4, Hull Structure Design System, Springer

PEIPPO, J.; PUURULA, J.; PUDD, M.; PONKRATOV, D. (2025), *Solution addressing industry concerns of model-based approval*, COMPIT Conf., Pontignano, pp.245-254

PUURULA, J.; PUDD, M.; GUSANI, S. (2024), *Model-Based FEA Reporting for Streamlining Approval Workflows*, ICCAS Conf., Amsterdam, pp.285–300

From Undisturbed Wind to Onboard Flow: Estimating WAPS Potential through CFD of Ship-Induced Flow Fields

Heikki Hansen, DNV, Hamburg/Germany, heikki.hansen@dnv.com
Nimra Khan, DNV, Hamburg/Germany, nimra.khan@dnv.com
Ole Hympendahl, DNV, Hamburg/Germany, ole.hympendahl@dnv.com
Karsten Hochkirch, DNV, Hamburg/Germany, karsten.hochkirch@dnv.com

Abstract

This paper presents a computationally efficient CFD-based methodology for assessing wind-assisted propulsion system (WAPS) installation locations by quantifying ship-induced modifications to local airflow without explicitly modelling WAPS devices. Steady-state RANS simulations of a simplified Capesize bulk carrier are performed at ballast and scantling draft for apparent wind angles from 0–105°. Flow conditions are evaluated using height-averaged horizontal dynamic-pressure ratios and apparent-wind-angle deviations. Results demonstrate significant spatial variations in wind energy availability and flow direction. Forward of midships, horizontal dynamic pressure increases by 5–15% and apparent wind angle by 2–5°, highlighting the importance of ship-induced effects when screening WAPS locations.

1. Introduction

Wind-assisted propulsion systems (WAPS) are increasingly considered as a means of reducing ship fuel consumption and greenhouse-gas emissions, *Traut et al. (2014)*. Their performance, however, depends strongly on the local wind conditions experienced at the installation position. The hull, deck structures and superstructure can significantly modify both the magnitude and direction of the undisturbed apparent wind through blockage, acceleration, separation and wake effects, *Viola et al. (2017)*.

A critical gap in current performance assessments lies in capturing realistic aerodynamic interactions, both between multiple WAPS installations and between WAPS units and the vessel's hull and superstructure. This paper focuses on the latter, investigating the fundamental impact of ship-induced flow disturbances on the available wind energy across potential WAPS installation locations.

These effects can be investigated using computational fluid dynamics (CFD) or wind tunnel tests, *Gouriou et al. (2026)*. Explicitly modelling WAPS devices, however, considerably increases the number of required simulations, particularly during early-stage design. Different device geometries, operating conditions, installation positions and interactions between multiple units may need to be considered. Utilizing high-fidelity numerical methods to model the interaction between hull, superstructure and WAPS devices still require significant simulation effort. The application of less computationally expensive surrogate models remains a challenge and the reliability of the results varies, *Malmek et al. (2026)*, *Vuillermoz et al. (2026)*.

This study therefore evaluates the ship-induced flow field independently of a specific WAPS technology. CFD simulations of a simplified Capesize bulk carrier without WAPS are performed at two drafts for a range of apparent wind angles. The resulting flow is sampled at potential WAPS installation locations and compared with the corresponding undisturbed inflow. The analysis considers both the available horizontal dynamic pressure and the change in local apparent wind angle. The objective is to provide a computationally efficient method for assessing potential WAPS locations during early-stage design, before a specific device configuration is selected.

2. CFD setup

During early-stage investigation of potential WAPS devices which may be installed on a vessel, conducting CFD analyses of the flow around ship and WAPS devices is often prohibitive due to the

high computational cost. Such simulations would need to consider sheeting angles or rotor speed, possibly for several devices, leading to many variants which need to be investigated.

Therefore, only the flow around the vessel is simulated in this paper; WAPS devices are not included in the CFD geometry. The resulting flow fields can be evaluated for different potential WAPS locations without requiring additional CFD simulations for individual device configurations.

The aerodynamic flow around the ship is simulated using a steady-state RANS-solver. Turbulence is modelled employing the $k-\omega$ -SST model, wall functions are applied. Hex-dominant meshes with prism layers are used, consisting of approximately 30 million cells.

For the flow velocity, an atmospheric boundary layer (ABL) model is applied to represent the height-dependent approaching wind. The wind direction is a constant apparent wind angle, wind shear is not considered, neither due to atmospheric wind shearing nor due to the speed of the vessel.

An atmospheric boundary-layer (ABL) inlet profile is prescribed to represent the height-dependent approaching wind. Sixteen apparent wind angles between 0° – 105° are simulated: $0^\circ, 5^\circ, 10^\circ, 15^\circ, 20^\circ, 25^\circ, 30^\circ, 35^\circ, 40^\circ, 45^\circ, 50^\circ, 55^\circ, 60^\circ, 75^\circ, 90^\circ, 105^\circ$.

3. Analysis methodology

The aim of the analysis is to quantify how the inflow to a potential WAPS installation is modified by the presence of the vessel compared with the corresponding undisturbed inflow. Two quantities are considered: the horizontal dynamic pressure, representing the locally available aerodynamic flow energy, and the local apparent wind angle, representing the change in inflow direction experienced by a potential WAPS device.

The dynamic pressure q is equivalent to the kinetic energy per unit volume of a flow.

$$q = \frac{1}{2}\rho U^2$$

Since a WAPS primarily utilizes the horizontal components of the wind, the horizontal dynamic pressure is defined as

$$q_{hor} = \frac{1}{2}\rho(U_x^2 + U_y^2),$$

U_x and U_y) are the longitudinal and transverse velocity components, respectively. ρ is the air density. To quantify the influence of the vessel independently of the atmospheric boundary-layer profile, the local horizontal dynamic pressure is normalized by the corresponding value in the undisturbed reference flow at the same height to give the ratio:

$$R_q(z) = \frac{q_{hor}(z)}{q_{hor,ref}(z)}.$$

A value of $R_q=1$ therefore represents the undisturbed inflow condition. Values greater than unity indicate an increase in horizontal dynamic pressure relative to the undisturbed flow, while values below unity indicate a reduction caused by the ship-induced flow field.

In addition to the available dynamic pressure, the change in local flow direction is investigated. The local apparent wind angle is determined from the horizontal velocity components and compared with the undisturbed apparent wind angle AWA_0 . The resulting apparent-wind-angle deviation is defined as

$$\Delta AWA(z) = AWA_{local}(z) - AWA_0.$$

Thus, $\Delta AWA=0^\circ$ indicates that the local flow direction is identical to the undisturbed inflow direction, while positive or negative values indicate a ship-induced change in the local apparent wind angle.

The flow field is evaluated at 18 longitudinal positions distributed along the vessel. At each longitudinal station, three transverse locations are considered: one at the vessel centreline and one towards either side of the deck at ± 0.425 of the breadth. The two off-centre locations are referred to relative to the apparent wind direction as the windward and leeward positions. This results in a total of 54 vertical sampling locations for each simulated apparent wind angle.

Fig.1 shows the investigated sampling arrangement¹. The longitudinal stations cover the cargo-hold region of the vessel, allowing the formation of the ship-induced flow field along the deck to be assessed. At each station, the three vertical sampling lines represent potential WAPS locations across the vessel breadth. Each line covers a representative WAPS height of 35 m, extending from 5 m to 40 m above the local deck level.

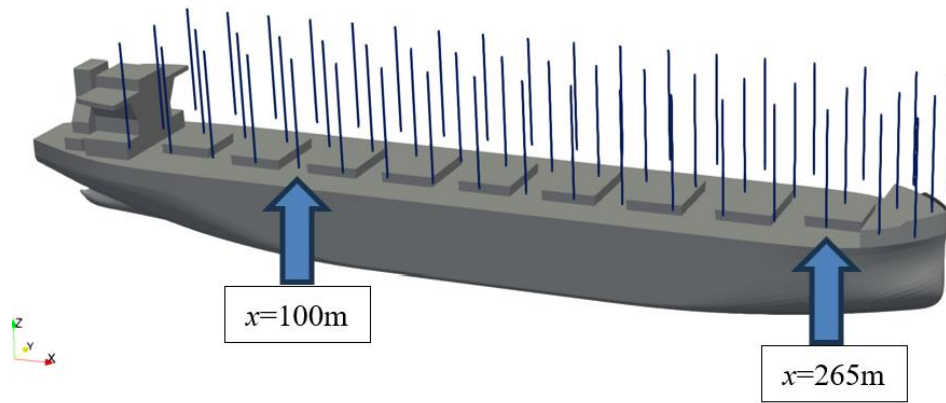


Fig.1: Investigated WAPS sampling locations above the vessel

At each sampling location, the flow quantities are extracted over the representative WAPS height at a uniform vertical spacing of 0.5 m, resulting in 71 sampling points along each vertical line. The selected vertical range represents a generic WAPS installation and is not associated with a particular WAPS technology.

To obtain representative values over the complete WAPS height, the vertical profiles are numerically integrated using Simpson's rule and divided by the investigated height. The WAPS-height-averaged horizontal dynamic-pressure ratio is therefore defined as

$$\overline{R_q} = \frac{1}{H} \int_{z_1}^{z_2} R_q(z) dz,$$

while the WAPS-height-averaged apparent-wind-angle deviation is

$$\overline{\Delta AWA} = \frac{1}{H} \int_{z_1}^{z_2} \Delta AWA(z) dz,$$

where $z_1=5$ m and $z_2=40$ m above the deck, giving $H=35$ m. The integrations are evaluated numerically using the composite Simpson rule with a uniform spacing of $\Delta z=0.5$ m.

The resulting height-averaged quantities allow the ship-induced flow conditions to be compared systematically across longitudinal and transverse positions, apparent wind angles and loading conditions. Both measures are considered together, since an increase in horizontal dynamic pressure does not necessarily represent a favourable condition for WAPS operation if it is accompanied by a significant change in the local apparent wind direction.

4. Results and discussions

Fig.2 gives an impression of the flow around the ship at an apparent wind angle of 45° . The flow is complex with significant separation zones, especially in the forward part of the cargo hold.

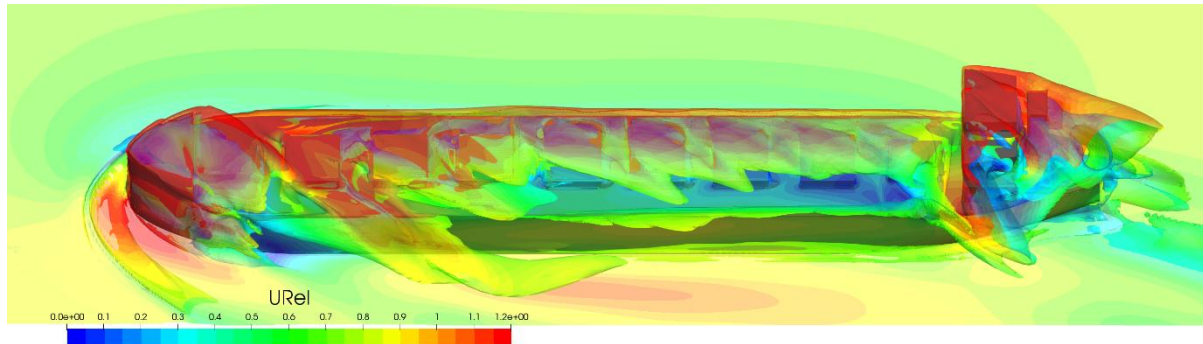


Fig.2: Vorticity iso-surface coloured by flow velocity, ballast draft, $AWA=45^\circ$

4.1. Horizontal dynamic pressure

The influence of the vessel on the horizontal flow is first examined through the horizontal dynamic-pressure ratio R_q . Fig.3 shows the flow velocities relative to the reference velocity at $x=265$ m and $AWA=45^\circ$ for ballast condition and Fig.4 shows the corresponding vertical profile on the centreline. The disturbed local and the undisturbed reference horizontal dynamic pressure are plotted on the left of Fig.4 and the resulting horizontal dynamic-pressure ratio R_q on the right. The ship-induced disturbance is strongest close to the deck, while the ratio approaches the undisturbed condition with increasing height. Within the representative WAPS region, values above and below unity indicate local acceleration and reduction of the horizontal flow relative to the undisturbed ABL.

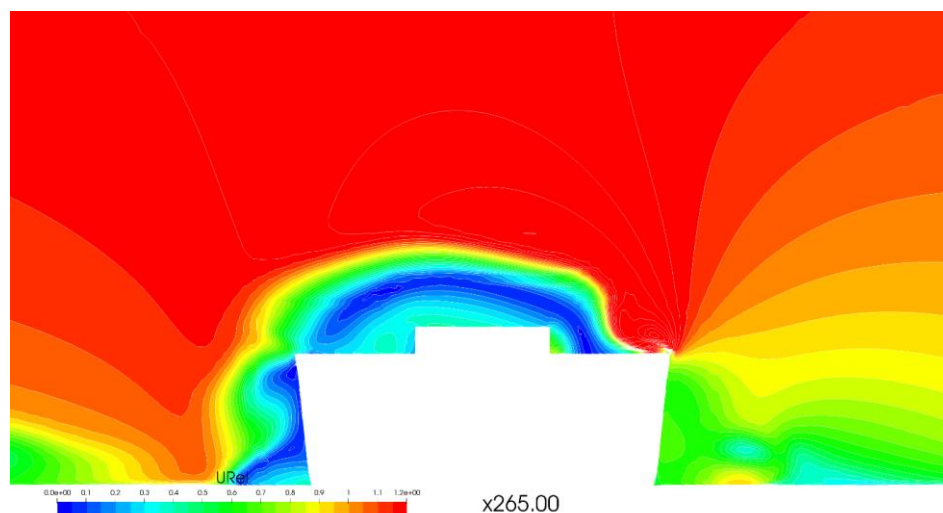


Fig.3: Flow velocities relative to reference velocity at $x=265$ m, ballast, $AWA=45^\circ$

The vertical distribution changes considerably with apparent wind angle and position on the vessel. Figure 5 therefore compares (R_q) over the investigated height for all 16 apparent wind angles at two representative longitudinal positions, $x=100$ m and $x=265$ m. The leeward, centreline and windward locations are shown separately for both loading conditions.

The profiles demonstrate that the ship-induced disturbance depends strongly on height, apparent wind angle and transverse position. The largest deviations from the undisturbed condition generally occur close to the deck, where the flow is directly affected by the hull and deck geometry. With increasing height, the profiles tend towards the undisturbed reference condition. Differences between the two

longitudinal positions further demonstrate that the flow experienced by a potential WAPS cannot be represented by a single correction applied over the complete vessel.

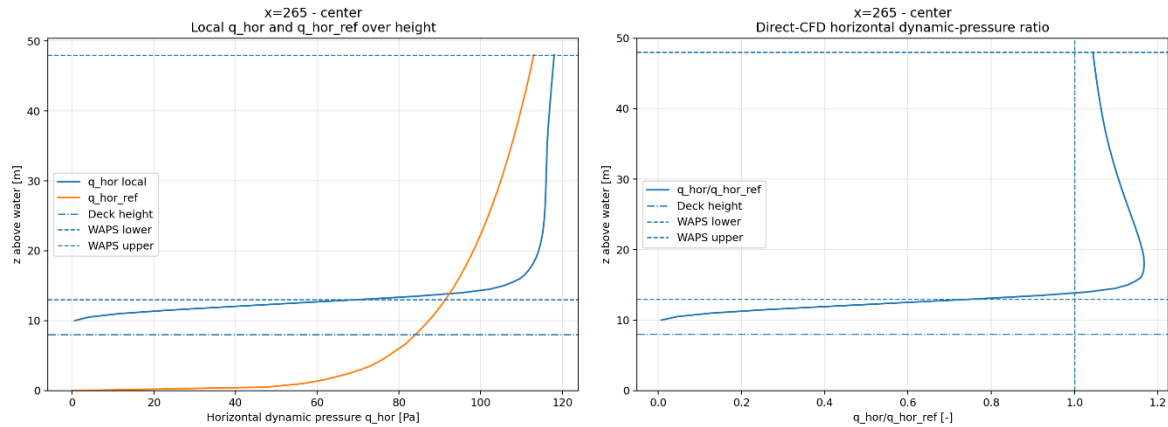


Fig.4: Vertical distribution of local and reference horizontal dynamic pressure (left) and resulting dynamic-pressure ratio (right) at $x=265$ m, centreline, ballast, $AWA=45^\circ$

This example demonstrates that the ship-induced disturbance is strongly dependent on height. Changes in flow magnitude occur simultaneously with changes in flow direction. To assess whether these effects are relevant over the complete WAPS operating region, the vertical distributions are subsequently averaged over the 5–40 m WAPS height using the procedure described in section 3. Fig.6 presents the resulting WAPS-height-averaged horizontal dynamic-pressure ratio for all investigated apparent wind angles as a function of longitudinal position.

The longitudinal distributions show that the available horizontal dynamic pressure is not uniform along the vessel. Regions with $R_q > 1$ indicate an increase relative to the undisturbed flow, while $R_q < 1$ represents a reduction. The magnitude of these effects depends on apparent wind angle and differs between the leeward, centreline and windward locations. Differences are particularly evident towards the ends of the investigated region close to the bow and the deckhouse, while the flow over parts of the cargo-hold region shows a more systematic longitudinal development.

The comparison between ballast and scantling drafts further indicates that the loading condition influences the resulting flow field. Nevertheless, the overall results from both drafts demonstrate the same important characteristic: the local horizontal dynamic pressure available to a potential WAPS can differ appreciably from that predicted using the undisturbed apparent wind alone.

To account for the occurrence of different wind conditions, the WAPS-height-averaged horizontal dynamic-pressure ratios are weighted according to the probability of the investigated apparent wind angles. The weights were taken from the apparent wind speed distribution from a systematic study of optimized routes for a bulk carrier with rotor sails, as shown in Fig.7, *Hansen and Hympendahl (2026)*. Fig.8 compares the resulting probability-weighted ratio for ballast and scantling draft.

The results show that the available horizontal dynamic pressure generally increases along the cargo-hold region towards the bow, with differences between the two drafts becoming more pronounced for the leeward and centreline positions. The windward location shows comparatively similar behaviour for both loading conditions.

The simulations for this vessel indicate that the WAPS-height-averaged horizontal dynamic pressure increases between 5–15% compared to the undisturbed flow forward of midships. The largest increase can be observed just aft of the forecastle above the forwardmost hatch. There is no significant difference in horizontal dynamic pressure on the centreline compared to placing the WAPS on one side considering that the vessel sails on both tacks.

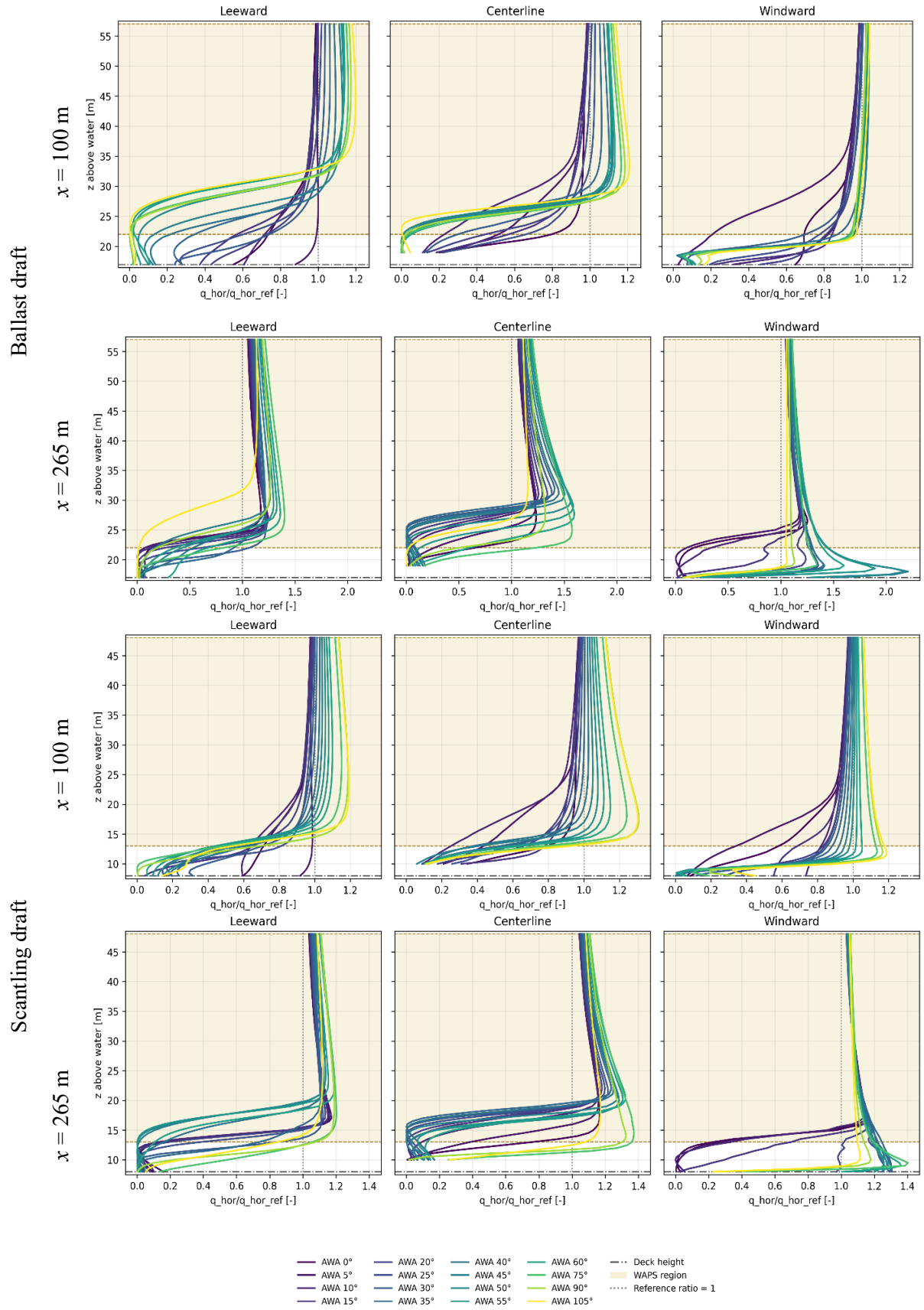


Fig.5: Vertical distribution of the horizontal dynamic-pressure ratio for all investigated apparent wind angles at $x=100$ m and $x=265$ m

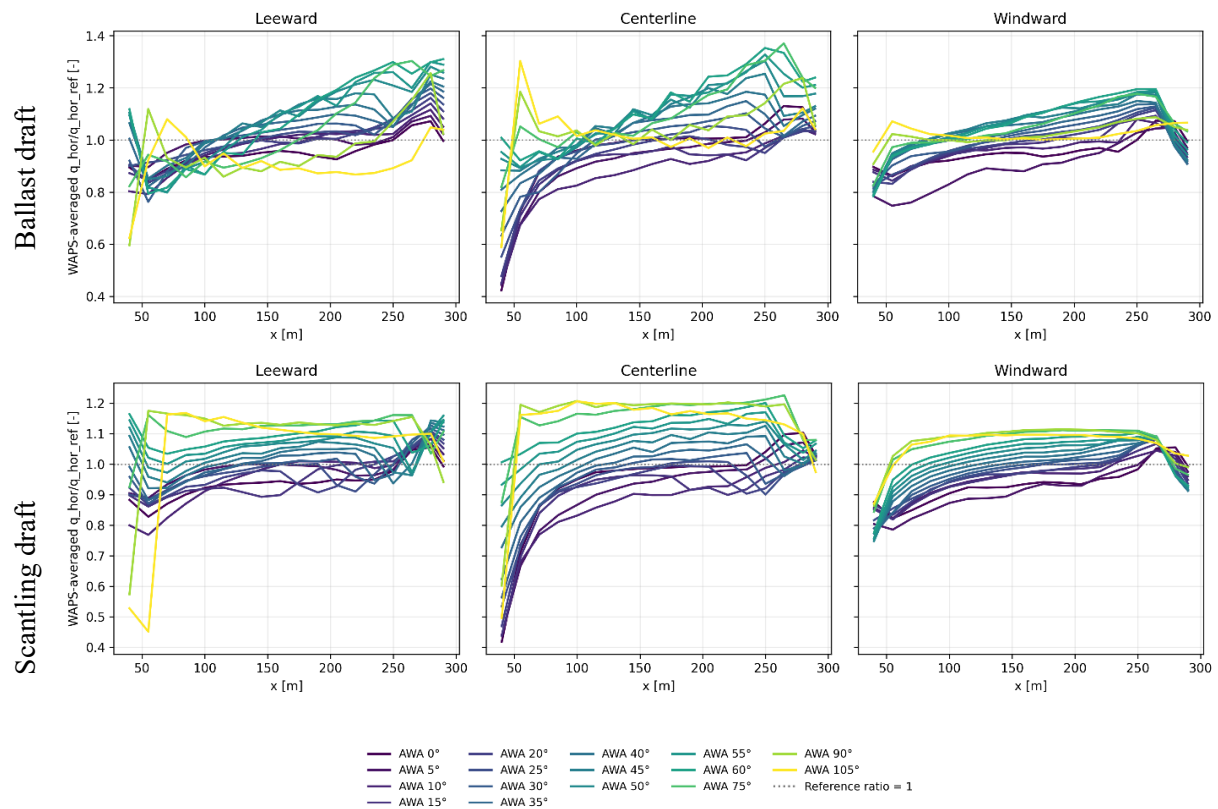


Fig.6: WAPS-height-averaged horizontal dynamic-pressure ratio along the vessel for all investigated apparent wind angles

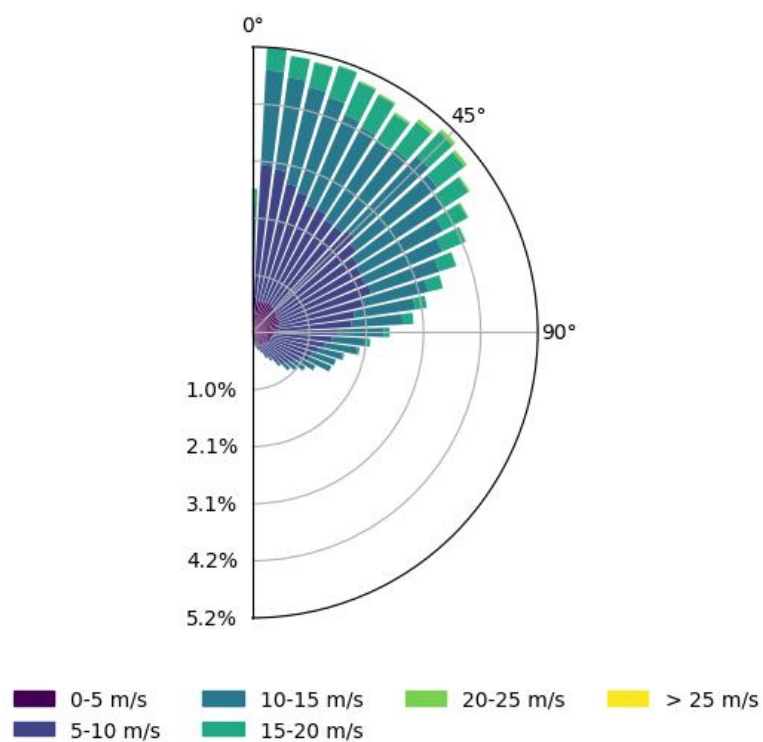


Fig.7: AWA distribution used for weighting

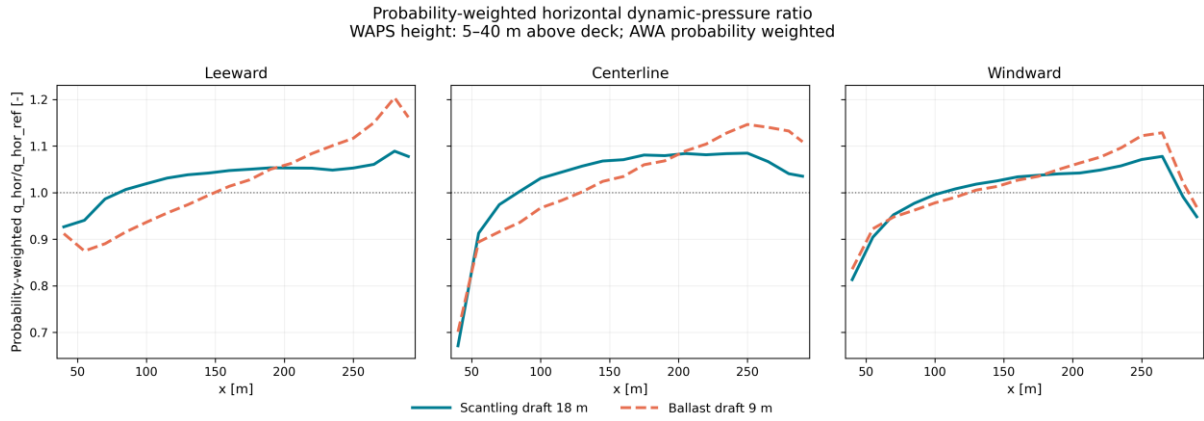


Fig.8: Probability-weighted WAPS-height-averaged horizontal dynamic-pressure ratio along the vessel for scantling and ballast draft

4.2. Local apparent wind angle

In addition to the available horizontal dynamic pressure, the ship-induced flow also modifies the local flow direction experienced by a potential WAPS. This is quantified by the apparent-wind-angle deviation, $\Delta AWA = AWA_{local} - AWA_0$. Fig.9 shows an exemplary vertical distribution at $x=265$ m on the centreline for the ballast condition at $AWA=45^\circ$. The largest deviations occur close to the deck, while the local apparent wind angle approaches the undisturbed value with increasing height.

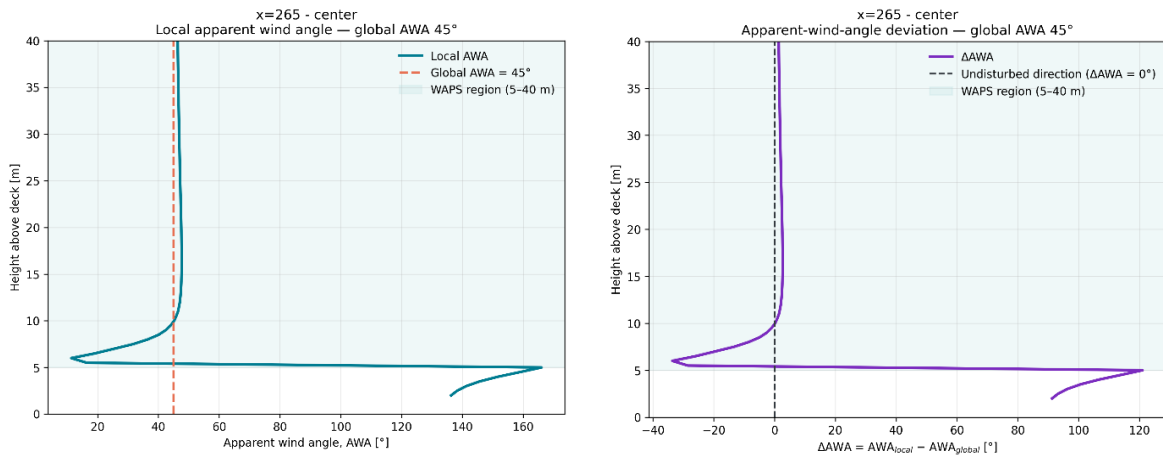


Fig.9: Vertical distribution of local apparent wind angle (left) and as difference to the undisturbed flow (right) at $x=265$ m, centreline, ballast, $AWA=45^\circ$

The dependence on apparent wind angle and position is illustrated in Fig.10, where the vertical distributions of ΔAWA are compared for all investigated apparent wind angles at $x=100$ m and $x=265$ m. The strongest directional disturbances occur close to the deck, whereas the deviations decrease rapidly with height. Differences are also visible between the leeward, centreline and windward locations and between the two longitudinal positions.

As for the horizontal dynamic pressure, the apparent wind angle deviation is averaged over the representative WAPS height of 5–40 m. Fig.11 shows the resulting height-averaged ΔAWA along the vessel for all investigated apparent wind angles.

The longitudinal distributions show that the local flow direction can differ by several degrees from the undisturbed apparent wind over substantial parts of the cargo-hold region. The effect depends on apparent wind angle and transverse position, with generally larger variations towards the ends of the investigated region close to the bow and the deckhouse. Differences between the two loading conditions are also visible for all three transverse positions.

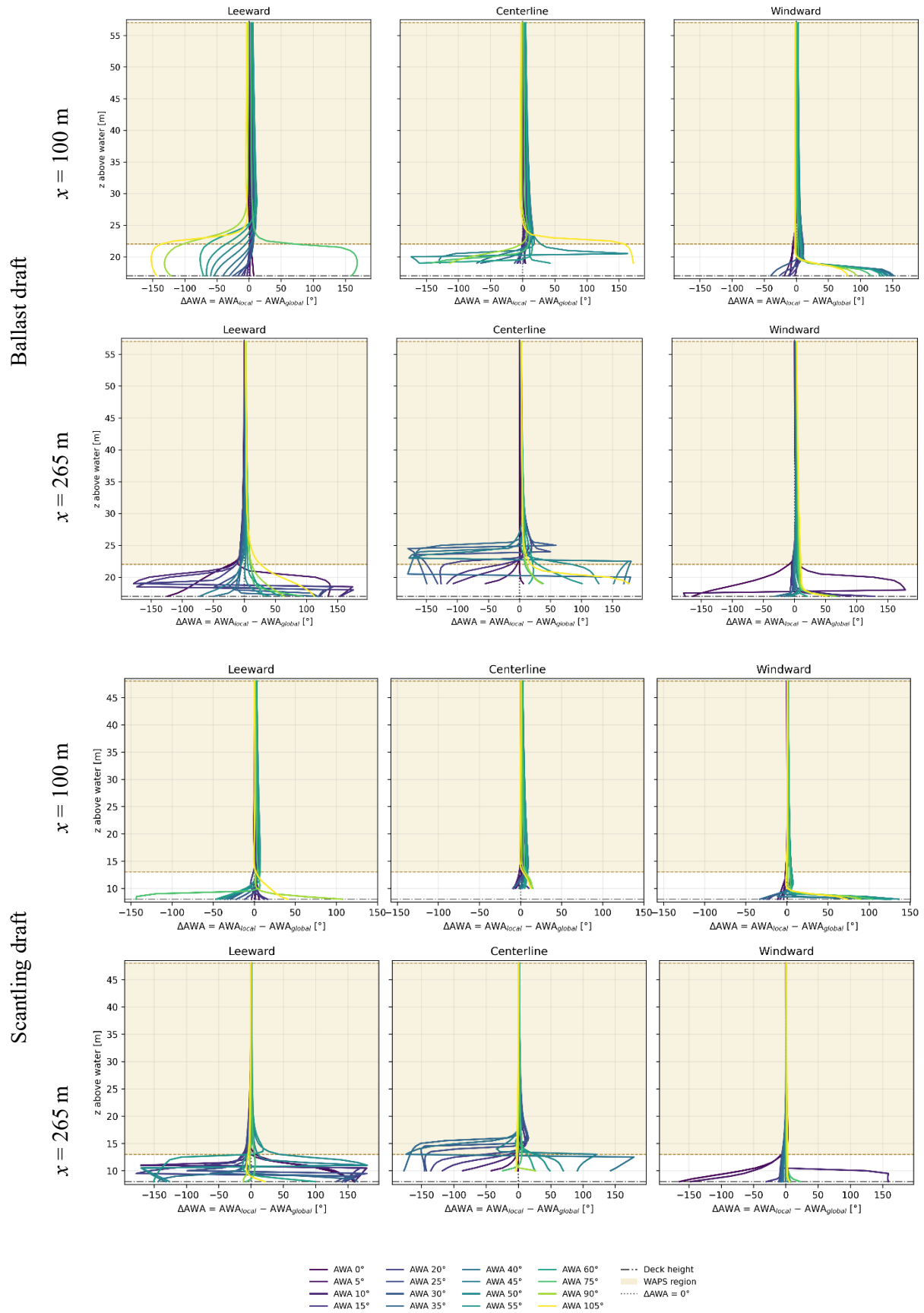


Fig.10: Vertical distribution of apparent wind angle deviation for all investigated apparent wind angles at $x=100$ m and $x=265$ m

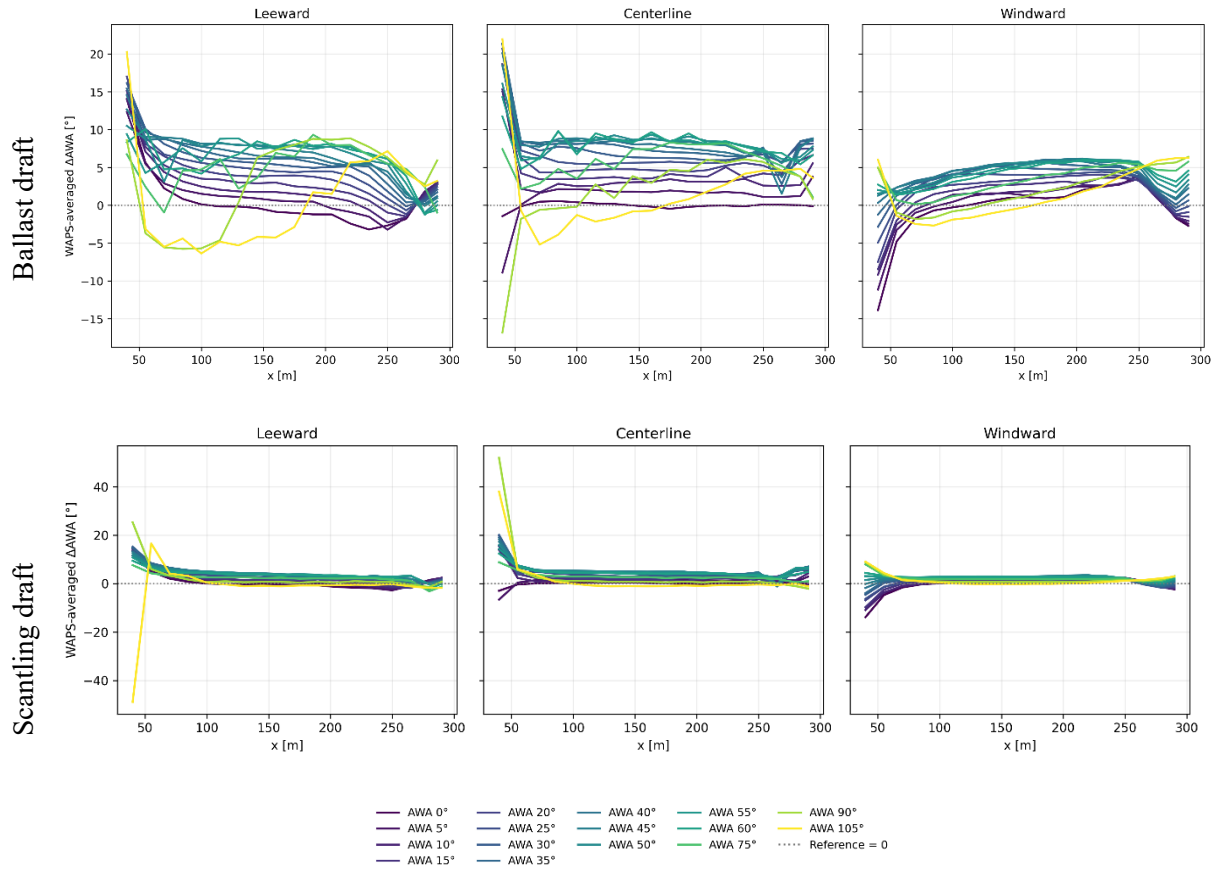


Fig.11: WAPS-height-averaged apparent wind angle deviation along the vessel for all investigated apparent wind angles

Finally, the height-averaged deviations are weighted according to the probability of occurrence of the investigated apparent wind angles. Fig.12 compares the resulting probability-weighted ΔAWA for the two loading conditions.

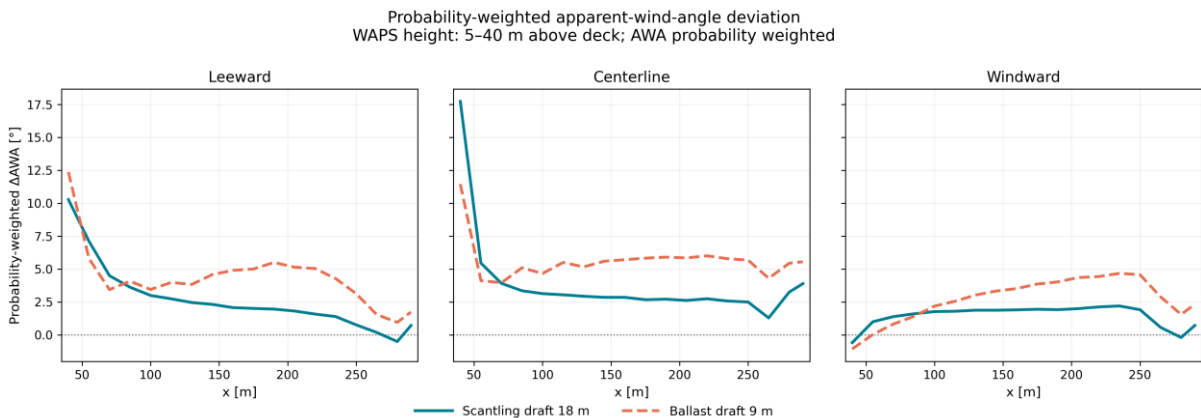


Fig.12: Probability-weighted WAPS-height-averaged apparent-wind-angle deviation along the vessel for scantling and ballast draft

The probability-weighted results indicate a predominantly positive deviation over much of the investigated region. The largest deviations occur towards the aft positions, while the influence generally decreases further along the vessel towards the bow. Differences between the drafts remain visible, particularly at the centreline and leeward locations.

5. Implications for WAPS placement

The results demonstrate that the aerodynamic flow environment available to a WAPS varies considerably over the vessel and cannot be fully characterized using only the undisturbed apparent wind condition. Both the magnitude and direction of the local flow are affected by the interaction between the atmospheric inflow and the vessel geometry.

The proposed approach can be employed as a pre-screening method during early-stage WAPS integration. Since the WAPS itself is not included in the CFD simulations, the calculated ship flow field can be interrogated at multiple candidate locations and over different representative device heights without repeating the CFD calculation for each individual WAPS configuration.

The method is not intended to replace detailed simulations of the final WAPS arrangement. Once candidate locations and a specific WAPS technology have been selected, device geometry, operating conditions and interactions between multiple units should be considered explicitly. Instead, the present approach provides a computationally efficient means of identifying and comparing the flow conditions available at potential installation locations before undertaking such detailed analyses.

6. Conclusions

A CFD-based methodology has been presented for evaluating the ship-induced flow conditions relevant to WAPS installation without explicitly modelling a WAPS device. The approach was demonstrated for a simplified Capesize bulk carrier at scantling and ballast draft for 16 apparent wind angles.

The flow was evaluated at 54 potential WAPS locations using the WAPS-height-averaged horizontal dynamic-pressure ratio and apparent-wind-angle deviation. The analysis shows that the vessel modifies both the magnitude and direction of the approaching flow and that these effects vary with longitudinal position, transverse position, apparent wind angle and loading condition.

The results highlight the importance of considering the local ship-induced flow when assessing potential WAPS locations. In particular, locations associated with increased horizontal dynamic pressure may simultaneously experience changes in apparent wind direction, demonstrating the need to consider both quantities together. For the investigated vessel an increase in horizontal dynamic pressure of 5–15% and an increase in apparent wind angle of 2–5° is observed for WAPS locations forward of midships.

The proposed ship-only CFD approach allows computationally efficient screening of potential WAPS locations during early-stage design. Detailed modelling of the selected WAPS geometry and its operating conditions can subsequently be performed for the most relevant candidate configurations.

References

- GOURIOU, C.; TARDIF, T.; STELLA, F.; BARRET, A.; FORTIN, P.-A. (2026), *An Experimental Approach to WAPS Performance Assessment for RoRo Vessel*, 7th Int. Conf. Innovation in High Performance Sailing Yachts and Wind Assisted Ships (InnovSail), Gothenburg, pp.217-235
- HANSEN, H.; HYMPENDAHL, O. (2026), *Route Optimization Effects on Wind-Assisted Bulk Carrier and General Cargo Ship*, 7th Int. Conf. Innovation in High Performance Sailing Yachts and Wind Assisted Ships (InnovSail), Gothenburg, pp.605-632
- MALMEK, K.; LARSSON, L.; LI, D.-Q.; BENSOW, R.; FINNSGÅRD, C. (2026), *Aerodynamic Hull-Sail Interaction for a Six-Wingsail PCTC: A Comparison of Wind-Tunnel Experiments, URANS Simulations and a Nonlinear Lifting-Line Method*, 7th Int. Conf. Innovation in High Performance Sailing Yachts and Wind Assisted Ships (InnovSail), Gothenburg, pp.71-84

TRAUT, M.; GILBERT, P.; WALSH, C.; BOWS, A.; FILIPPONE, A.; STANSBY, P.; WOOD, R. (2014), *Propulsive power contribution of a kite and a Flettner rotor on selected shipping routes*, Applied Energy, 113, pp.362-372

VIOLA, I.M.; SALVATORE, F.; RICCIARDI, A.; FLAY, R.G.J. (2017), *Ship aerodynamic analyses and optimisation: State of the art and future challenges*, Ocean Engineering, 130, pp.122-134

VUILLERMOZ, L.; BLANCHIER, L.; SAETTONE, S.; DUPUY, M. (2026), *Impact of Aerodynamic Sail Interactions on Weather Routing Performance for a Wind-Assisted Ship*, 7th Int. Conf. Innovation in High Performance Sailing Yachts and Wind Assisted Ships (InnovSail), Gothenburg, pp.633-652

A Single Source of Truth Needs Shared Meaning: Ship Data Ontology in the Age of AI

Henrique M. Gaspar, Norwegian University of Science and Technology, Ålesund/Norway,
henrique.gaspar@ntnu.no

Abstract

A 2025 paper by the author argued for aggregating six data domains of the ship life-cycle into a single source of truth. This paper argues that such aggregation is only as coherent as the shared meaning underneath it. Following one water ballast tank through data structures, life-cycle phases, standards, rules and disciplines, it distinguishes descriptive from prescriptive knowledge, and shows why each difference requires a dictionary. Ongoing efforts in SEUS, INTELLIA, MAI and NMRI illustrate the problem. The paper closes with a speculation: as large language models become translators between schemes, the meaning of ship data must travel with it, as metadata everywhere.

1. Single Source of Truth (SSoT) and Inference

I want to extend here the idea discussed last year, *Gaspar (2025)*, about a single source of truth (SSoT) for the ship life-cycle. In practice, an SSoT is seen as a digital system: a database, or a federation of databases, in which the ship's data are stored and from which they are retrieved. *Gaspar (2018)* presented such data through the data, information, knowledge and wisdom (DIKW) hierarchy, following *Anderson (2015)*, here exemplified: the hours spent on a design project are data, their comparison with similar projects is information, understanding the consequences is knowledge, and using that knowledge in the next design is wisdom. Each step up the hierarchy is an act of inference: finding the right piece of data, knowing what it means, relating it to others, and deciding what follows. Whether done by an experienced engineer or by a computer, that inference depends on the meaning the data carry.

Each of these steps depends on how the data are organised: to be found, a piece of data must be located; to be understood, its definition must be known; to be related, its relations must be recorded. An experienced engineer can compensate for poor organisation, because the missing structure lives in their head. A computer cannot. This becomes more serious now that AI is increasingly used not only to query ship data, but to organise it: to classify, link and convert it between schemes. An AI will always return an answer, fluent and confident, whether or not the organisation beneath it supports one. The organisation of data is therefore not a preliminary to inference, but its precondition.

Organisation, however, is not only a matter of structure. Aggregating data from multiple domains, and inferring from them, also assumes that the data mean the same thing to everyone involved. In practice they rarely do. The thesis of this paper is that a single source of truth is only as coherent as the shared meaning underneath it. Without that shared meaning, aggregation produces one place where many incompatible descriptions of the same ship sit side by side.

2. Basis for inference: how we organise data, and where meaning gets lost

The lack of shared meaning in maritime data has many causes. A central one is that, unlike the car or aircraft industries, maritime never settled on a dominant unified software or data standard, largely because of the one-of-a-kind nature of ships. There are (and have been) many attempts - SFI, the ship-specific parts of ISO 10303/STEP, DNV VIS, OCX - but none has yet become *the* standard. I reiterate the conclusion from *Gaspar (2018)*, that finding a common standard may be practically unfeasible, and that the incompatibility of a single standard must be included in the process.

Tracing the history of computational tools in maritime, we find each of the classic ways humans store and structure data in computers in use somewhere along the life-cycle. Here I briefly revisit them,

simplifying as I do when teaching it, from the simplest to the richest, using one running example: Project Ship A, from specification through design, construction and operation. The same data is used in every case:

- Project Ship A: length 100 m, beam 20 m, draught 6 m, speed 14 kn
- Water ballast tank WBT-3P: volume 450 m³, part of the ballast system, in Block 12, built from steel plates
- Ballast pump BP-01: capacity 300 m³/h, fills WBT-3P, supplied by X

We have thus a Project, then two basic items that differ in origin: the tank is built, steel that becomes a tank in the yard, while the pump is bought, a finished product ordered from a supplier.

a) Variables. Direct storage in memory.

```
L = 100; B = 20; T = 6; V = 14
WBT3P_volume = 450; WBT3P_block = 12
BP01_capacity = 300; BP01_supplier = "X"
```

The simplest and most direct form, as in hand calculations and early programs. The meaning, however, lives only in the variable name: nothing but a naming habit ties the WBT3P_ values together, a second ship requires renaming everything (L_A, L_B), and units are not stored anywhere.

b) Lists. Arrays and vectors.

```
ShipA = [100, 20, 6, 14]
WBT3P = ["ballast", 12, 450]
BP01 = ["ballast", 300, "X"]
```

Values are now grouped and easy to compute over, as with vectors or hull offsets. The meaning moves to the position: the reader must know that the third value of ShipA is the draught, and inserting a new field shifts every position after it.

c) Tables. Spreadsheets (Excel) and relational databases (SQL).

```
Ships:      Ship | L   | B   | T   | V
            A   | 100 | 20  | 6   | 14

Tanks:      Tank  | Ship | System | Block | Volume
            WBT-3P | A    | Ballast | 12    | 450

Equipment:  Tag   | Ship | System | Fills  | Capacity | Supplier
            BP-01 | A    | Ballast | WBT-3P | 300      | X
```

Column headers name the meaning. Ships B and C are simply more rows, which can be sorted, filtered and compared, and columns such as Ship and Fills link the tables. Tables are everywhere in practice. Their limitation is that every row shares the same columns, so a tank and a pump need different tables, and each department builds its own tables with its own column names.

d) Hierarchical. Objects (JSON, XML) or folders.

```
ShipA
├─ Main: L 100, B 20, T 6, V 14
├─ Systems
│   └─ Ballast
│       └─ WBT-3P: volume 450
│           └─ BP-01: capacity 300, supplier X
```

or, as folders: ShipA/Systems/Ballast/BP-01/datasheet.pdf

This is how the industry organises ship data: SFI codes, CAD assembly trees, PLM product structures and shared folders are all hierarchies. Each node can carry its own attributes, and the structure is easy to read. But each item has only one parent, and every department, sometimes every person, divides the ship differently: design by system, the yard by block, purchasing by supplier, operation by maintenance plan. The yard therefore builds its own tree, ShipA/Block12/WBT-3P, in which BP-01 does not appear at all. Choosing a tree means choosing a viewpoint, and each new viewpoint means another tree duplicating the first. Bill-of-materials structures in PLM share this limitation, lacking a concept of topological connections, relations, and dependencies between the parts, *Ando and Seppälä (2025)*.

e) Graphs and tags.

```
Tags:   WBT-3P  #shipA #ballast #block12

Graph:  WBT-3P -part_of→      Ballast system -part_of→ Ship A
        WBT-3P -located_in→   Block 12        -part_of→ Ship A
        WBT-3P -bounded_by→   steel plates
        BP-01  -fills→         WBT-3P
        BP-01  -supplied_by→   X
        (volume 450 m³ and capacity 300 m³/h as properties)
```

Graphs are the newcomer. Knowledge graphs and graph databases are being explored in research and pilot projects, including those discussed in Section 6, but are not yet used in everyday ship design and construction. Their promise is that the system view and the block view coexist without duplication: relations are explicit and named, and new ones can be added freely. They are, however, harder for people to read and browse. Tags are flexible but do not say how WBT-3P relates to Block 12. In a graph, anyone can name the relations as they like (*located_in* or *installed_in*?), so the graph holds the meaning only if people agree on those names.

Tags have been proposed before as a way out of fixed hierarchies in ship design. *Gaspar (2018)* called for multi-hierarchical data with plural tags, so that the main machinery could belong to the propulsion system in a functional division and to the hull in a physical one. *Gaspar (2022)* proposed a *#hashtag* catalogue in which each component carries the information needed to place it in any taxonomy, with rules generating each view from the tags: a propeller would sit in the aft body of the physical taxonomy, in the propulsion system of the functional one, at its supplier's factory in a geographical one and in a budget line in an economic one. Tags give views. What they lack is a statement of how the tagged items relate, and according to whom.

What all these structures lack, to different degrees, is metadata: data about data, which is what the data dictionary of *Wolanyk (1979)* was (probably) meant to hold. A variable has none beyond its name, and a list has none at all. A table has headers and data types, a hierarchy has keys, attributes and file properties, and a graph can carry properties on both nodes and relations. In practice, most ship files, whether spreadsheets, drawings or PDFs, carry metadata about the file (author, date, version), but rarely about the meaning of what is inside: the unit, the convention, the scheme it follows, where its definition comes from, and who is responsible for it. Yet this is exactly what makes a value readable outside its original context. $L = 100$ becomes unambiguous the moment it declares that it is the length between perpendiculars of Ship A, in metres, following a given rule. Whatever structure is used, the meaning of the data is safest when it travels with the data.

Table I: Where the meaning lives in each data structure, and what metadata it can carry

Structure	Where the meaning lives	Metadata it can carry
a) Variables	in the name	name only
b) Lists	in the position	none
c) Tables	in the column header	headers, data types
d) Hierarchical	in the path (one parent)	keys, attributes, file properties
e) Graph	in named relations (many parents)	properties on nodes and relations

Loosely read against the DIKW hierarchy, each step up the ladder stores more of the meaning needed to climb from data towards knowledge, while wisdom remains the inference made over them.

3. Life-cycle and data domains

3.1. The six domains

Data domains are logical groups of data referring to interrelated information about a common purpose, object or concept. They are a helpful categorical tool for managing the information of complex systems. *Gaspar (2025)* and *Bronson et al. (2024)* converged on six domains commonly used in the shipping industry:

- **Systems (functional)**: information on what the systems in a ship are intended to do. This is one of the most popular ways to break down a vessel, with SFI and the US SWBS as the most widely used system-oriented breakdowns. It is managed mainly through CAD/CAE tools for performance, weight, stability and structural assessment.
- **Product**: information on the form or physical realisation of the ship parts that perform those functions: the hull and the outfitting with its equipment and furnishings. During construction, these are typically broken down by physical connection into zones and blocks. It is managed through CAD/CAM, asset software with detailed parts lists, and company ship catalogues.
- **Process**: information on the workflows that produce the ship, both administrative (planning and scheduling) and technical (e.g. structural FEA, electrical load analysis). Tools range from in-house Excel solutions to advanced planning software.
- **People**: information on the persons carrying out tasks and the many organisations involved, from designers' security compliance to crew certification. It is managed through human-resource and crew-management tools.
- **Geolocation**: information on the location of ship assets and of material or labour resources, which are highly dispersed in a globally distributed industry. It ranges from offshored yards to AIS tracking in operation.
- **Contextual**: information related to decision-making, giving a high-level understanding of the other domains, from market to vessel behaviour. Fleet dashboards are a common example.

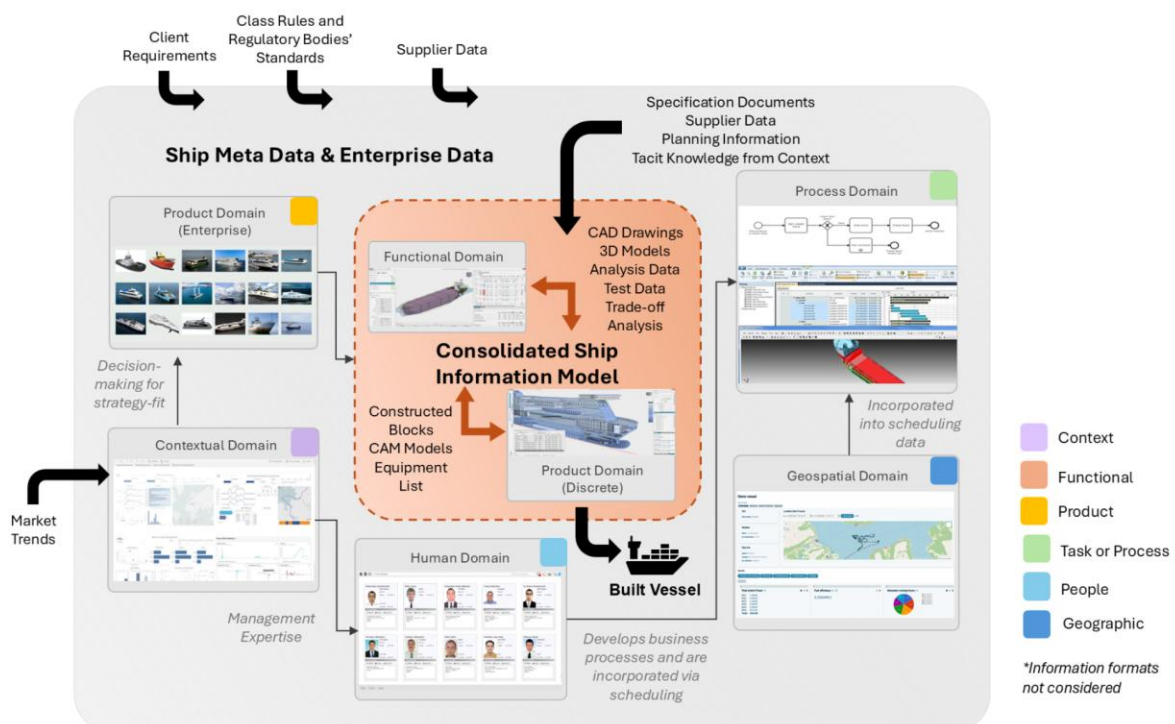


Fig.1: Six data domains around a ship information model, *Bronson et al. (2025)*

The domains are highly linked: information within one domain can be reinterpreted in another, based on intent and function. Many shipbuilding work definitions, such as planning units, already combine Systems with the Product and Process domains. A similar idea exists in ISO 81346, which describes an object through its function, product, location and type aspects.

Fig.1 shows how the domains feed one consolidated ship information model, each with its own data, tools and inputs, from class rules and supplier data to tacit knowledge. The challenge is not the lack of data or tools in each domain, but the lack of shared meaning between them.

3.2. The domains change over the life-cycle

The domains are not fixed: their content changes over the life-cycle, and so does which of them dominates. Design is dominated by a functional breakdown (Systems), *Bronson et al. (2024)*. The view moves to Product in detailed engineering and construction, where Process and Geolocation also dominate in coordinating materials and labour in the yard. The view returns to Systems for the overall evaluation of the ship at delivery. The same ship is therefore understood differently in each phase, and the same part is described differently by each domain.

Even the life-cycle itself is not described the same way by everyone. *Ando and Seppälä (2025)* show that shipowners, naval architects and shipyards each divide it into different phases, presenting a particular challenge for unification. Phases also overlap, and each typically starts with incomplete input from the previous one.

Moreover, for a design company the life-cycle is not a single sequence. A ship is designed once but lived with for decades, and a design company rarely works with just one ship: it works with an idea of a ship, which evolves, is customised, and is built differently each time. At any moment, different instances of that idea exist side by side, some on the drawing board, some under construction, others in operation and sending back sensor data. WBT-3P on Ship A, its counterpart on a sister ship, and a modified version in a new variant are the same tank idea at different phases at once. Series production of one design may continue for nearly a decade, and sometimes for over 20 years, *Ando and Seppälä (2025)*.

3.3. Examples

Table II follows WBT-3P from Section 2 through a very simple life-cycle (design, construction and operation), for the sake of exemplification.

Table II: WBT-3P across domains and life-cycle phases

Domain	Design	Construction	Operation
Product	a volume in the GA, 450 m ³	plates and stiffeners, then a coated and tested tank	a tank that corrodes and gets inspected
Geolocation	a position in the hull	steel mill → yard workshop → Block 12 → ship	onboard; the ship moves
Process	arrangement, tank table	cut, weld, coat (SFI 285), test (SFI 204)	ballast operations, class surveys
Systems	ballast system (SFI 801)	piping connected	in service
People	Designer	welders, painters, class surveyor	crew, inspector
Contextual	class rules, the Ballast Water Management Convention	schedule, steel price	ballast water regulations, survey intervals

The bought pump BP-01 follows a different path. Its Product exists before the project starts, as supplier X's datasheet. Its Geolocation starts at the supplier, then moves to the yard warehouse and finally the

engine room. *Ando and Seppälä (2025)* trace a similar path for a bought power generator: from a line in the specification and the electrical load balance, to a symbol in a single-line diagram, to a 3D component with connections, to a delivery and installation date, and finally to sensor data in operation. The same tank is also described differently by each discipline, as shown in Table III.

Table III: One tank, many descriptions

Discipline	What WBT-3P is
Stability	volume, centre of gravity, free-surface moment, sounding table
Structure	plates and stiffeners, scantlings, design pressure head
Shipyard	parts of Block 12, weld lengths, coating area, test procedure
Systems	a node in the ballast piping (SFI 801)

Koelman (2023b) makes the same observation for the pipes connected to it: in a stability tool such as PIAS, a pipe is modelled for damage stability, as shape and connectivity between tanks, while in a production CAD tool such as CADMATIC it is modelled for installation, with brackets and gaskets. Both systems require different levels of detail, and other systems will have yet other requirements. The same holds at a larger scale: the hull is a set of volumes for stability, a smooth surface for hydrodynamics, plates and frames for structure, and steel blocks for the shipyard. It's the same word for very different objects.

4. Descriptive knowledge: Ontology for humans and for computers

A tank that is a volume to a stability engineer, a set of plates to the structural engineer and a block to the person at the yard is not three tanks, but one tank with three descriptions. Recognising it as one requires knowing what each description means. Descriptive knowledge concerns what things are and how they relate, and it follows the same ladder as the data structures of Section 2. Variables (a) and lists (b) carry no description of their own: their meaning rests on names and positions understood by convention, outside the data. Tables (c) add a first, flat layer of description, a schema of named attributes that works as a controlled vocabulary, which is what a data dictionary formalises. A classification or taxonomy then sorts things into kinds through is-a relations, and a hierarchy arranges them through part-of relations. Both are trees, structure (d). An ontology goes further: it allows any relation, provided that every term and every relation is defined. It is structure (e), with the meaning agreed.

These terms are themselves used differently in the maritime literature. *Ando and Seppälä (2025)*, for instance, use ontology for the purpose of the data and its use, citing SFI as the most advanced maritime example, and taxonomy for data standards and structure, such as the IGES, STEP, IFC, JT and OCX formats. Neither use is wrong; the difference simply shows that even the vocabulary for describing meaning lacks a shared ontology. This paper uses the terms as defined above.

Ontology, however, means something different for humans and for computers. *Gruber (1993)* defined an ontology as an explicit specification of a conceptualization. Humans hold the conceptualization, but rarely the explicit specification. Experienced professionals resolve meaning from context: who is asking, in which phase, and for what purpose. In the words of *Polanyi (1966)*, we can know more than we can tell. In ship design, much of what is decided early is, as *Andrews (2018)* argues, stylistic by definition, and it is a hard exercise to formalize what cannot be formalized, *Gaspar (2022)*. Reconciling top-down and bottom-up design data remains in the hands of the experienced designer, *Gaspar (2018)*. This is why, despite the many standards available, experienced professionals still "just know" which information to fetch, and from where.

A computer has only what is explicitly specified. Its ontology works almost like a large dictionary in which every entry must be defined, together with a grammar stating how the entries may relate. *Wolanyk (1979)* described the data dictionary as a central repository of information containing standardized descriptions of data, answering what data exist, where they are, how they are structured, and who is responsible for them. It was an early form of this idea.

The central point is that both humans and computers infer from the ontology they hold. The engineer infers from experience, and an AI infers from the ontologies in its training data or its context. Neither can infer beyond what that ontology covers.

The existing maritime standards each cover only part of the domains. Fig.2, from *Bronson et al. (2025)*, compares ISO 10303-215, OCX and IFC4 by domain. Each covers its own region, the overlaps are partial, and none covers all the domains. Geographic location and time series, for instance, appear only in IFC, which comes from the building industry. Training an AI on SFI, VIS, OCX, STEP and IFC would therefore give it several partial ontologies, each valid within its own region and unreliable outside it.

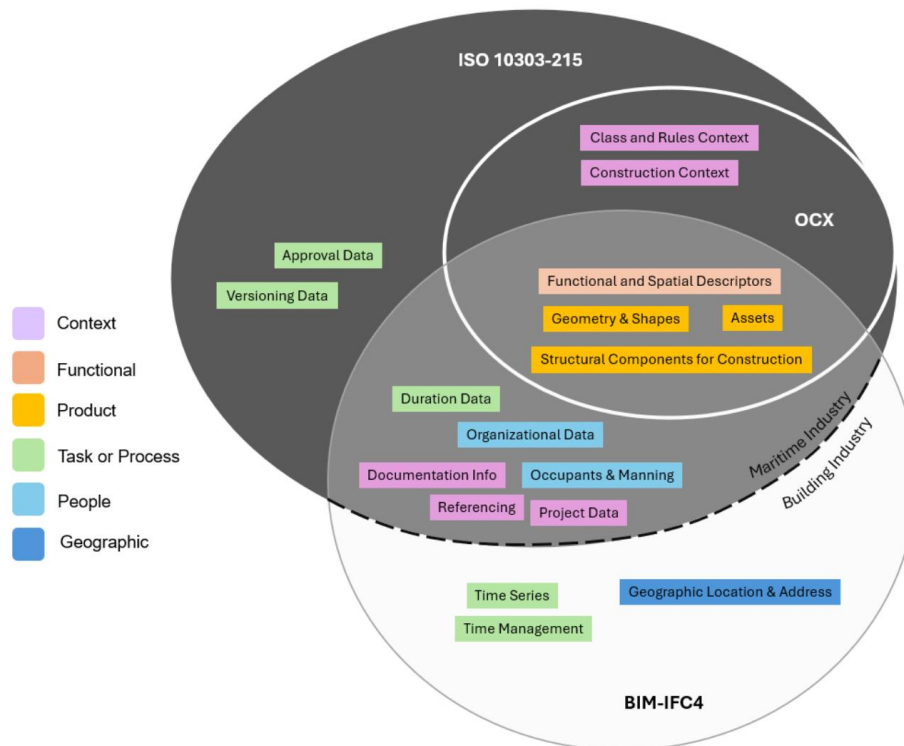


Fig.2: Comparison of domains covered by IFC, STEP and OCX, *Bronson et al. (2025)*

This partial coverage can also be measured. *Astrup and Polini (2025)* mapped a ship-specific hull taxonomy, derived from STEP AP218 and AP215, onto OCX. The format covers about 85% of the taxonomy and of the hull drawing requirements in DNV's class documentation rules, *DNV-CG-0550 (2024)*. A real Ro-Pax model exported to OCX, however, covered only about 59% of those requirements: welds, design loads and lightweight, among others, were missing.

Back to WBT-3P. In taxonomy terms, it is a ballast tank, which is a tank. In hierarchy terms, it is part of the ballast system, SFI 801. In ontology terms, it is also located in Block 12, bounded by steel plates, and filled by BP-01, with each relation defined.

An experienced engineer asked for the volume of WBT-3P knows from context what is meant. For stability it is the net volume at a given filling, and for class it is the tank as tested, while the yard would rather ask for its steel weight and coating area. A computer needs each of these defined separately.

A further illustration came up while preparing this paper. An AI assistant (Claude) was used to discuss and structure the running example. When asked to place (and name) WBT-3P in the SFI system, it could name the SFI main groups, but it could not place the tank until it had searched for the sub-group codes. Once it had them, it inferred that the tank has no single SFI code. Its steel falls under the hull region groups (21–24), its coating under 285, its tank test under 204, and its piping under 801, while

the bought pump BP-01 sits entirely under 801. That inference is valid only within SFI. Even there it depended on which version of the code list was found, since two public sources disagreed on the three-digit codes for tank structure. The AI's inference was exactly as reliable as the ontology it reasoned over: no better, and no more complete. I learnt early in my career that even inside the same design department people do not agree all the time about which SFI code is correct.

5. Prescriptive knowledge: rules, definitions and dictionaries

Section 4 dealt with descriptive knowledge: what things are and how they relate. A second kind of knowledge is just as present in ship design, construction and operation: prescriptive knowledge, which states what should be done or what is allowed. A tank's volume, its position and its connections are descriptive. The requirement that it be tested, coated or kept free of oil is prescriptive.

The distinction is an old one. *Hume (1739)* observed that one cannot derive an 'ought' from an 'is': no amount of describing how things are tells us how they should be. In principle, then, the two kinds of knowledge are logically distinct. In practice they are routinely mixed. Rules need descriptions to apply to; descriptions are drawn up to meet rules; and both end up stored side by side in the same tables, trees and tools. The result is that a classification built to comply with a rule is later read as if it described the ship, and a description is treated as if it carried the rule's authority. This section first explains the gradient of prescriptive knowledge, then the mechanism through which it produces descriptive structure.

Prescriptive knowledge forms a gradient: laws, conventions, class rules, guidelines, best practices and, at the far end, gut feeling. At one end are written rules with explicit definitions. In the middle are guidelines and best practices, partly written. At the other end is the tacit judgement of experienced professionals, whose definitions live only in experience. Gut feeling is not a weakness to be removed: following *Provost and Fawcett (2013)*, *Gaspar (2018)* saw good design decisions as a merge between intuition and the analysis of data.

A rule can only apply to something it has defined. Every law, convention or class rule therefore starts with definitions that set its scope, and those definitions sort things into kinds. In other words, they form a taxonomy. This is how prescriptive knowledge produces descriptive structure. The resulting taxonomy, however, is drawn for a purpose (compliance), not to describe the ship, and each rule draws its own boundaries. The same physical object is therefore classified differently by each rule. Once in use, these rule-made taxonomies circulate in data and tools as if they described the ship itself, and the two kinds of knowledge get mixed up.

Rules are not the only source of definitions. At least four layers of definition sit on top of each other:

1. The prescribed definition, from conventions and class rules.
2. The software definition, from each tool's data model and manual. A stability tool, a structural CAD system and an ERP each define their objects as their own data model needs.
3. The company or project definition: in-house naming, templates and codes.
4. The personal definition: what the experienced professional just knows.

Gaspar (2018) described an earlier version of these layers for ship design data, based on input from Per Olaf Brett (Ulstein Group): overall standards shared by all stakeholders (tender packages, 2D and 3D drawings, office documents, SFI), third-party choices (class rules, CAD and analysis software, supplier and yard information) and proprietary data (the know-how of the designer and the yard). Of the third-party layer, I observed that definitions and standards differ so much that the mapping between one data type to another data type ends up as being a large part of the designer's work.

The software layer deserves attention, because a tool's requirements quietly become prescriptive: the software requires it acts as a rule, although no regulator wrote it. This echoes the warning of *Knapp (1980)*, quoted in *Gaspar (2025)*, of planning to suit the computer rather than the shipbuilding process.

Each pair of layers needs a dictionary to convert from one to the other. With many rules, tools and companies, the number of dictionaries grows much faster than the number of schemes, since every new scheme needs a converter to each existing one. In large projects, up to a hundred different 3D models may be used for different purposes, *Ando and Seppälä (2025)*. Many of these dictionaries are not software but people. Examples are the engineering offices that translate the designer's data to suit a yard's way of working, and the engineers who convert a concept model between CAD and analysis tools, *Gaspar (2025)*. They work because they hold the tacit ontology described in Section 4.

Back to our Ship A example, the ballast tank WBT-3P can be classified differently by each rule:

- Under the Ballast Water Management Convention, what matters is its content, ballast water, defined by purpose as water with its suspended matter taken on board a ship to control trim, list, draught, stability or stresses of the ship (Art. 1).
- Under MARPOL Annex I, if the ship is a tanker, the question is whether the tank holds segregated ballast: ballast water introduced into a tank which is completely separated from the cargo oil and oil fuel system and which is permanently allocated to the carriage of ballast (Reg. 1). The boundary here is system separation, not purpose.
- Under class rules, the tank falls into a category by design load and pressure, used to determine its scantlings.
- In SFI (Section 4), it is split across hull, material protection and ballast system groups.
- In SOLAS probabilistic damage stability, the probability of damage is computed from compartment boundaries. Because that method has known flaws, practice often replaces real compartments with simplified fictitious zones, *Koelman (2023a)*. WBT-3P may then appear in the damage stability model not as itself, but as part of a zone: a taxonomy created by the way a rule is calculated.

(This example was inspired by *Koelman (2023a)*, and the excerpts were also gathered by AI, as I could never know the rules by heart. If correct, they make a good example; if wrong they make the point of the paper even more valid, so a win-win.)

It is one tank in five taxonomies, four of them born from rules. Each tool used along the life-cycle then adds its own definition, and each conversion between them needs a dictionary, written down or held in someone's head.

The mechanism is visible, step by step, in recent work on model-based class approval, *Astrup and Polini (2025)*. Starting from the documentation that DNV's rules require for hull approval, *DNV-CG-0550 (2024)*, the authors identify 73 information elements and map them to OCX through the ship-specific taxonomy mentioned in Section 4. Each taxonomy entry has an identifier, a parent, a label and a description: a dictionary, in the sense of Section 4. They also suggest that classification societies specify which information items they need for approval, which is a rule shaping a taxonomy, explicitly. The tank appears here too, through the tank and capacity plan and the protected tank location drawing required by class. So does Ship A: the midship section drawing must state the ship's length, breadth, depth, draught, block coefficient and service speed. In a real 3D model, however, rule length, moulded summer draught and deadweight were either missing from the model or had no counterpart in OCX. The model held the geometry, but not the rule's definition of L: the $L = 100$ problem of Section 2, now inside a classification requirement.

6. Ongoing efforts: SEUS, MAI and NMRI

6.1. SEUS: PLM-centric and ship-centric approaches

The Smart European Shipbuilding project (SEUS) is a Horizon Europe Innovation Action (2023–2026), coordinated by NTNU. Its consortium brings together academia (NTNU, NHL Stenden, University of Turku), software developers (CADMATIC, CONTACT Software, SARC) and shipyards (Ulstein,

Gondan) from five countries. Its aim is an integrated platform connecting CAE, CAD, CAM and PDM tools for shipbuilding. When the project was presented at COMPIT, *Gaspar et al. (2023)*, several of its starting challenges were, in the terms of this paper, problems of shared meaning:

- Information lost over time: information "disappears between the different design phases, from the initial design to the basic design to construction". This is the life-cycle shift of the domains described in Section 3.
- Company definitions: fabrication methods vary from yard to yard, and "the standard by which ship design drawing packages are prepared and communicated varies greatly".
- Software definitions: mechanical CAD and shipbuilding CAD model the same ship at different levels of detail and with different product structures.
- Human dictionaries: at Ulstein, up to 8% of production time was spent on coordination by foremen and up to 3% on project management, *Agis (2020)*. This is in good part the cost of people converting information between definitions.

The project also tested orthogonal classification, meaning several independent classifications of the same item, against ISO 10303, SFI and the German ship assembly register for PDM master data. In Section 2 terms, that is an attempt to give each item more than one parent. Early in the project, the software partners agreed that a comprehensive general data standard for SEUS does not exist, *Koelman (2023b)*. Two parallel approaches followed: one PLM-centric and one ship-centric.

PLM-centric: linked views for managing the project. CONTACT Software proposed a multi-structure data model for a shipbuilding PLM platform, implemented in CADMATIC Wave, which is built on CONTACT Elements, *Stade et al. (2025)*. Shown in Fig.3, it organises the ship in four parallel structures:

- a functional structure based on SFI (group, detail code, code)
- a spatial structure (deck, zone, compartment/room)
- a manufacturing structure (grand block, block, unit)
- a module structure for free collections, deliberately "without structural purity"

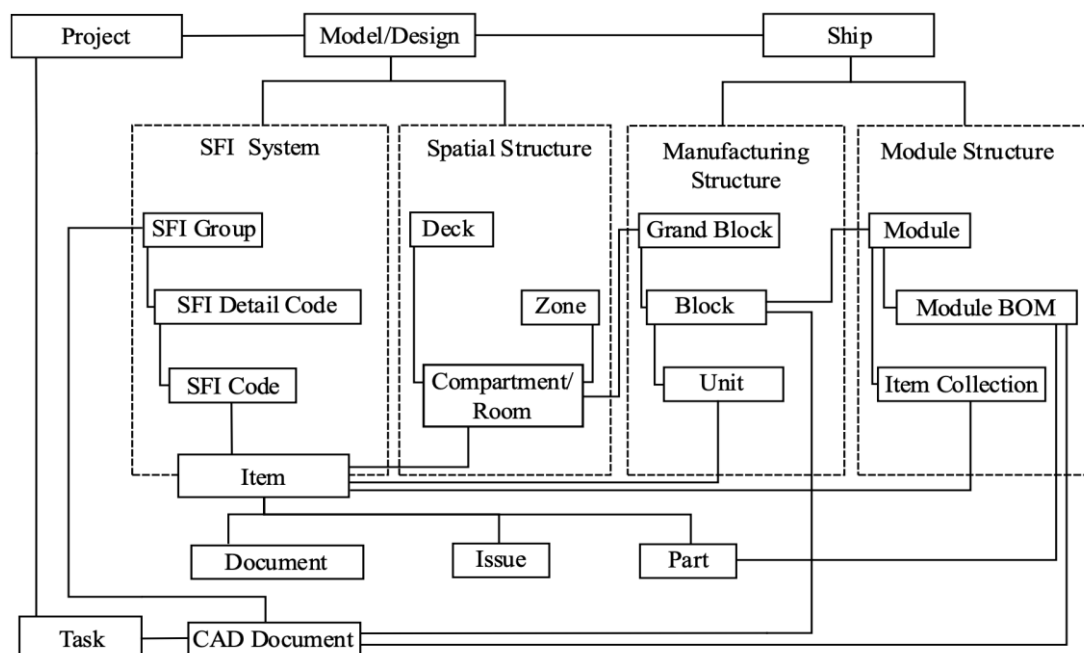


Fig.3: PLM Data model implementation in SEUS (CADMATIC WAVE shipbuilding platform, via CONTACT Elements), from *Stade et al. (2025)*

All four connect to a central Item, a functional location acting as a placeholder. To that Item are linked the supplier-specific Part, documents, issues, CAD documents and project tasks. The relations between levels are named: 'is part of / consists of' for systems, and 'includes / is located in' for spatial and manufacturing structures. Wave has since been taken into use at Ulstein Group, as part of the project's shipyard implementation step.

Ship-centric: engineering objects with facets. SARC proposed starting from the naval architect's objects instead: bulkhead, compartment, pipe segment, *Koelman (2023b)*. Each entity has a name, a unique identifier, links to its parent entities (plural) and its children, and the subsystem it belongs to. On top of that, it carries facets per application, one for each level of detail or discipline. A bulkhead holds its topology and geometry, the compartments on each side, and its panels. PIAS (stability) manages the first two and never sees the panels; CADMATIC uses the first two and manages the panels. A pipe segment is, for stability, connectivity between tanks for damage flooding, and for CADMATIC its geometry, valves, spools and brackets. The entity also carries functions as well as data. A bulkhead's weight is computed by summing its parts rather than stored, and CADMATIC can ask PIAS to compute a tank's volume when compiling a tank plan. This builds on the earlier CADMATIC–PIAS link, which avoided a common file format because then to a very deep level representations of all components have to be the same, or have to be convertible from one to the other. Instead, the two programs send each other commands and questions, acting as two faces of the same system, without exchanging files, *Koelman et al. (2015)*, *Koelman (2023a)*, Fig.4.

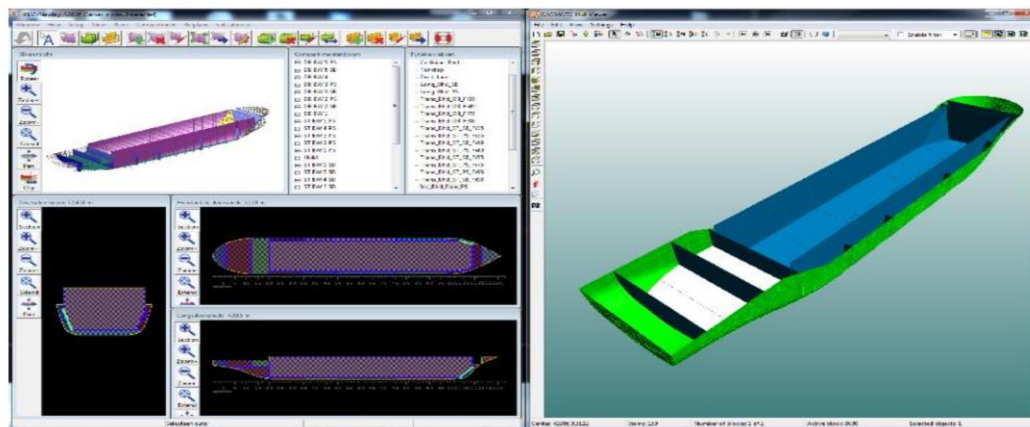


Fig.4: CADMATIC (right) and PIAS (left) working as one system, from *Koelman (2023a)*

The same proposal also illustrates the $L = 100$ problem from Section 2 at its smallest scale: coordinate systems. Length may run forward from the aft perpendicular or, in US practice, aft from the forward perpendicular, and axes may or may not be right-handed. Koelman's suggestion is to give L, B and H their own types, each carrying its convention and conversion functions, rather than storing plain numbers.

Read through our framework, both approaches move from structure (d) toward (e) (see Section 2), but from opposite directions.

- The PLM-centric model splits the ship into views, linked through a shared Item. Its relations are named, and SFI serves as the common dictionary "across disciplines and tools", which gives a shared reference but inherits SFI's partial coverage (Section 4). The authors also recommend labelling data "according to the terminology common in shipbuilding", so the software adapts to the human ontology rather than the reverse (Section 5).
- The ship-centric model gives each object views of itself. Each application owns its facet, and instead of agreeing on one complete representation, the tools agree on which questions they can ask each other. It acknowledges what *Gielingh (2008)* observed, that "the neutral model doesn't really exist".

Neither approach draws a fixed boundary around the ship. The PLM-centric model keeps a module structure deliberately free of structural purity, to collect what does not fit the other views. In the ship-centric model, which application owns which facet shifts with the level of detail and the phase. Both, in other words, accept that the description of a ship cannot be closed in advance, and leave room for it to change.

The running Ship A example in both models:

- **PLM-centric:** BP-01 starts as an Item (the need for a ballast pump under SFI 801, chosen later) and becomes a Part (supplier X's model). WBT-3P appears as a Compartment in the spatial structure, while its steel belongs to a Block in the manufacturing structure. All saved in a CONTACT/CADMATIC database within WAVE.
- **Ship-centric:** WBT-3P is one entity, with a stability facet owned by PIAS (its volume, computed on request), a structural facet owned by CADMATIC (its plates), and links to the pipe segments connecting it to BP-01.

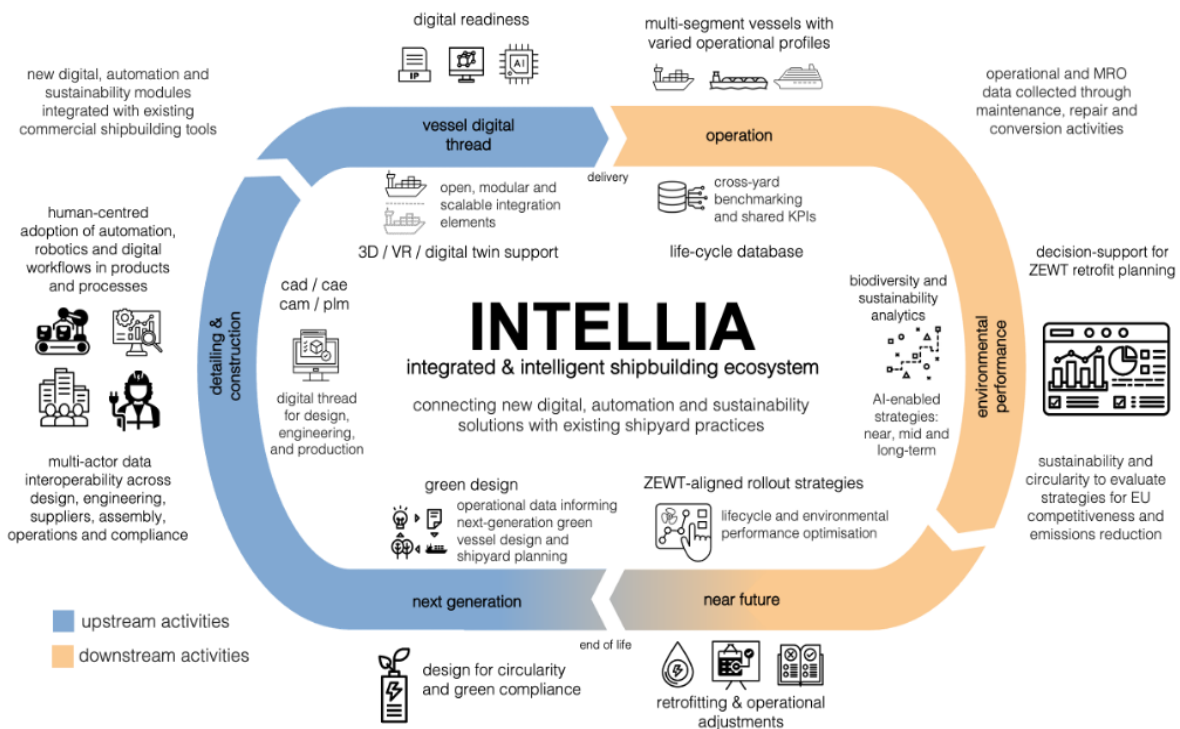


Fig.5: INTELLIA integrated and intelligent shipbuilding ecosystem, connecting upstream and downstream life-cycle activities through a shared vessel digital thread

SEUS continues in INTELLIA (Integration & Intelligence for Europe's Future Shipbuilding - Fig. 5), a Horizon Europe project starting in early 2027 (hopefully), which extends the SEUS platform to pilots in eleven shipyards. One of its objectives is explicitly to map, reconcile and integrate existing data models, ontologies and data standards across the value chain. Its integration layer, the INTELLIA Intelligent Shipbuilding Ecosystem (IISE), is designed to be open, modular and non-intrusive, connecting existing tools through standard interfaces without replacing current ICT infrastructure. In the terms of this paper, the IISE does not impose one shared ontology. It translates between the ones each yard already has, much as the ship-centric approach lets each tool keep its own facet. INTELLIA also plans a shipyard-trained large language model for decision support, a point we return to in Section 7. Fig.5 updates the life-cycle perspective into INTELLIA's vision.

6.2. MAI and NMRI: AI over ship knowledge

The Norwegian Maritime AI Center (MAI, ntnu.edu/mai) is funded by the Research Council of Norway and its partners for 2026–2030, with about 25 PhD positions, seven research partners and about 30 user partners from the maritime sector. Ship design and construction is one of its use cases, which asks how AI technologies, including design automation, ontologies and knowledge graphs, can improve the ship design process.

One of its research questions starts from the observation in Section 3: one idea of a ship exists as many instances at once, and the knowledge connecting design, construction and operation rarely lives in one coherent place. It is scattered across documents, tools, and people, out of reach of the AI methods that could otherwise make efficient use of it. The research calls for comparing ontologies, knowledge graphs, graph databases and multi-taxonomy databases, to see where each fits, where each falls short. The aim is a representation that supports AI-assisted inference: surfacing what else is affected by a change, what may be missing from an incomplete design, or what the likely consequences are elsewhere in the vessel. It also leaves open whether such a representation should fit existing practices and tools, or justify changing them: the tension between software adapting to people and people adapting to software from Section 5.

Where the SEUS approaches design the structure first, the National Maritime Research Institute in Japan explores building it bottom-up with AI, *Matsuo (2025)*. Ship design is still managed at the level of drawings, so the relations between pieces of design information remain largely tacit and person-dependent: the human ontology. The proposed method uses an LLM to extract small design information units from drawings and specifications, to classify them while normalising terminology (e.g. FO Line or SWC Pipe), and to infer directed relations between them, stored in a graph database, Fig.6. The LLM acts thus as a translator, building a dictionary as it goes. Its illustrative engine-room example includes our pump: a ballast system flow-rate requirement determines the ballast pump capacity, which in turn determines the ballast line diameter. A prescriptive requirement drives a descriptive component, which drives another. The challenges listed by the author also map onto this paper. The multiplicity of definitions of what a relation means is the need to agree on relation names (Section 2, structure e). The variability in practice, where dependencies "resist full formalization", is the personal layer of definitions (Section 5). The granularity of information units is the level-of-detail problem raised by *Koelman (2023b)*.

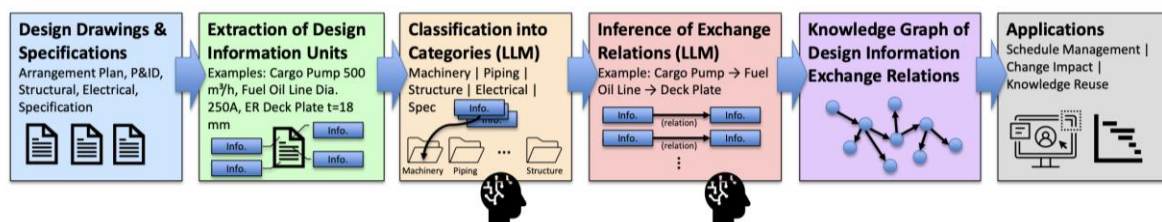


Fig.6: LLM-based construction of design information exchange relations, from *Matsuo (2025)*

7. AI as translator: Who owns the meaning, and how it is kept

The cases above suggest a sequence in how the gap between definitions is bridged, much like the ladder of data structures in Section 2:

- **Dictionaries:** static mappings from one scheme to another, written and maintained by people or by the engineering offices that convert data between tools (Section 5).
- **Parsers:** programs that read one format and write another. They convert syntax, but carry over only the meaning their authors wrote into them.
- **APIs:** agreed interfaces, where tools no longer share a full representation but agree on which questions they may ask each other, as in the CADMATIC–PIAS link (Section 6.1). This is what *Wolanyk (1979)* called data independence: the ability to change a program or to change a

database and not have to change the other. The Model Context Protocol (MCP), suggested by *Ando and Seppälä (2025)* to connect proprietary applications to AI-managed data governance, is a current example of this step, built for the translator that follows.

- Translators: LLMs, which convert between schemes on demand by inferring meaning from context, much as an experienced engineer does.

This has happened before. Explaining why the STEP application protocols never gained broad uptake in shipbuilding, *Astrup and Polini (2025)* list, among other reasons, fears that easy conversion would encourage yards to switch tools, undermining supplier relationships and the concurrent emergence of XML, XSD and 3D API standards that lowered the cost of bespoke, point-to-point translators, lessening the impetus to agree on a single common protocol. LLMs are the same effect, one step further. The first reason is not irrational: economic models of mixed open and proprietary software show that compatibility raises the value created overall, but that incompatibility can pay off for a firm whose modules are of substantially higher quality, *Casadesus-Masanell and Llanes (2010)*, as discussed for the maritime case in *Gaspar (2022)*.

The last step changes the economics of the problem. Much of the case for a unified ontology has rested on the cost of the dictionaries between schemes: people, conversion offices and interfaces, which grow much faster than the number of schemes. *Lincoln et al. (1980)* framed the adoption of computers in shipyards as a question of economics. Seen the same way, if an LLM can produce a converter in minutes, that cost collapses, and with it much of the incentive to agree on one ontology. Each company, tool and person can keep its own way of organising the ship, and the translator bridges them. *Gaspar (2022)* identified the cost of interfaces as a main obstacle to openness in maritime software: in a low-volume, one-of-a-kind market, the cost of developing the APIs and open interfaces, filtering, cleaning, and adapting one model/code to the standards of other's model code rarely pays for itself. That is precisely the cost an LLM translator promises to reduce. This is less a defeat for standardisation than the acknowledgement anticipated in Section 2: incompatibility is included in the process rather than removed.

Current initiatives already point this way. INTELLIA's IISE (Fig. 5) already translates at the level of interfaces, and its planned shipyard-trained LLM extends it to meaning. *Ando and Seppälä (2025)* similarly suggest a dedicated small language model for the maritime industry. The analogy with language translation is useful. Machine translation did not make everyone speak one language: it made doing so unnecessary. It also made errors easier to miss, because a fluent translation looks right whether or not it is. The same holds for ship data. Each LLM conversion is an inference, made on the LLM's own ontology (Section 4). Its errors are silent, and nothing records why 'tank' in one scheme became 'compartment' in another. A human dictionary at least knows when a question has left its territory.

This raises a question that now becomes pressing: how is the dictionary updated? A written standard is revised by a committee, slowly but visibly, with versions. A person's tacit dictionary is updated by experience, invisibly, but within one head that can be asked. An LLM's dictionary changes with each model version and each context it is given, invisibly and at scale. The same conversion may give a different result next year, and no one would be notified. A translator records neither who owns a definition nor who changed it. Software has its own clock as well: CAD, CAE and PLM applications typically last 3 to 5 years, against 25 to 30 for the ship, so practice falls back on simple, established formats, which in turn limits richer representations of topology and relations, *Ando and Seppälä (2025)*.

This leads to a more fundamental question: who owns the ontology, if it is so fragmented? Wolanyk's answer was simple, because his dictionary served one shipyard: its data administrator. Across the life-cycle there is no such owner. Each actor owns a piece of the meaning. Regulators own the prescribed definitions. Classification societies and standards bodies own their schemes. Software vendors own their data models, and, as the history of STEP shows, they may have commercial reasons not to make conversion too easy. Yards and designers own their company codes. The shipowner, the only actor present throughout operation, usually has the least say over any of them. *Bronson et al. (2024)* showed the drastic shift in ownership of ship data at handover. The ship changes hands, but the data and its

meaning often do not. *Ando and Seppälä (2025)* observe that ownership of a data set is not transferred to the next stage but remains with the creator, while only agreed-upon outcomes of the data are transferred to the subsequent phases, creating dead-end routes in the digital thread. They also note the "absence of the main interested party to collect and maintain all related data".

If nobody owns a shared reference, LLM translators fill the gap, and the de facto owner of maritime meaning becomes whoever trains the translator. One possible answer is federated ownership. No single actor owns the whole meaning; each definition stays with its source: the rule with its definition, the vendor with its data model, the yard with its codes. The shared reference owns only the mappings between them, and records where each definition comes from, who is responsible for it, and who changed it. This brings back Wolanyk's question of who is responsible, now moved from one shipyard to the whole life-cycle. Federation does not require that data be given away. As argued in *Gaspar (2022)*, open does not mean free, nor does it mean sharing the totality of the data: a tool with open, well-documented inputs, outputs and interfaces can keep its intellectual property while sharing what is needed to connect it. What is shared is the meaning, the definitions and the mappings, while the data themselves can stay with their owners. Someone must still steward those mappings. The building industry offers one model: IFC is kept by a neutral body, buildingSMART, and was pushed into use by public procurement mandates, *Gaspar (2025)*. *Astrup and Polini (2025)* draw similar lessons from past maritime standards: open governance through ISO or joint industry bodies rather than single vendors, a regular maintenance cycle, and clear IP policies. They also call for a normative ship-specific taxonomy, built one dedicated scope at a time from STEP, SFI and ISO 81346. This paper agrees that such a reference is needed, but suggests that translators may weaken the pressure for it to become the format everyone uses.

What, then, should those who produce ship data do if, for the good or for the worse, AI translation is the future? The answer proposed here is metadata everywhere. Agreeing on a full ontology has proven hard for half a century; agreeing on a minimal set of metadata is much easier. *Gaspar (2022)* made a first version of this proposal with the #hashtag catalogue (Section 2): an element does not need to fit a predefined place in a hierarchy, as long as it has the necessary information to be allocated, searched, and edited. Metadata everywhere extends that idea from tags to declared meaning. If every file and every record had to declare its own meaning, the data structures of Section 2 would no longer depend on names, positions, headers or paths to carry it. A minimal set could include:

- Scheme: which classification or data model the data follows, and which version.
- Units and conventions: including coordinate systems, as in Koelman's typed L, B and H.
- Source of definition: the rule, standard or company code a term comes from.
- Responsible owner: who is accountable for the definition.
- Change history: who changed it, and when.

Three practical consequences follow. First, every object needs a stable identifier across schemes, so that WBT-3P is recognisably the same tank in SFI, in the stability model and in the yard's block list. Second, rule-based classifications should be stored as dated, sourced metadata attached to an object, not as properties of it. That keeps descriptive and prescriptive knowledge apart in the data itself (Section 5). Third, metadata must outlive the software that created it, since the ship outlives its tools several times over.

Part of this already exists. Class rules require every submitted drawing to carry general information such as asset identification, issuer, issue date, measurement units and revision number, *DNV-CG-0550 (2024)*. Model-based definition keeps these management elements in the 3D model rather than in a drawing's title block, *Astrup and Polini (2025)*, after *Quintana et al. (2010)*. Rules can thus be the visible hand that makes metadata mandatory.

With metadata in place, AI translators work through it rather than around it. Each conversion can be checked against declared meaning, and the mappings the translators propose can be approved and stored

as metadata too: explicit, versioned and reviewable. Version control as practised in software development, with tools such as Git, and proposed for ship design documents in *Gaspar (2022)*, offers the model: every change visible, attributed and reversible. AI then writes the dictionary rather than replacing it. Translators can also be tested. The OCX Interoperability Forum already does this for its format translators: twice a year, statistics from the exporting system are compared with those from the importing system, and the results are published, *Astrup and Polini (2025)*. The same could be done for AI translators, re-checking a set of known mappings with each new model version. Over time, the most-used declarations may become a shared vocabulary from the bottom up, without anyone imposing an ontology, much as Dublin Core and schema.org spread on the web. The problem does not vanish, since metadata needs a vocabulary of its own, but it moves to a much smaller and more achievable agreement.

The unified ontology, then, may not disappear, but change role. Rather than the format everyone stores data in, it becomes the reference that translators work through and are checked against: an explicit, versioned dictionary with known owners. That is close to what *Wolanyk (1979)* asked for, with one difference: his dictionary served one shipyard, and the question now is what serves the whole life-cycle. The paper began with $L = 100$, a value whose meaning lives only in its name. It ends with the remedy: never store a value without saying what it means. The inference that AI offers over ship data, surfacing what else is affected, what is missing, and what follows, will be only as reliable as the ontology it reasons over, and only as traceable as the metadata that ontology is kept in.

References

- AGIS, J.J.G. (2020), *Effectiveness in decision-making in ship design under uncertainty*, PhD thesis, NTNU, Trondheim
- ANDERSON, C. (2015), *Creating a Data-Driven Organization*, O'Reilly, Sebastopol
- ANDO, H.; SEPPÄLÄ, L. (2025), *Information architecture in shipbuilding and shipping: Ship lifecycle, digital models, twins, and fragmented data thread*, 23rd COMPIT Conf., Pontignano.
- ANDREWS, D. (2018), *The sophistication of early stage design for complex vessels*, Int. J. Maritime Eng. (Trans. RINA), Special Edition, <https://discovery.ucl.ac.uk/id/eprint/10079021/>
- ASTRUP, O.C.; POLINI, M.A. (2025), *From blueprints to bytes: OCX is the future for model-based class approval*, 23rd COMPIT Conf., Pontignano, pp.314-332
- BRONSON, J.A.; FONSECA, I.A.; GASPAR, H.M. (2024), *Challenges towards an integrated digital twin platform for maritime systems: Tackling shifts in data ownership*, 43rd OMAE Conf., Singapore
- BRONSON, J.A.; GASPAR, H.M.; FONSECA, I.A. (2025), *A markup language for maritime systems: Advancing data interoperability*, 44th OMAE Conf., Vancouver
- CASADESUS-MASANELL, R.; LLANES, G. (2010), *Mixed source*, Harvard Business School Working Paper, Boston
- DNV-CG-0550 (2024), *Maritime services – terms and systematics*, DNV, Høvik
- GASPAR, H.M. (2018), *Data-driven ship design*, 17th COMPIT Conf., Pavone, pp.426-439
- GASPAR, H.M. (2022), *Open and collaborative ship design*, 21st COMPIT Conf., Pontignano, pp.416-428
- GASPAR, H.M. (2025), *A single-source-of-truth for future ship life-cycle data*, 17th HIPER Conf., Tullamore, pp.295-306

- GASPAR, H.M.; SEPPÄLÄ, L.; KOELMAN, H.; GARCIA AGIS, J.J. (2023), *Can European shipyards be smarter? A proposal from the SEUS project*, 22nd COMPIT Conf., Drübeck, pp.407-416
- GIELINGH, W. (2008), *An assessment of the current state of product data technologies*, Computer-Aided Design 40/7, pp.750-759
- GRUBER, T.R. (1993), *A translation approach to portable ontology specifications*, Knowledge Acquisition 5/2, pp.199-220
- HUME, D. (1739), *A Treatise of Human Nature*, John Noon, London
- KNAPP, S. (1980), *Shipyard planning and the computer: Fact or fantasy*, 7th REAPS Technical Symp.
- KOELMAN, H.J. (2023a), *Tell me*, 22nd COMPIT Conf., Drübeck, pp.64-72
- KOELMAN, H.J. (2023b), *Proposal on SEUS data model*, SEUS project workshop, Ålesund, (presentation)
- KOELMAN, H.J.; VAN DER ZEE, J.; DE JONGE, T. (2015), *A virtual single ship-design system composed of multiple independent components*, 14th COMPIT Conf., Ulrichshusen
- LINCOLN, D.E.; BURBANK II, J.A.; McREYNOLDS, J.R. (1980), *Economics of computers in shipyard production control*, 7th REAPS Technical Symp.
- MATSUO, K. (2025), *Constructing design information exchange relations in shipbuilding using large language models*, 23rd COMPIT Conf., Pontignano, pp.150-157
- POLANYI, M. (1966), *The Tacit Dimension*, Doubleday, New York
- PROVOST, F.; FAWCETT, T. (2013), *Data Science for Business*, O'Reilly, Sebastopol
- QUINTANA, V.; RIVEST, L.; PELLERIN, R.; VENNE, F.; KHEDDOUCI, F. (2010), *Will model-based definition replace engineering drawings throughout the product lifecycle? A global perspective from aerospace industry*, Computers in Industry 61/5, pp.497-508
- STADE, D.; IDJEN, M.; BRANDENBURG, E. (2025), *Multi-structure product data management in ship design*, 23rd COMPIT Conf., Pontignano, pp.238-244
- WOLANYK, O.J. (1979), *Exploiting database management systems in shipbuilding*, 6th REAPS Technical Symp., San Diego, pp.529-532

Index by Authors

Albert	77	Otjens	104
Albrechtsen	127	Peippo	206
Aoyama	138	Plowman	15
Bertram	7,15,24	Pudd	206
Bjordal	114	Puurula	206
De Leon	104	Rabliås	127
DeNucci	24	Radonic	93
Favero	189	Ramaswamy	165
Gaspar	234	Ratte	53
Gresens	178	Sandvik	127
Gui	138	Steen	127
Hansen	222	Theiss	42
Hochkirch	222	Van den Berg	189
Hottois	77	Wagner	42
Hympendahl	222	Wakita	64
Ichinose	64	Weyne	77
Khan	165	Wittkop	53
Koelman	104,151	Wölke	53
Korpel	151	Zerbst	42,53,178
Lützenberger	42		
Mitrik	24		
Obic	93		
Oneto	83		
Ostovar	165		

24th Conference on
Computer Applications and Information Technology in the Maritime Industries
COMPIT'27
Mülheim / Germany, 12-13 October 2027

Topics: Artificial Intelligence / CAX Techniques / Digital Twin / Simulations /
Virtual & Augmented Reality / Robotics / Autonomous Technology
In Design, Production and Operation of Maritime Systems

Organiser: Volker Bertram (volker@vb-conferences.com)

Advisory Committee:

Marco Bibulii	CNR, Italy	Inno Gatin	Cloud Towing Tank, Croatia	Jialun Liu	WUT, China
Marcus Bole	CADMATIC, UK	Ken Goh	Knud E. Hansen, Australia	Kohei Matsuo	NMRI, Japan
Jean-David Caprace	COPPE, Brazil	Stefan Harries	Friendship Systems, Germany	Rodrigo Perez	Siemens, Spain
Thomas DeNucci	USCG, USA	Woo-Sung Kil	Korean Register, S. Korea	Ludmila Seppälä	NAPA, Finland
Henrique Gaspar	NTNU, Norway	Herbert Koelman	SARC, Netherlands	Carsten Zerbst	PROSTEP, Germany

Venue: The conference will be held in the “Die Wolfsburg” in Mülheim/Germany



Format: Papers to the above topics are invited and will be selected by a committee.

Deadlines:	anytime	Optional “early warning” of intent to submit paper
	10.6.2027	First round of abstract selection (approx. 1/2 slots)
	10.7.2027	Final round of abstract selection (remaining slots)
	15.9.2027	Payment due for authors
	20.9.2027	Final papers due (60 € surcharge for late submission)
Fees:	700 € / 400 €	regular / PhD student – early registration (by 1.8.2027)
	800 € / 450 €	regular / PhD student – late registration

Fees are subject to VAT
Fees include proceedings, lunches, coffee breaks and conference dinner

Sponsors: Tutech Innovation, (further sponsors to be announced)

Information: volker@vb-conferences.com