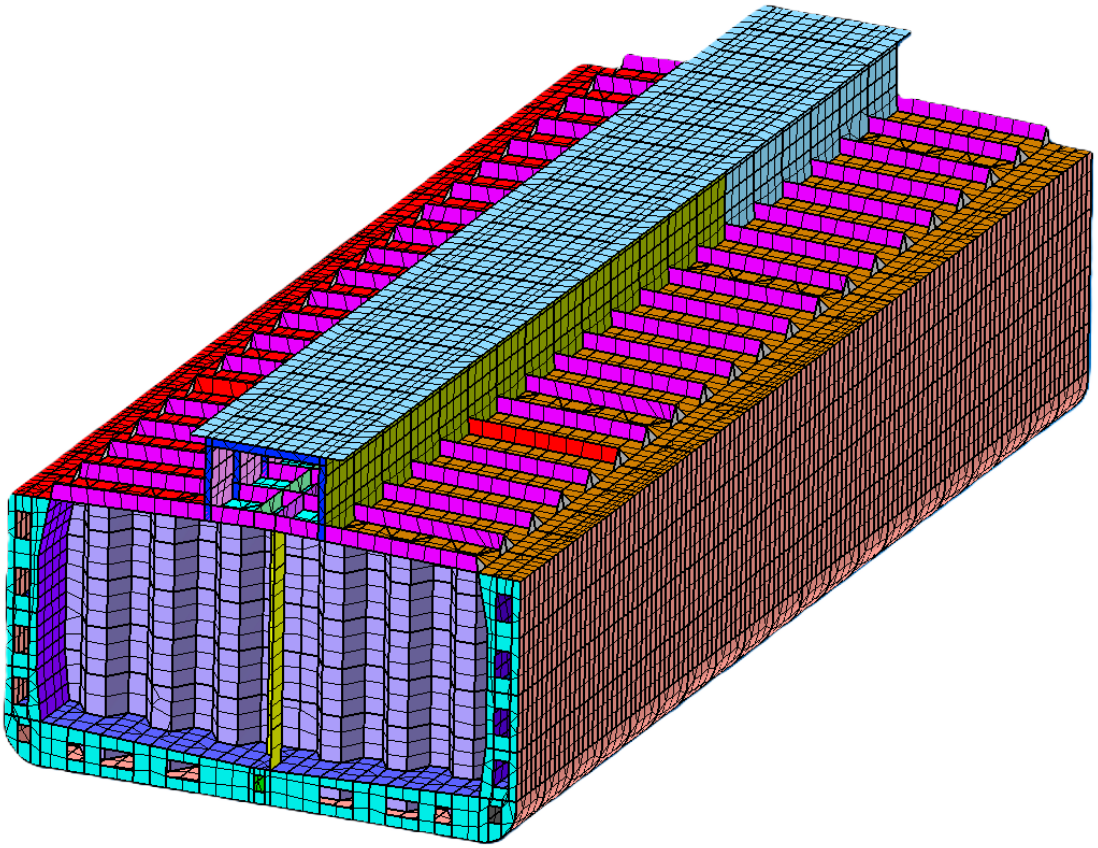
CHALMERS
UNIVERSITY OF TECHNOLOGY



Development of an integrated FEA solution for shipbuilding

Streamlining Compliance with Classification Society Standards

JESPER ARONSSON

DEPARTMENT OF INDUSTRIAL AND MATERIALS SCIENCE
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2025
www.chalmers.se

Master's thesis in Product Development 2025

Development of an integrated FEA solution

Streamlining Compliance with Classification Society Standards

JESPER ARONSSON

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Industrial and Materials Science
Division of Product Development
Chalmers University of Technology
Gothenburg, Sweden 2025

Development of an integrated FEA solution
Streamlining Compliance with Classification Society Standards
JESPER ARONSSON

University Supervisor: Kanishk Bhadani, Department of Industrial and Materials Science
Supervisor: Aron Jalkesen, FKAB Marine Design
Examiner: Gauti Asbjörnsson, Department of Industrial and Materials Science

Master's Thesis 2025
Department of Industrial and Materials Science
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Oil tanker midship finite element model, created in MSC Apex.

Typeset in LaTeX
Printed by Chalmers Digitaltryck
Gothenburg, Sweden 2025

Development of an integrated FEA solution
Streamlining Compliance with Classification Society Standards
JESPER ARONSSON
Department of Industrial and Materials Science
Chalmers University of Technology

Abstract

This thesis presents a Python-driven, MSC Apex–integrated software that automates load-application in alignment with rules specified by classification societies. The software applies bending, shear, torsion, and pressures—directly from spreadsheet inputs to the finite element model. Utilizing an iterative software development process, the solution was systematically built and validated against external benchmarks on oil tanker midship models. The developed software delivers accurate, repeatable results while eliminating manual data transfers and significantly reduce the different interfaces used throughout the finite element analysis. Its modular architecture easily adapts to other vessel types and rule sets and is designed for seamless future integration into Design-For-Compliance (DFC), paving the way for faster, more transparent structural compliance analyses.

Keywords: Maritime, Finite Element Analysis, CSR, IACS, Compliance.

Acknowledgements

This master's thesis was written by Jesper Aronsson in fulfillment of the requirements for the Master of Science degree in Product Development at Chalmers University of Technology, Gothenburg, Sweden.

The project was carried out in collaboration with FKAB Marine Design, Accentus Aero, and Hexagon. Special thanks go to Aron Jalkesen, JJ Joubert, and Stefan Tynnelius for their supervision and for contributing to a engaging and rewarding experience.

Additionally, sincere gratitude is extended to Gauti Asbjörnsson, who served as examiner, and Kanishk Bhadani, who served as supervisor at Chalmers University of Technology. Their guidance and continuous support have been invaluable throughout the thesis work.

Jesper Aronsson, Gothenburg, May 2025

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

BDF	Bulk Data File
CAE	Computer Aided Engineering
CSR	Common Structural Rules
CSV	Comma Separated Value
EDW	Equivalent Design Wave
FEM	Finite Element Method
FEA	Finite Element Analysis
IACS	International Society of Classification Societies

Contents

List of Acronyms **ix**

1 Introduction **1**
1.1 Company background . . . 1
1.1.1 Hexagon . . . 1
1.1.2 Accentus Aero . . . 1
1.1.3 FKAB Marine Design . . . 1
1.2 Problem Statement . . . 2
1.3 Purpose And Objectives . . . 2
1.4 Scope and Limitations . . . 2

2 Theoretical Background **3**
2.1 Naval Architecture . . . 3
2.1.1 Terminology . . . 3
2.1.2 Structural Members . . . 6
2.1.3 Load types . . . 10
2.2 Classification Societies . . . 11
2.3 CSR Regulations . . . 11
2.4 Finite Element Analysis in Shipbuilding . . . 11
2.4.1 Global Hull Girder Model . . . 12
2.4.2 Local Fine-Mesh Models . . . 13
2.4.3 Scantling and Corrosion Deduction . . . 13
2.4.4 Simulating Load Cases . . . 14
2.4.5 Load Scenario . . . 15
2.4.6 Application of Boundary Conditions . . . 16
2.4.7 Load Application . . . 17
2.4.8 Integration and Adjustment of Local Loads . . . 17
2.4.9 Solution and Post-Processing . . . 17
2.4.10 Yield-Strength Assessment . . . 18
2.5 Software Development . . . 18
2.5.1 Requirements Analysis . . . 18
2.5.2 Software Design . . . 18
2.5.3 Implementation . . . 18
2.5.4 Testing and Result Validation . . . 18
2.6 Takeaways . . . 19

3 Development Procedure **21**
3.1 Tools . . . 21
3.1.1 MSC Apex . . . 21
3.1.2 MSC Nastran . . . 21
3.1.3 Design For Compliance (DFC) . . . 22
3.1.4 CSR Finite Element (CSR FE) . . . 23
3.1.5 Raytracing . . . 23
3.1.6 Application programming interface . . . 23
3.1.7 Python . . . 23
3.1.8 Mathcad . . . 24
3.2 Methodology . . . 24
3.2.1 Planning and Concept Development . . . 24
3.2.2 Spiral Product Development Process . . . 25
3.3 Validation of the Method . . . 26
3.3.1 Framework Overview . . . 26

4 Result of the Preliminary study **29**
4.1 Researching Existing Solutions . . . 29
4.2 Observational Study . . . 30
4.2.1 Current FEA workflow at FKAB Marine Design . . . 30
4.2.2 Highlighted issues . . . 32
4.2.3 Geometry vs Mesh . . . 32
4.3 Functionality Specification . . . 32
4.4 Mathcad Prototype . . . 33
4.5 Flowcharts . . . 34

5 Software Development **37**
5.1 Iterative approach . . . 37
5.2 Developing Validation tests . . . 37
5.3 Final Code implementation . . . 38
5.3.1 Input & output interface . . . 38
5.3.2 Data generation . . . 38
5.3.3 Moulded Breadth at the waterline . . . 45
5.3.4 Local loads . . . 45
5.3.5 Load integration . . . 47

6 Results **55**
6.1 Software implementation summary . . . 55
6.2 Streamlined workflow . . . 55

7 Discussion **57**
7.1 Implementation into DFC for Maritime . . . 57
7.2 Validation Using the Validation Square . . . 58
7.3 Strengths and Limitations . . . 59
7.4 Opportunities for Future Work . . . 59
7.5 Summary of the Discussion . . . 60

8 Conclusion **61**

Bibliography **63**

A Appendix 1 **I**

1
Introduction

This chapter establishes the context and motivation for creating an integrated finite element analysis solution for shipbuilding. It introduces the primary industry partners whose tools and expertise shape this work. The problem statement is presented, and the project's purpose and objectives are outlined. The scope of the project is defined, along with its limitations.

1.1 Company background

This section introduces the organizations relevant to this thesis. The key collaborators in the project are: Hexagon AB, Accentus Aero, and FKAB Marine Design.

1.1.1 Hexagon

Hexagon is a global leader in digital solutions, offering tools that enhance productivity and quality in industries such as manufacturing and geospatial analysis[10]. A central product in their portfolio is MSC Apex, an advanced FEA software suite known for its intuitive interface, modular architecture, and integrated Python API. The software enables users to integrate custom scripts and tools, making it a flexible platform for diverse engineering applications.

1.1.2 Accentus Aero

Accentus Aero specializes in developing software solutions for the aerospace sector [13]. Their tool, Design For Compliance (DFC), integrates seamlessly with MSC Apex, automating repetitive tasks, generating certification-ready stress reports, and simplifying the assignment of structural properties. Designed initially for aerospace applications, DFC's functionality demonstrates the potential for adaptation to other industries. Its ability to streamline processes and improve efficiency offers significant promise for shipbuilding, where similar automation and standardization challenges exist.

1.1.3 FKAB Marine Design

FKAB Marine Design is a company within the shipbuilding industry, providing innovative design and engineering solutions for a wide range of vessels [14]. With decades of experience, FKAB specializes in creating efficient and sustainable ship designs tailored to meet client and regulatory requirements. Their operations currently rely on

multiple FEA tools to comply with varying classification society standards. FKAB's collaboration in this project brings invaluable industry insights and practical testing opportunities for the proposed developments.

1.2 Problem Statement

Within the shipbuilding industry, multiple classification societies exist, each enforcing its regulations and providing dedicated finite element analysis software. Consequently, ship design firms must invest in numerous software licenses, master several distinct interfaces, and endure a cumbersome process of exporting and importing ship models between platforms. Current software used are also found to be rather inflexible in their design, offering limited options and feedback for the designer. There is an opportunity to create a unified tool which combines the different rulesets.

1.3 Purpose And Objectives

The purpose of this project was to investigate the feasibility of developing an "all-in-one" FEA software for shipbuilding by creating an extension to MSC Apex, based on DFC software. This new software would facilitate analysis according to the marine classification society standards, reducing reliance on proprietary tools and enhancing workflow efficiency. The objectives are listed below.

- Identify similar software in the shipbuilding industry to understand best practices and potential applications.
- Compile a list of essential features required for an integrated FEA software in shipbuilding, including defining the FE model, mesh policies, and associated functionality.
- Develop a software implementation of the Common Structural Rules.

1.4 Scope and Limitations

The scope of this project includes researching and developing a method to create an integrated load application solution in MSC Apex. The load application is to be conducted per the International Association of Classification Societies (IACS) Common Structural Rules regulation. The project is limited to developing an automated solution for the load application for midship finite element models, with the loading scenario specified as seagoing. The project prioritizes developing a modular, structured solution that will be easy to further add to in future projects.

2

Theoretical Background

This chapter presents a literature review that establishes the theoretical foundation for the work conducted in this thesis. It draws on established knowledge from the fields of naval architecture, structural engineering, and software development, with a particular focus on the Common Structural Rules (CSR) defined by the International Association of Classification Societies (IACS). The chapter begins with an overview of ship structural design and terminology, followed by a description of the CSR framework, its requirements, and how finite element analysis (FEA) is applied to ensure compliance. Finally, it outlines principles of software development relevant to implementing the CSR in an engineering software environment. Together, these sections provide the necessary context for understanding the methodology and implementation described in later chapters.

2.1 Naval Architecture

Naval architecture is the discipline concerned with the design, construction, and analysis of ships and marine structures. From a structural perspective, a ship is idealised as a hull girder, a beam-like structure extending continuously from bow to stern. This hull girder must be capable of withstanding a wide range of forces throughout the vessel's operational life, including hydrostatic pressure, wave-induced loads, cargo weight, and dynamic effects caused by ship motion and sea conditions. The hull must also provide sufficient strength under accidental scenarios such as grounding or collision, while maintaining structural integrity and water tightness. In this section, terminology, structural members, and load types specific to naval architecture will be presented.

2.1.1 Terminology

To describe a ship's geometry and motions, naval architects use a right-hand, ship-fixed coordinate system. This subsection defines the most common terms as described by Rawson and Tupper (2001) [5].

2.1.1.1 Principal Particulars

Principal particulars are the fundamental reference measurements that describe a ship's overall dimensions. These measurements are used to define the ship's size, shape, displacement, and other geometric and hydrostatic properties. Common principal particulars include:

- **Length Between Perpendiculars**: The horizontal distance between the forward perpendicular (FP) and aft perpendicular (AP). The FP is typically located where the summer load line intersects the bow, and the AP is usually located at the centerline of the rudder stock (see figure: 2.1

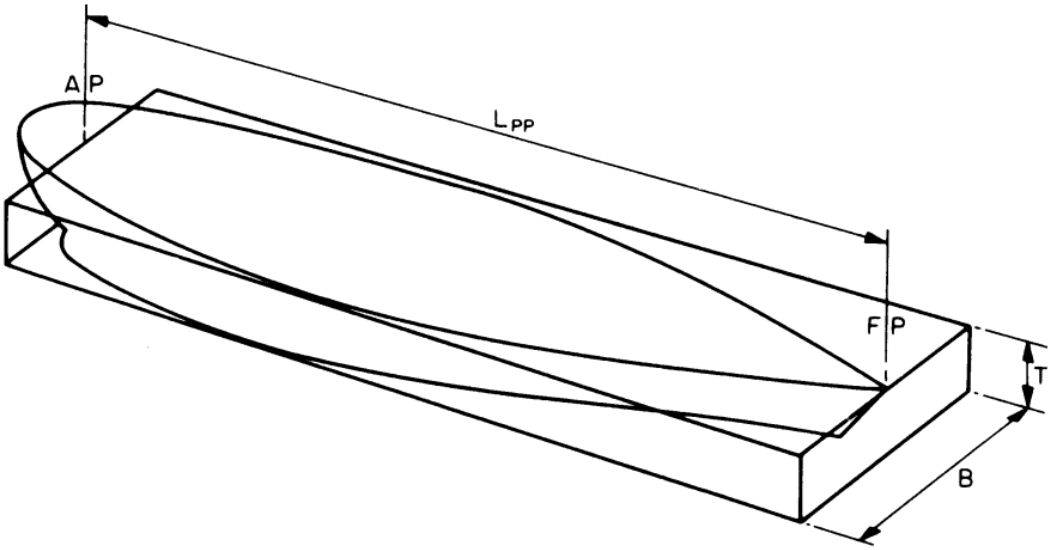


Figure 2.1: Figure showing the Length Between Perpendiculars, denotedL_{PP} [5].

- **Moulded Breadth**: The maximum breadth of the ship measured at the widest point of the hull, taken between the inner sides of the shell plating (inside the steel, see figure: 2.2).

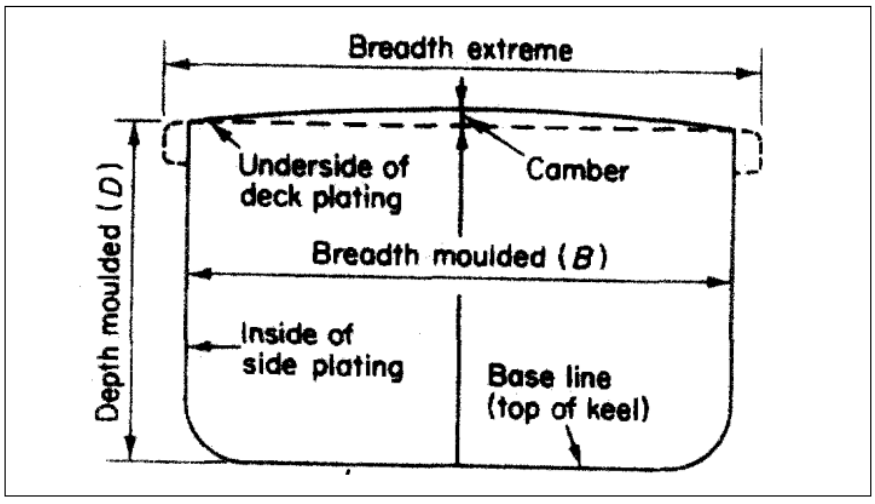


Figure 2.2: Figure showing the Moulded Breadth, denoted B [26].

- **Moulded Depth**: The vertical distance measured from the baseline (top of keel) to the top of the freeboard deck beam at the side. Like moulded breadth, this excludes the thickness of the shell plating (see figure: 2.3.

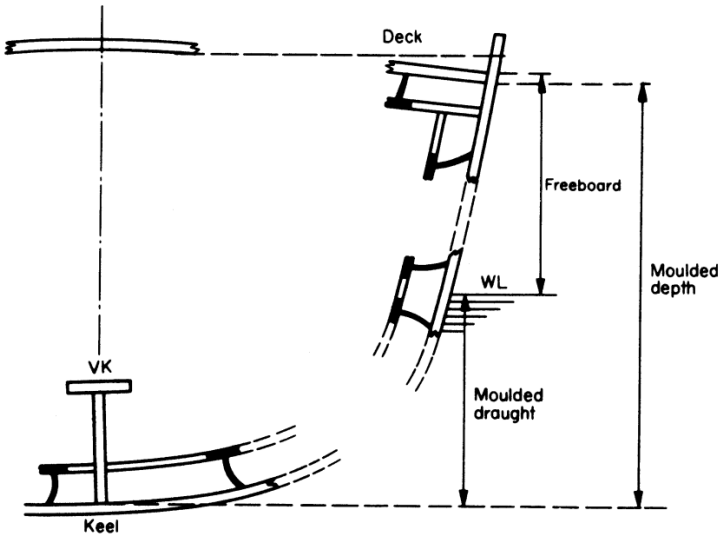


Figure 2.3: Figure showing the Moulded Depth and Moulded Draught [5].

- **Draught (T)**: The vertical distance between the baseline and the ship's waterline. It indicates how deeply the ship sits in the water and varies with loading conditions. Figure 2.3 shows the moulded draught, which equals the draught minus the thickness of the shell plating.
- **Block Coefficient**: A non-dimensional ratio of the underwater volume of the ship to the volume of a rectangular block with the same length, breadth, and draught.

These values are typically summarised early in any ship design project and serve as the foundation for structural design, stability assessment, propulsion requirements, and compliance with classification society rules.

2.1.1.2 Ship axes

The "X-axis" is called the longitudinal axis, and runs from fore to aft end along the ship's centerline. The positive direction is defined as towards the fore end of the ship. The "Y-axis" is called the transverse axis, which runs across the ship from port to starboard. The "Z-axis" is the vertical axis which extends upwards from the keel through the deck with a positive direction upwards.

2.1.1.3 Ship Motions

To describe the ship's motions in its degrees of freedom is described using standard terminology, as can be seen in figure 2.4.

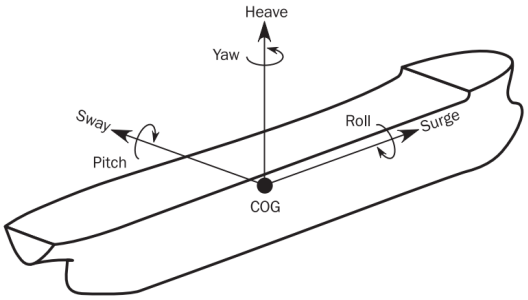


Figure 2.4: Figure describing the definition of positive shipmotions [2].

- Translational Motions
  - **Surge**: Translation along the longitudinal (X) axis (forward/backward).
  - **Sway**: Translation along the transverse (Y) axis (side-to-side).
  - **Heave**: Translation along the vertical (Z) axis (up/down).

- Rotational Motions
  - **Roll**: Rotation about the longitudinal (X) axis (tilting port/starboard).
  - **Pitch**: Rotation about the transverse (Y) axis (bow/stern up/down).
  - **Yaw**: Rotation about the vertical (Z) axis (heading change).

2.1.1.4 Ship Geometrical Measurements

In ship design and drafting, specific terminology is used to locate any feature in three dimensions. The hull is divided longitudinally into a series of *frames*, each corresponding to a cross-section at a specified distance from a reference point (typically the aft perpendicular). Vertically, positions are defined by *waterlines*, which are horizontal planes at given heights above the baseline. Transversely, locations are given by *buttock lines*, which are vertical planes parallel to the centerline.

2.1.2 Structural Members

The primary load-bearing components of a ship's structure are collectively referred to as structural members. Typically, a type of structural member is defined as belonging to one of two categories: primary structural member or secondary support member. Their arrangement, geometry, and material properties govern the vessel's ability to withstand loads such as dynamic wave loads and the weight of the cargo without yielding or buckling. In this section, the main classes of structural members are defined and their functions outlined as defined by Eyres and Bruce (2023) [25].

2.1.2.1 Girders

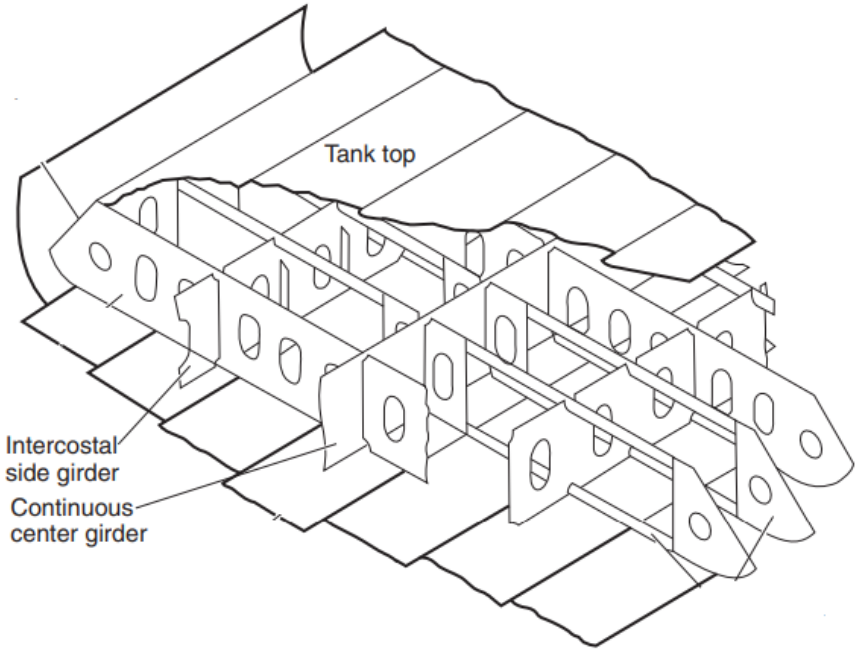


Figure 2.5: Figure showing an example of girders located at the bottom of the vessel [25].

Girders (see figure: 2.5) are primary supporting members often welded to hull, deck or bottom of the vessel. They are major longitudinal or transverse beams that carry bending moments and other loads form deck and superstructure into the hull girder.

2.1.2.2 Stringers

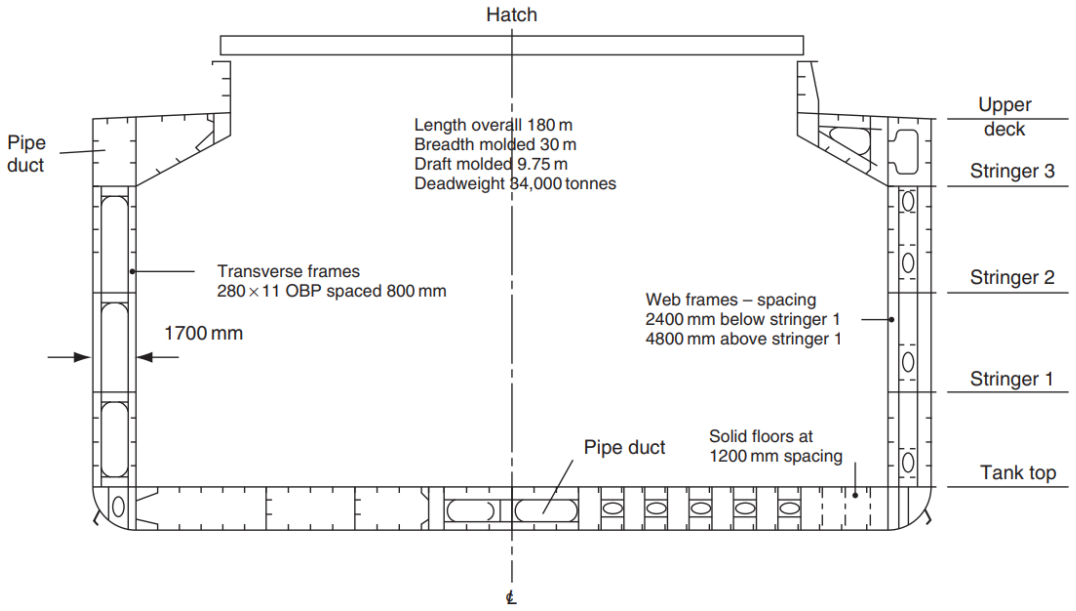


Figure 2.6: Figure showing stringers in a technical drawing of a cross-section [25].

Stringers (see figure: 2.6) are horizontal stiffeners, also counted among the primary supporting members. They are placed at the vessel's side shell plating, and for a double hull construction, they are used for both inner and outer shells.

2.1.2.3 Webframes

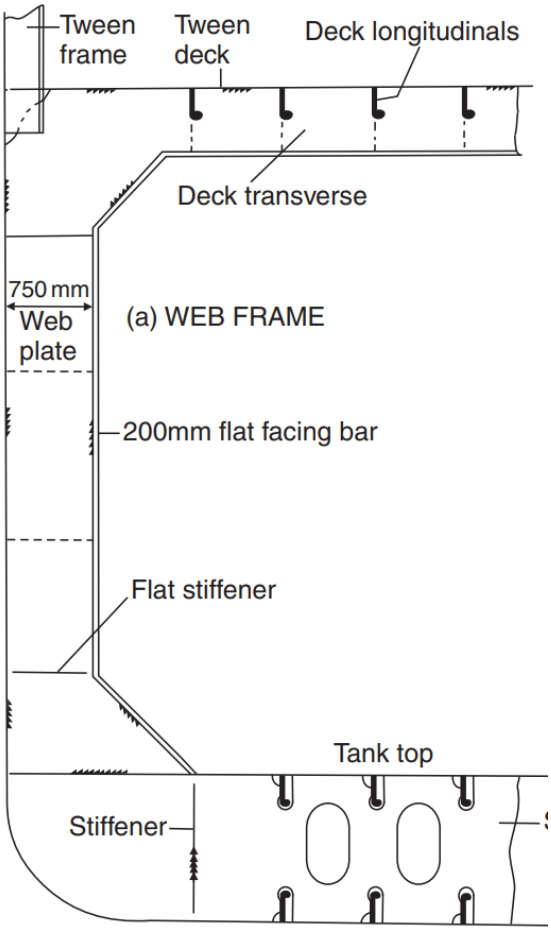


Figure 2.7: Figure showing technical drawing of a cross-section with a web frame [25].

Web frames (see figure: 2.7) are transverse ribs spaced along the hull of the vessel. They are primary supporting members that define the shape of the hull and stiffen the plating against bending and shear.

2.1.2.4 Bulkheads

Bulkheads are primary supporting members that subdivide the hull into watertight or structural compartments by vertical division. They reinforce transverse stiffeners and support longitudinal members. A subtype of bulkheads commonly used to divide cargo holds into multiple compartments is corrugated bulkheads. They utilize a corrugated profile either vertically or horizontally to increase out-of-plane stiffness and buckling resistance without adding too much weight.

2.1.2.5 Deck

The deck serves as the main working platform but also as a significant primary supporting member contributing to the hulls bending stiffness.

2.1.2.6 Secondary supporting members

2.1.2.7 Webframes

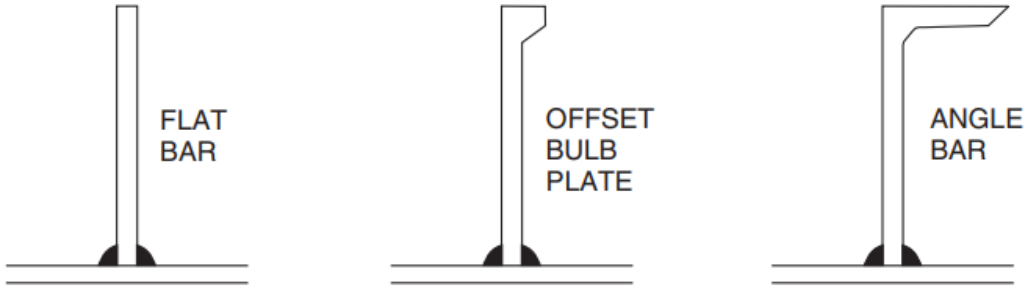


Figure 2.8: Figure showing examples of typical stiffener profiles [25].

The secondary supporting members include stiffeners, pillars, tripping brackets, and more. Stiffeners are used to reinforce plating to prevent local buckling. They typically come in flat, L, T, or bulb profiles (see figure: 2.8).

2.1.3 Load types

When developing a new vessel, the designer needs to make sure that the hull girder can withstand the various forces that it is exposed to during its lifespan, which is typically at least 25 years. The main types of loads can be categorized as static and dynamic loads. There are also accidental loads that will need to be accounted for, such as grounding and collision.

2.1.3.1 Static Loads

Static loads are those that arise from the gravity and buoyancy acting to the vessel [2]. A typical oil tanker has the bulk of the cargo weight distributed towards the longitudinal centre of the ship, meaning that the ship will be subject to a bending moment. It can be described as the ship being a idealized as a simple beam resting on elastic supports (buoyancy). This is know as still water bending moment. Static loads also include the weight of the structure itself.

2.1.3.2 Dynamic Loads

Dynamic loads are those caused by sea waves [2]. These loads act both inside the ship from liquid cargo and ballast, and on the exterior hull of the vessel. These hydrodynamic loads create hogging and sagging of the hull girder. Hogging is when the midship is lifted by a wave crest whilst the aft and fore end remain in troughs, with sagging condition being the inverse.

2.2 Classification Societies

Classification societies are independent organizations responsible for setting and maintaining technical standards for the design, construction, and operational maintenance of ships and offshore structures. Their primary role is to ensure the structural integrity, safety, and seaworthiness of vessels throughout their lifespan. Ships must comply with the rules of a classification society to obtain a class certificate, which is often required for insurance, port entry, and commercial operation.
One of the main organizations coordinating efforts between the major classification societies is the International Association of Classification Societies (IACS). IACS was founded to promote uniformity and consistency in the interpretation of technical rules and standards. Its members include the largest and most influential societies in the industry. The unified set of technical rules and standards is called the Common Structural Rules (CSR).

2.3 CSR Regulations

The IACS CSR applies to Oil Tankers over 150 meters in length and for Bulk Carriers over 90 meters in length. The rules cover structural requirements for [2]:

- Global hull girder strength
- Local scantling of plating and stiffeners
- Buckling
- Yielding
- Fatigue assessment
- Corrosion protection

The CSR contains general rules applicable to both ship types, as well as separate, ship-type specific rules for oil tankers and bulk carriers. Additionally, it provides detailed guidelines for how FEA should be carried out to comply with the structural assessment requirements.

2.4 Finite Element Analysis in Shipbuilding

Finite Element Analysis takes a central role in the CSR's [2] direct strength assessment, providing means to verify both global hull-girder behavior and detailed local strength criteria. As laid out in the CSR, FEA is structured into three complementary stages:

1. Global hull-girder model, to assess bending, shear, and torsional response of the hull girder and primary supporting members over a three-hold "midship" model.
2. Local fine-mesh models, targeting areas prone to stress concentrations such as hopper knuckles, brackets or double-bottom intersections, with higher mesh resolution.

3. Very fine-mesh (fatigue) analysis, where detailed “hotspot” stresses feed into the fatigue assessment.

2.4.1 Global Hull Girder Model

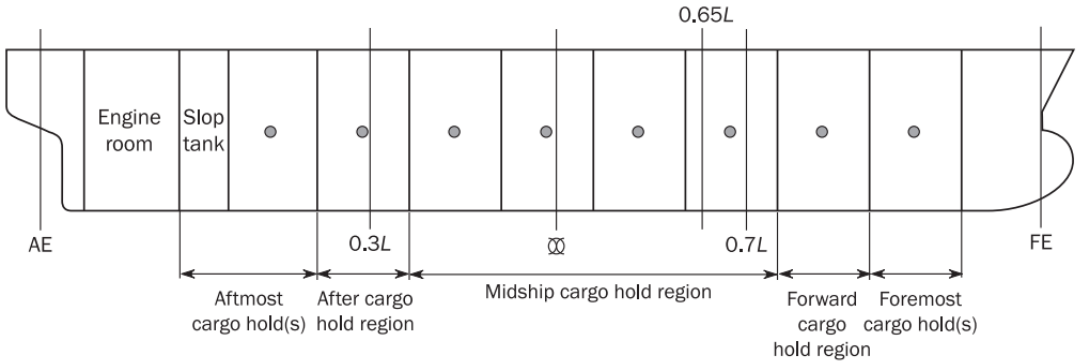


Figure 2.9: Figure showing definitions of cargo hold regions for structural assessment [2].

A typical global FE model spans three adjacent cargo holds and includes all main structural elements, outer and inner shell plating, decks, double-bottom girders, primary stiffeners, and bulkheads, subjected to the full static and dynamic load combinations defined [2]. This midship model (see figure: 2.9) is used to verify longitudinal strength criteria (yielding and buckling) across the full cross section under still-water bending, wave induced hogging and sagging moments, shear forces and torsional loads.

2.4.2 Local Fine-Mesh Models

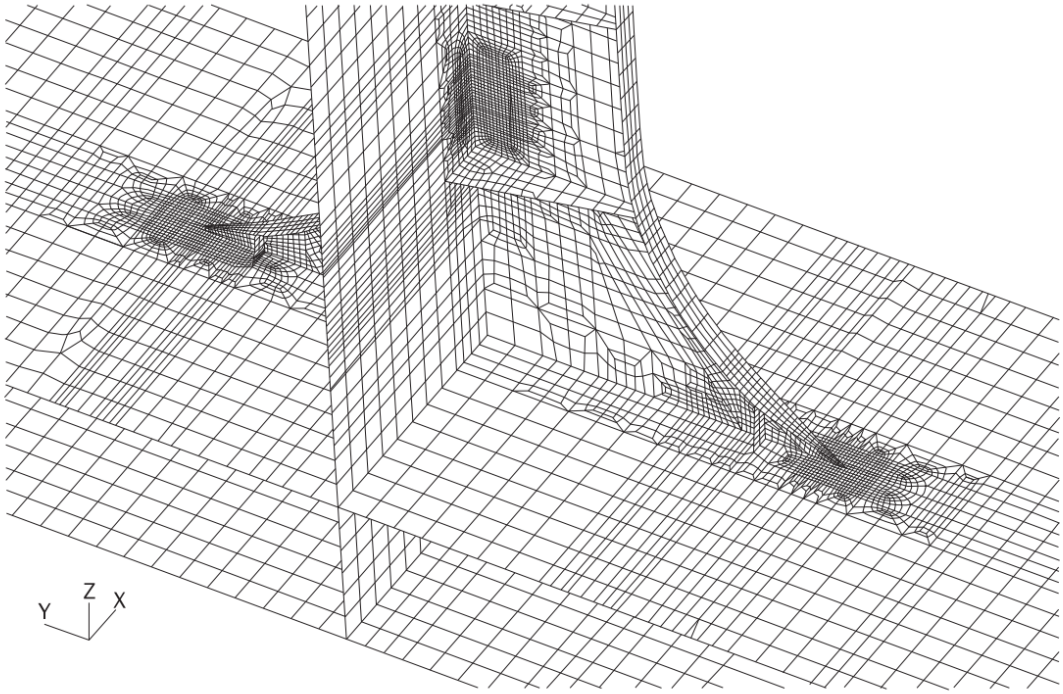


Figure 2.10: Figure showing examples locations with finer mesh applied [2].

Where global analysis identifies potential hotspots, focused fine-mesh models are created around critical details (see figure: 2.10). These include, for example, the hopper knuckle region, bracket toes, web-frame intersections, and double-bottom discontinuities. Global FEA results supply the boundary conditions (nodal displacements and average section forces) for each finemesh slice, ensuring consistency with the overall structural response.

2.4.3 Scantling and Corrosion Deduction

In ship structural design, scantlings refer to the dimensions of structural members such as plates, stiffeners, girders, and bulkheads [6]. These dimensions are selected to ensure the structure can withstand expected loads during the vessel's operational life, including static, dynamic, and accidental loading scenarios.

To account for material loss due to corrosion over time, the net scantling approach is adopted under the CSR. This involves deducting predefined corrosion margins from the gross thickness of structural members to determine their net thickness, which is then used in strength, buckling, and fatigue assessments.

Corrosion deductions vary depending on the location and exposure of the structure. For example, structural elements exposed to ballast water, cargo, or seawater are assigned higher corrosion additions than those in protected internal spaces. These

values are specified in the CSR and must be incorporated in both manual calculations and finite element models to ensure regulatory compliance and long-term structural integrity [2].

2.4.4 Simulating Load Cases

To capture wave effects, the CSR employs the Equivalent Design Wave (EDW) concept: a deterministic wave profile calibrated to induce the most critical hull-girder bending, shear, and torsion for each dynamic load case [3]. The EDWs used in the load cases are as follows:

- HSM: Maximizes vertical bending moment in head seas.
- HSA: Maximizes acceleration in head seas.
- FSM: Maximizes vertical bending moment in following seas.
- OST: Maximizes torsion in oblique/stern quartering seas.
- OSA: Maximizes acceleration in oblique/bow quartering seas.
- BSP: Maximizes side shell pressure at water line in beam seas.
- BSR: Maximizes roll in beam seas.

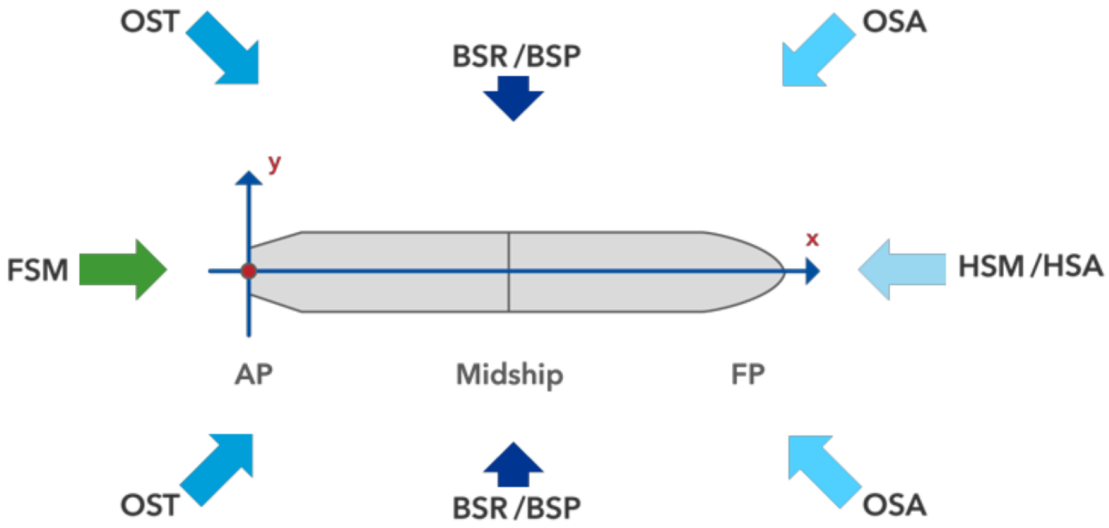
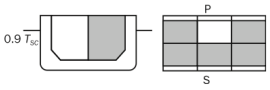
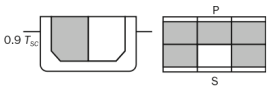
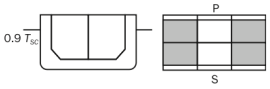


Figure 2.11: Graphic showing the Equivalent Design Waves [3].

Beam sea refers to waves approaching the vessel from the transverse (sideways) direction, whereas head sea indicates waves approaching from the longitudinal (forward) direction, and following sea is the opposite of head sea. The EDW:s are also denoted with either -1 or -2 (HSM-1, HSM-2 as an example), meaning the loadcase in hogging or sagging scenario. Each EDW case is combined with specific loading patterns, i.e. which cargo holds are full, to represent the full spectrum of service conditions from ballast-exchange to fully laden scenarios as specified in the CSR [2]. Below, in table 2.1 are a few examples of loading patterns.

Table 2.1: Sample of load patterns for oil tankers with one centerline bulkhead [2].

No.	Loading pattern	Still water loads			Dynamic load cases		
		Draught	$C_{BM\text{-}LC}$: % of perm. SWBM	$C_{SF\text{-}LC}$: % of perm. SWSF	Midship cargo region		
Seagoing conditions							
B1		$0.9T_{SC}$	100% (sagging)	100%	HSM-1 HSA-1	BSP-1P/S	N/A
			100% (hogging)	100%	HSM-2 FSM-2	BSR-1P BSP-1P	OST-2P
B2		$0.9T_{SC}$	100% (sagging)	100%	HSM-1 HSA-1	BSP-1P/S	N/A
			100% (hogging)	100%	HSM-2 FSM-2	BSR-1S BSP-1S	OST-2S
B3		$0.9T_{SC}$	100% (hogging)	100% [3] Max SFLC	HSM-2 FSM-2	N/A	N/A
				100% [4] Max SFLC	HSM-2 FSM-2	N/A	N/A
				100%	N/A	BSP-1P/S	N/A
			0%	100% [5] Max SFLC	HSM-1 FSM-1	N/A	N/A

2.4.5 Load Scenario

The load cases are additionally referenced to as belonging to a load scenario. The load scenario describes whether the ship is seagoing, harbored condition, ballast exchange, or accidental conditions. Depending on the load scenario, different loads are applied to the FE model. As an example, no dynamic loads should be applied when the load case specifies the harbored scenario (see table: 2.2).

Table 2.2: Load application for different loading scenarios for strength assessment [2].

Design load scenario			Harbour and sheltered water and testing	Seagoing conditions with extreme sea loads	Ballast water exchange (4)	Accidental flooded conditions (4)	
Load components			Static (S)	Static + Dynamic (S+D)	Static + dynamic (S+D)	Static (A: S)	Static + dynamic (A: S+D)
Hull Girder		VBM	$M_{sw\text{-}p}$	$M_{sw} + M_{wv\text{-}LC}$	$M_{sw} + M_{wv\text{-}LC}$	$M_{sw\text{-}f}$ (2)	$M_{sw\text{-}f} + M_{wv\text{-}LC}$ (3)
		HBM	-	$M_{wh\text{-}LC}$	$M_{wh\text{-}LC}$	-	$M_{wh\text{-}LC}$ (3)
		VSF	$Q_{sw\text{-}p}$	$Q_{sw} + Q_{wv\text{-}LC}$	$Q_{sw} + Q_{wv\text{-}LC}$	-	$Q_{sw\text{-}f} + Q_{wv\text{-}LC}$ (3)
		TM	-	$M_{wt\text{-}LC}$	$M_{wt\text{-}LC}$	-	-
Local Loads	P_{ex}	External deck for green sea	-	P_D	-	-	-
		Hull envelope	P_S	$P_S + P_W$	$P_S + P_W$	-	-
	P_{in}	Ballast tanks (1)	$Max\ (P_{ls}, P_{ST})$	$P_{ls} + P_{ld}$	$P_{ls} + P_{ld}$	-	-
		Liquid cargo tanks			-	-	
		Other tanks			-	-	
		Watertight boundaries	-	-	-	P_{fs}	$P_{fs} + P_{fd}$
		Cargo holds	P_{bs}	$P_{bs} + P_{bd}$	-		
	P_{dk}	Internal decks for dry spaces	$P_{dl\text{-}s}$	$P_{dl\text{-}s} + P_{dl\text{-}d}$	-	-	-
		External deck for distributed loads	$P_{dl\text{-}s}$	$P_{dl\text{-}s} + P_{dl\text{-}d}$	-	-	-
		External deck for heavy units	$F_{U\text{-}s}$	$F_{U\text{-}s} + F_{U\text{-}d}$	-	-	-

2.4.6 Application of Boundary Conditions

In each three-hold model, the fore and aft model ends are constrained via point-type supports. These are applied at bulkhead-stool intersections and the neutral axis on the inner bottom and centreline bulkhead. This minimal restraint allows realistic bending and torsion of the model while avoiding stress peaks at the boundaries (see table: 2.3).

Table 2.3: Boundary condition as per the CSR for midship model [2].

Location	Translation			Rotation		
	δ_x	δ_y	δ_z	θ_x	θ_y	θ_z
Aft End						
Independent point	-	Fix	Fix	$M_{T\text{-}end}$	-	-
Cross section	-	Rigid link	Rigid link	Rigid link	-	-
	End beam					
Fore End						
Independent point	-	Fix	Fix	Fix	-	-
Intersection of centreline and inner bottom	Fix	-	-	-	-	-
Cross section	-	Rigid link	Rigid link	Rigid link	-	-
	End beam					

2.4.7 Load Application

Loads are introduced as follows:

- Internal loads: hydrostatic pressures in cargo tanks and ballast spaces, dry-cargo pressures where applicable, and prescribed sloshing loads.
- External loads: still-water hull-girder loads, wave pressures on shell and deck, and slamming pressures in bow sections.
- Gravity: self-weight of all modeled structural components.

These loads are discretized into equivalent nodal forces and distributed frame loads.

2.4.8 Integration and Adjustment of Local Loads

Following CSR requirements, local loads such as hydrostatic pressures, tank loads, and cargo loads are applied directly to the finite element model based on the load cases. However, these local distributions may not, on their own, reproduce the global bending moment, shear force, and torsional moment magnitudes required for each load case. To ensure consistency with the prescribed global load envelopes, balancing forces and moments are applied at the model boundaries (typically at bulkheads or end structures). This adjustment does not alter the local load application but supplements it to match the global force and moment values derived from longitudinal strength calculations.

2.4.9 Solution and Post-Processing

Once boundary conditions and loads are applied, the FE solver computes nodal displacements and, by extension, element stresses. In post processing, stresses are compared with the permissible stress limits for yielding and checked against buckling utilization factors. Local finemesh results inform whether plate panels or stiffened

panels meet the buckling acceptance criteria, while very fine-mesh “hotspot” results feed into fatigue life calculations [2].

2.4.10 Yield-Strength Assessment

Finally, all von Mises and principal stresses are evaluated against the net-scantling yield limits (taking corrosion additions into account). Any exceedance prompts a redesign, either locally (such as increased stiffener size or plate thickness) or globally (via section modulus adjustment) to ensure full compliance with the CSR requirements [2].

2.5 Software Development

Software development in an engineering context involves the systematic design, implementation, and testing of computational tools that solve domain-specific problems or automate workflows [7]. In structural engineering, software is essential for performing detailed calculations, executing simulations, and managing large datasets. To ensure reliability, maintainability, and usability, the development process typically follows a structured methodology comprising several key phases.

2.5.1 Requirements Analysis

This phase involves gathering and defining the functional and technical requirements of the software. It includes identifying user needs, specifying expected inputs and outputs, and outlining any performance or integration constraints. Clear and well documented requirements form the foundation for all subsequent stages.

2.5.2 Software Design

In the design phase, the software architecture is outlined. This includes decisions on programming language, data structures, algorithms, and external libraries or tools to be used. The design also defines how different components will interact, how user interfaces will behave, and how data will flow through the system.

2.5.3 Implementation

During implementation, the software is developed according to the design specifications. Developers write the source code using the chosen programming language and tools. Emphasis is placed on modularity, clarity, and adherence to coding standards to facilitate future maintenance and scalability.

2.5.4 Testing and Result Validation

Once the software is implemented (and throughout the implementation), it undergoes rigorous testing to verify that it behaves as intended. Testing may include unit tests, integration tests, and system-level tests. Validation ensures that the software

produces accurate and consistent results, often by comparing outputs with known solutions, analytical models, or benchmark data.

2.6 Takeaways

To conclude the literature review and to summarize, a mind-map of the required input/output parameters was made (see figure: 2.12). Opportunities primarily aimed at automating certain inputs into the software were made evident, along with challenges such as how to apply loads to the correct nodes throughout the model.

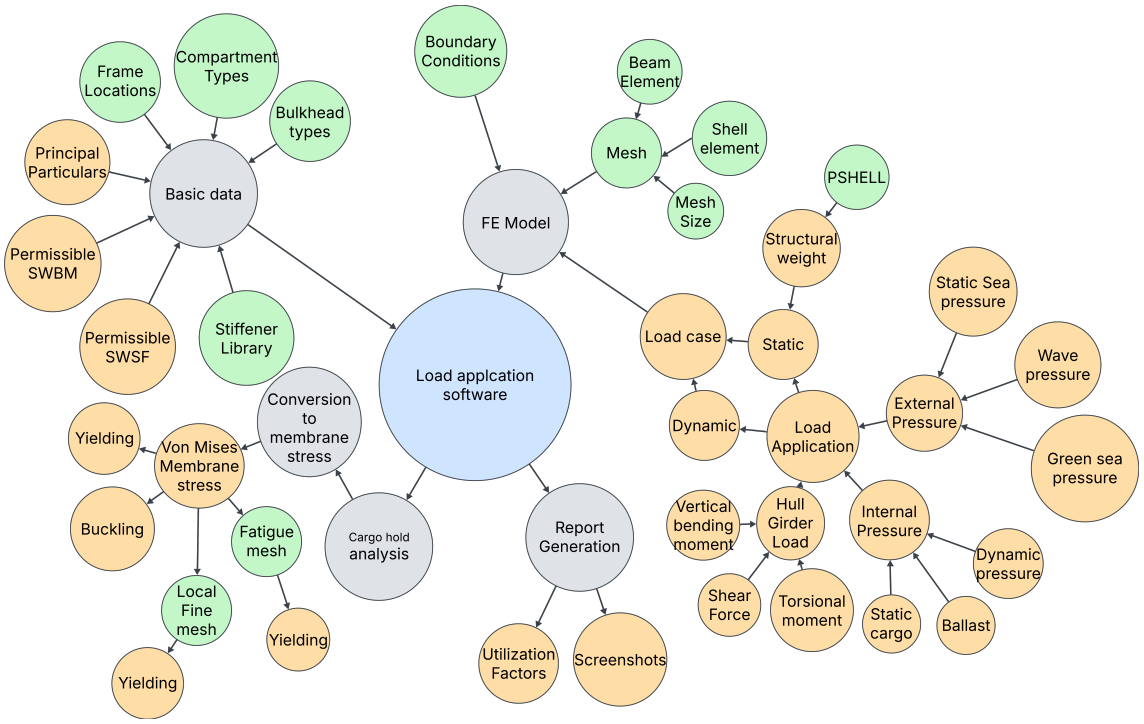


Figure 2.12: Input to output flowchart.

3
Development Procedure

This chapter outlines the overall procedure used in developing the finite element load application framework. Starting from the selection and integration of tools, through the methodology adopted for software development, and ending with the validation methodology. The procedure follows an iterative and modular design philosophy, allowing for continuous testing, verification, and expansion of the capabilities throughout the project timeline.

3.1 Tools

The tools adopted in this project center around integrating industry-standard software platforms and custom Python scripting to automate and streamline finite element analysis for ship structures. Key tools include MSC Apex for preprocessing and visualization, MSC Nastran as the FE solver, and the Design For Compliance (DFC) software.

3.1.1 MSC Apex

MSC Apex is an integrated software platform developed by Hexagon (formerly MSC Software) for computer-aided engineering (CAE), with a primary focus on finite element modeling and structural analysis [10]. It provides engineers with an environment for building and evaluating structural models, combining geometric preprocessing, meshing, analysis setup, and results interpretation into a single application. Together with this, MSC Apex comes with an integrated Python interpreter and API, which makes it possible for the user to create their tools and macros that integrate into the user interface. Finally, MSC Apex comes with the possibility of directly running the FE solver MSC Nastran from within the interface.

3.1.2 MSC Nastran

MSC Nastran is a multidisciplinary structural analysis solver currently developed by Hexagon (originally created by NASA), widely used in industries such as aerospace, automotive, and maritime engineering. It is based on the finite element method and is capable of performing linear and nonlinear static, dynamic, and thermal analyses. Within the context of this project, MSC Nastran is used as the primary solver for running the final linear structural analysis. The software reads models and analysis data from Bulk Data Files (BDF), which include elements, materials,

loads, and boundary conditions defined by cards. The cards of a Nastran BDF define all the entities and attributes required to describe a complete finite element model. Each card corresponds to a specific type of data entry, such as geometry, material properties, loads, or boundary conditions [8]. For example:

- GRID — Defines the coordinates and identification numbers of nodes (grid points) in the finite element mesh. These are the reference points used to construct elements and apply loads or constraints.
- CQUAD4 — Represents four-node quadrilateral shell elements. This card specifies the node connectivity, material ID, element thickness, and orientation.
- PLOAD4 — Applies surface pressure loads to quadrilateral or triangular elements. The load can be defined by magnitude, direction, and distribution over the element face.
- MAT1 — Defines linear isotropic material properties, including Young's modulus, Poisson's ratio, and material density.
- PSHELL — Specifies shell section properties and links elements like CQUAD4 to the appropriate material and thickness definitions.
- SPC/SPC1 — Used to apply single-point constraints (boundary conditions), restricting degrees of freedom at specified grid points.
- FORCE, MOMENT, LOAD — Used to define concentrated loads and organize them into load sets. FORCE and MOMENT specify individual loads, while LOAD groups them into specific load cases.

Together, these cards form the structured input necessary for Nastran to perform an analysis, with strict formatting rules governing field placement to ensure proper parsing by the solver.

3.1.3 Design For Compliance (DFC)

Design For Compliance (DFC) is a software tool developed by Accentus Aero, designed to streamline structural analysis and automate labour-intensive tasks such as report generation within the aerospace industry. The tool integrates seamlessly with Hexagon software, including MSC Apex and MSC Nastran, and applies stress methods based on established engineering techniques. DFC utilizes Microsoft Excel sheets for much of the input and output values in its calculations. Key features of DFC include:

- Automated Data Extraction**:** DFC is integrated into MSC Apex and provides the capability to automatically extract required data from the CAE environment. This includes node IDs, boundary conditions, loads, materials, and other essential model attributes.
- Automated Report Creation**:** The tool can automatically generate documents that summarize the results of the structural analysis, complete with images and annotations to support detailed explanations.
- Custom MSC Apex Tools: DFC includes custom-built tools within MSC Apex to further support automation and enhance user productivity. For example, a tool is included that can automatically create geometry for orphan meshes.

Since DFC essentially serves the same functional purpose as this project's objective, albeit originally developed for the aerospace industry, it naturally represents an ideal platform for adaptation. A significant aspect of the methodology thus involves ensuring that the existing DFC software architecture can be smoothly extended and adapted into a maritime-specific variant, termed "DFC for Maritime". This adaptation leverages DFC's established capabilities to streamline structural analysis, thereby facilitating compliance with maritime structural regulations.

3.1.4 CSR Finite Element (CSR FE)

CSR FE is a software created as a joint venture between the American Bureau of Shipping (ABS) and Lloyd's Register (LR) in order to provide a way for designers to assess compliance to the CSR regulation [12]. The program is based on MSC Patran, a software created by Hexagon and can be seen as a predecessor to MSC Apex. As CSR FE was created specifically to ensure compliance with CSR, it was an appropriate source of inspiration for the project and its output could be used to verify results during the software development.

3.1.5 Raytracing

Ray tracing [16] is a versatile set of algorithms for simulating how rays (representing light, sound, particles, etc.) travel through and interact with a geometric scene. Its core idea is to launch rays from a source , detect where they hit objects, and then accumulate those interactions to compute visibility, shading, collision events, or other physical quantities.

3.1.6 Application programming interface

An application programming interface (API) [17] serves as the bridge between the core functionality and any client that needs to interact with it, providing a clear and consistent contract for exchanging data and invoking operations. The MSC Apex API was used to for example insert calculated pressure loads, forces, moments and constraints from the developed software into MSC Apex.

3.1.7 Python

The language chosen for the software development was Python [18], due to its relative ease of use and good fit with the task. Python's modularity allows for easy implementation of helpful modules, which greatly expedites the process and enables more advanced operations.
The following key Python libraries were integrated to support various aspects of the workflow:

- PyNastran: Provides parsers and writers for NASTRAN finite element models, enabling direct manipulation of FE data [19].
- Pandas: Offers high-performance data structures and analysis tools for handling tabular data [20].

- SciPy: Supplies fundamental algorithms and utilities for scientific computing, including optimization, integration, and statistical routines [21].
- Trimesh: Facilitates 3D mesh processing, offering functions for loading, analyzing, and repairing geometric models [22].
- PyEmbree: Intel's high-performance ray tracing kernels, used for accelerated geometric intersection tests and rendering tasks, wrapped for Python use [23].

3.1.8 Mathcad

Mathcad is an engineering calculation software used for documenting, analyzing, and solving mathematical problems [24]. It combines standard mathematical notation with a visual interface, allowing users to create readable and interactive technical worksheets. The software is particularly well-suited for structured engineering workflows, enabling symbolic and numerical calculations, unit management, and embedded documentation within a single environment.

3.2 Methodology

The overarching methodology adopted in this thesis follows the Spiral Product Development Process (see figure: 3.1) as described by Ulrich and Eppinger (2015) [15]. This approach was selected due to its compatibility with iterative and exploratory development.

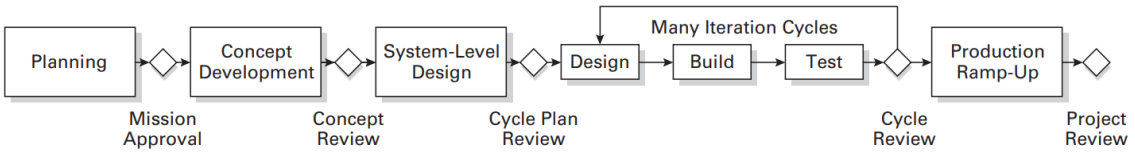


Figure 3.1: The Spiral Product Development Process [15].

3.2.1 Planning and Concept Development

A preliminary study was conducted in order to gain a better understanding of the current available software used for finite element analysis within shipbuilding. Understanding how these software are used and in what part of the process was crucial to enable the development of a functionality specification. Another important factor to consider was how to ensure compatibility with the DFC software in order to ensure a smooth integration. In addition to literary studies, there were also observational studies of the current workflow, along with a stakeholder analysis. The goal of the preliminary study was to develop a flowchart of the intended flow of user inputs to the final finite element solver result, which serves as the concept. In the list below are the methods used during the preliminary study to reach the result.

- Functionality Specification – To guide the development of the software tool, a structured functionality specification process was employed. Drawing from

Ulrich and Eppinger's methodology on translating customer needs into engineering specifications, functional, non-functional, usability, and scalability requirements were systematically formulated. These were informed by industry standards, stakeholder input, and analysis of existing solutions. Each requirement was classified as either essential ("Need") or beneficial ("Want"), ensuring a clear prioritization for implementation. This structured specification supported traceability from initial needs to final implementation.

- Observational Study – An observational study was conducted to understand the current FEA workflow used in CSR compliance at FKAB Marine Design. This involved direct observation of engineers interacting with existing tools (e.g., MSC Apex and CSR-FE) and mapping the sequence of operations from geometry preparation to load application and analysis. The goal was to identify critical user interactions, tool limitations, and integration points. Insights from this study were used to define user-centered requirements and to align the software development process with actual industrial practices.

- Stakeholder analysis - In order to get a better understanding of how the different stakeholders influence a project and to guide the exploration of required functionality, a stakeholder analysis was conducted. This is especially useful as the project's stakeholders have varying roles: customers, competitors, and so on.

- Prototype - To aid the understanding of user inputs, a Mathcad model was created as an intermediary step. The Mathcad model was used to manually calculate internal and external loads for one specific chosen load case in order to better understand and compile the CSR regulations down to a series of formulas.

3.2.2 Spiral Product Development Process

After the initial planning and concept development, the spiral development model emphasizes a cyclical flow through three main phases:

1. Design – Developing system structure, defining software modules, and interpreting CSR logic into functional workflows.
2. Build – Implementing the modules using Python and integrating them into MSC Apex using its API.
3. Test – Validating functionality and correctness by comparing against CSR-FE results and visual outputs in MSC Apex.

Each cycle is followed by intermediate reviews, allowing incremental improvements and early error detection. This methodology was critical in managing the complexity of translating CSR regulations into working code while maintaining flexibility to revise assumptions or correct implementation errors.

While the Spiral Product Development Process was originally formulated for physical product development, it was adapted for use in this software context. Specifically, the final stage of the model, Production Ramp-Up, was not included, as the result of this thesis is not a physical product. Instead, the software equivalent of production ramp-up was achieved through final code integration, internal validation using CSR-FE benchmarks, and alignment with the DFC interface environment. This adaptation ensures the spiral model remains relevant while reflecting the iterative software development.

3.3 Validation of the Method

This subchapter describes the application of the Validation Square framework, as introduced by Pedersen et al. (2000) [9], for systematic validation of design methods. The Validation Square combines theoretical and empirical perspectives across structural and performance dimensions to build confidence in a method's validity.

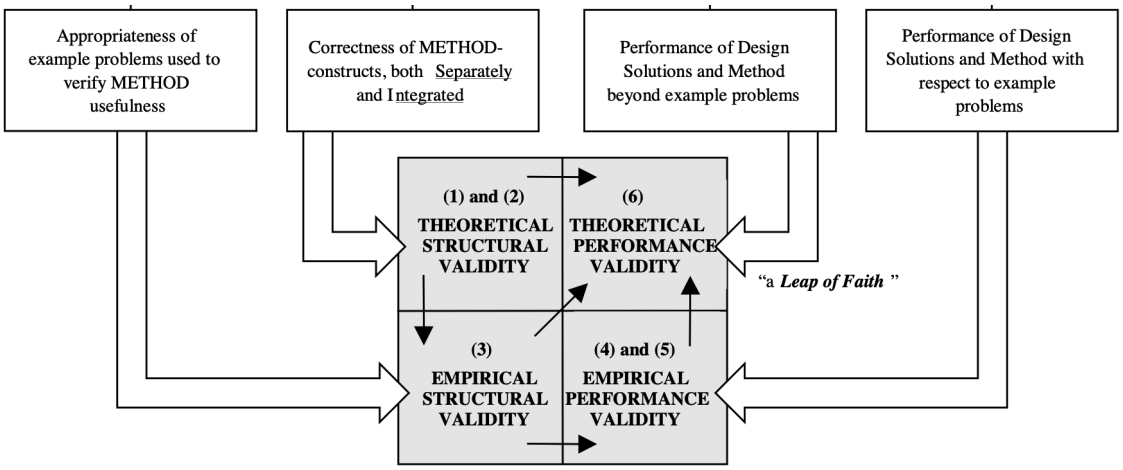


Figure 3.2: The Validation Square [9].

3.3.1 Framework Overview

The Validation square is arranged as a two by two matrix, whose vertical axes describe empirical and theoretical validity, and horizontal axes describe internal consistency and external relevance. Together, the axes create four validity modes:

- Theoretical Structural Validity (TSV)
- Theoretical Performance Validity (TPV)
- Empirical Structural Validity (ESV)
- Empirical Performance Validity (EPV)

3.3.1.1 Theoretical Structural Validity

Demonstration of logical consistency among core method constructs, typically via literature review and formal analysis. For example, the absence of redundant assumptions and alignment with established theory.

3.3.1.2 Theoretical Performance Validity

Justification that the chosen case studies or example problems are structurally appropriate to exercise all facets of the method. This often involves inductive reasoning to show that examples map cleanly onto the method's constructs.

3.3.1.3 Empirical Structural Validity

Evaluation of the method's outcomes on the example problems, demonstrating feasibility and usefulness (reduced cost, improved performance). Case results must exhibit that each construct contributes to the intended purpose.

3.3.1.4 Empirical Performance Validity

Inductive inference that the method will perform usefully beyond the specific examples. This relies on linking prior evidence of structure and performance to broader contexts.

4

Result of the Preliminary study

This chapter presents the findings from the preliminary investigation into CSR-compliant hull structure analysis. It begins by surveying the existing classification society software, then moves on to an observational study of current workflows, stakeholder needs, and early prototyping efforts. Together, these results frame the requirements and opportunities that will drive the design of a more flexible, designer-centric analysis environment.

4.1 Researching Existing Solutions

Table 4.1 lists the member societies of IACS along with their proprietary software tools for hull structure analysis. These tools facilitate compliance checks based on the respective society's ruleset and, where applicable, IACS CSR. Some societies provide separate software for CSR compliance, while others incorporate CSR modules within their main software. It is also important to note that the Russian Maritime Register of Shipping was excluded from IACS in 2022, though it remains included in the table for completeness.

Table 4.1: Classification societies and their associated software tools

Classification Society	**Software (Ruleset)**
Det Norske Veritas (DNV)	Nauticus Hull (DNV Rules & CSR)
American Bureau of Shipping (ABS)	Containership Structural Assessment (ABS Rules) Eagle CSR FEA (CSR)
Lloyd's Register (LR)	ShipRight (LR Rules) Common Structural Rules Software (CSR)
Bureau Veritas (BV)	Veristar HULL (BV Rules & CSR)
Nippon Kaiji Kyokai (ClassNK)	PrimeShip-HULL (ClassNK Rules) PrimeShip-HULL/HCSR (CSR)
Registro Italiano Navale (RINA)	Leonardo Hull (RINA Rules & CSR)
Korean Register (KR)	SeaTrust - HullScan (CSR)
China Classification Society (CCS)	COMPASS HCSR (CSR)
Indian Register of Shipping (IRS)	IR-CSR (CSR) NovaHull (IRS Rules)
Russian Maritime Register of Shipping (RS)	RS Hull Calculation System (CSR)
Polish Register of Shipping (PRS)	Not found
Croatian Register of Shipping (CRS)	CRS Software tool (Mentioned on site, no details found)
Turk Loydu (TL)	TL Hull Scantling (Likely TL Rules)

4.2 Observational Study

To ground the following analysis in real-world practice, an observational study was conducted at FKAB Marine Design to document the end-to-end FEA workflow from MSC Apex preprocessing and CSR-FE global analysis through local and fatigue assessments, and to identify designer-focused limitations.

4.2.1 Current FEA workflow at FKAB Marine Design

Figure 4.1 illustrates the current workflow used to go from a finite element model to finalized FEA results. This specific graphic represents the process when CSR-FE is used as the third-party compliance load application software. The color coding of the steps corresponds to the software used in each stage.

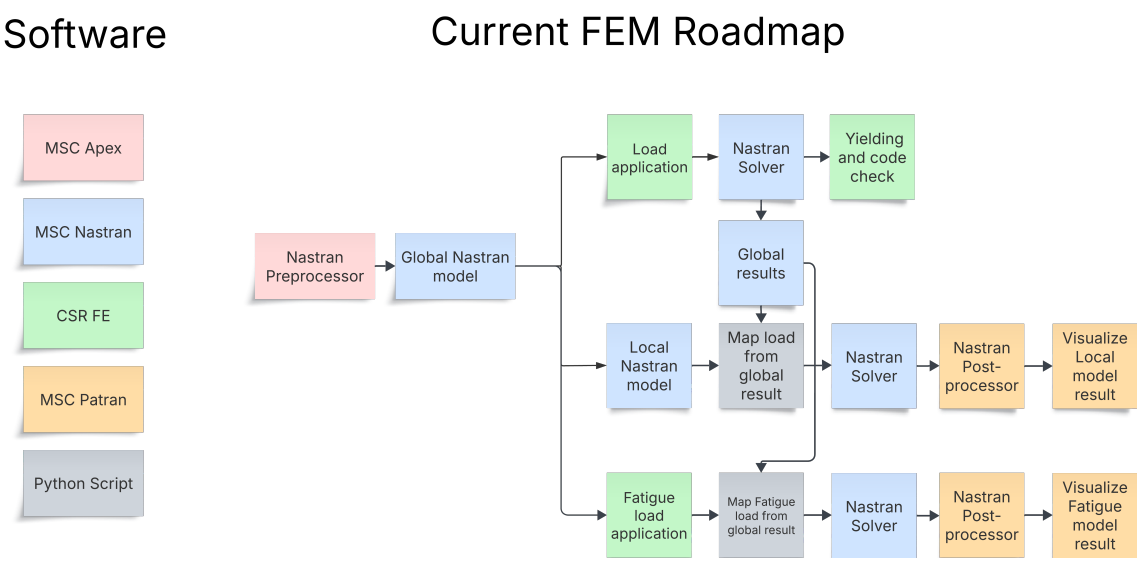


Figure 4.1: The current FEM workflow at FKAB Marine Design.

4.2.1.1 Nastran Preprocessor Stage

The process begins with creating a global FE model using preprocessing software—MSC Apex in this case. This involves importing or generating geometry, meshing, and applying boundary conditions such as loads and constraints.

4.2.1.2 Global Analysis

Loads are defined and applied via CSR-FE's load application functionality. The MSC Nastran solver is then used to compute the global results. CSR-FE performs yielding and code checks to assess compliance with the Common Structural Rules (CSR).

4.2.1.3 Local Analysis

Load data is extracted from the global results and mapped to a local FE model for detailed analysis using a custom Python script. The local model is simulated using MSC Nastran, and the results are post-processed and visualized in MSC Patran.

4.2.1.4 Fatigue Analysis

In parallel, a fatigue-specific load application is created, often requiring a finer mesh. Loads are mapped from the global results, and the fatigue behavior is simulated using the Nastran solver. Results are post-processed and visualized to evaluate fatigue performance over the structure's lifetime. The final results are compiled into reports delivered to the client and the classification society. However, these reports are not used for full compliance verification, as classification societies perform their own simulations to confirm the results.

4.2.2 Highlighted issues

The observational study and examination highlighted interesting issues with current CSR-compliant software regarding flexibility. In general, the software is designed in order to test for compliance to standardized loadcases and scenarios, but outside of that, the softwares tends to be quite limited. This is exemplified by the limited and/or tedious procedure of changing calculation parameters or loadcases by the designer when using the software. This rigidity makes the process slow and cumbersome, hence, a more "designed by designers" mindset is sought after, where in addition to allowing for more flexibility, more detailed compliance reports would greatly increase usability. Current reports are said to be rather binary, either in compliance or not. More detailed reports with mathematically based example changes or just more clarity in where and why the model is not in compliance are useful.

4.2.3 Geometry vs Mesh

During the observational study, an interesting distinction regarding the methodology of load application was highlighted within the MSC Apex software. The Nastran Bulk Data File (BDF) is standard in the industry, and also happens to be used by a wide variety of software to store FE mesh and loadcases in. This makes.BDF files are very flexible and allow the designer to utilize whatever software they feel comfortable with. The distinction arises with the choice of a geometry-based FE preprocessor. MSC Apex has the capability for either a so-called "orphan mesh", where the mesh is by itself, or it can be connected to geometry.

4.3 Functionality Specification

Based on the conducted research on current FEA solutions, the classification society software and the stakeholder interests, a list of required functionality could be made. The list is split into functional requirements, non-functional requirements, usability requirements, and scalability requirements. Furthermore, each requirement is labeled either a need or a want, with a “Need” denoting a requirement that has priority and is expected to be part of the final product. The “Want” label indicates a requirement that is expected to significantly increase the value of the final product, but whose implementation is not critical. The full compilation of requirements can be found below in table 4.2.

Figure 4.2: Functionality specification

Req. ID	Requirement Description	Type
FR1.1	The solution must integrate seamlessly with MSC Apex's Python API, allowing direct import/export of geometry	Need
FR1.2	The solution must enable custom tools/plugins within the MSC Apex GUI for ship-specific analysis workflows.	Need
FR2.1	The tool must implement IACS CSR (Bulk Carriers & Oil Tankers) including geometric checks (e.g., slenderness	Need
FR2.2	Provide standardized procedures for applying loads aligned with CSR guidelines.	Need
FR2.3	Support verification of compliance (e.g., local element checks, plate thickness, stiffener spacing) per CSR.	Need
FR3.1	Leverage DFC code structure and methods, adapting them for shipbuilding standards.	Need
FR5.3	Handle multiple load cases within one workflow.	Need

Req. ID	Requirement Description	Type
NFR1.1	The software should handle ship structures efficiently, aiming for reasonable analysis times.	Need
NFR1.2	Minimize memory usage and overhead to accommodate large FEA models.	Want
NFR2.1	Accuracy must be within acceptable tolerances of classification society reference tools.	Need
NFR3.1	Use a modular architecture for ease of updates and addition of new classification rules.	Need
NFR3.2	Follow clear code conventions and version control for maintainability.	Need
NFR4.1	Safeguard proprietary data (material libraries, geometry) to prevent unauthorized access.	Want
NFR4.2	Restrict or control code modifications to preserve result validity (avoid tampering).	Need

Req. ID	Requirement Description	Type
UI1.1	New functionalities must appear as additional menu options or tool panels in MSC Apex	Need
UI2.1	Provide step-by-step guidance for classification compliance processes.	Want
UI2.2	Offer quick links to relevant documentation (CSR guidelines, geometry constraints) within the tool.	Want
UI3.1	Display descriptive error messages if input data (geometry, materials, loads) is missing or invalid.	Want
UI3.2	Alert users to potential rule violations (e.g., out-of-range stiffener dimensions) early in the workflow.	Want

Req. ID	Requirement Description	Type
SF1.1	The core design should allow relatively easy addition of other classification rulesets (e.g., DNV, ABS).	Need
SF1.2	Use a modular code structure to add new rule modules without heavy refactoring.	Need
SF2.1	Accommodate advanced analyses (e.g., permissible still water bending moment) for further differentiation.	Want
SF3.1	While focused on shipbuilding, the solution could be generalized for offshore or similarly regulated sectors.	Want

4.4 Mathcad Prototype

The Mathcad prototype served as the first practical implementation of local load calculations, providing a transparent and flexible environment for verifying key numerical methods before full-scale Python implementation. Initial steps involved translating CSR equations for section modulus, bending moments, and shear forces into live Mathcad worksheets. Each worksheet was structured to accept geometric inputs: plate thickness, section dimensions, and load positions, then compute stresses under specified loading scenarios. An example from the Mathcad prototype

can be seen in figure 4.3, demonstrating the formula for calculating the acceleration at any position in the FE model.

Acceleration at any position:

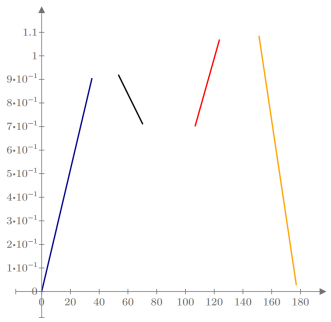
$$a_x := -C_{XG} \cdot g \cdot \sin\left(\varphi \cdot \frac{3.141}{180}\right) + C_{XS} \cdot a_{surge} + C_{XP} \cdot a_{pitch} \cdot (z - R)$$

$$a_y := C_{YG} \cdot g \cdot \sin(\theta) + C_{YS} \cdot a_{sway} - C_{YR} \cdot a_{roll} \cdot (z - R)$$

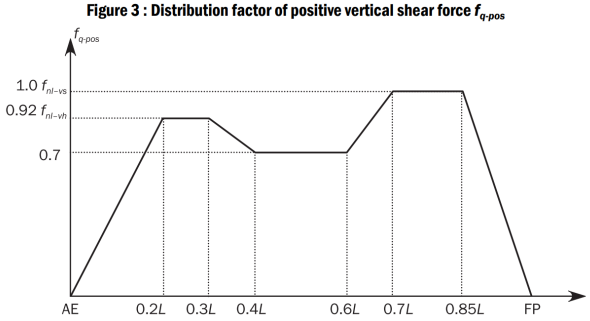
$$a_z := C_{ZH} \cdot a_{heave} + C_{ZR} \cdot a_{roll} \cdot y - C_{ZP} \cdot a_{pitch} \cdot (x - 0.45 \cdot L)$$

Figure 4.3: Example calculation of acceleration in the x,y and z direction at any position.

The Mathcad prototype made it apparent that many of the correction factors used in for example the calculation of yaw, pitch, and rotational accelerations, along with hull girder loads were only specified for a select points along the ship's longitudinal. The rest needed linear interpolation to calculate the intermediate values. An example of the distribution factor of positive vertical shear force (see figure: 4.4 (b)) interpolation using Mathcad is shown in figure 4.4 (a), where the colored lines are the interpolated intermediate values.



(a) Intermediate value graph for the distribution factor of positive vertical shear force.



(b) Graph showing the specified values for the distribution factor of positive vertical shear force.[2]

Figure 4.4: Interpolated and specified values.

4.5 Flowcharts

Finally, as a result of the preliminary study, a plan for the software implementation could be made. This plan can be described using a flowchart, which can be found below in figure 4.5

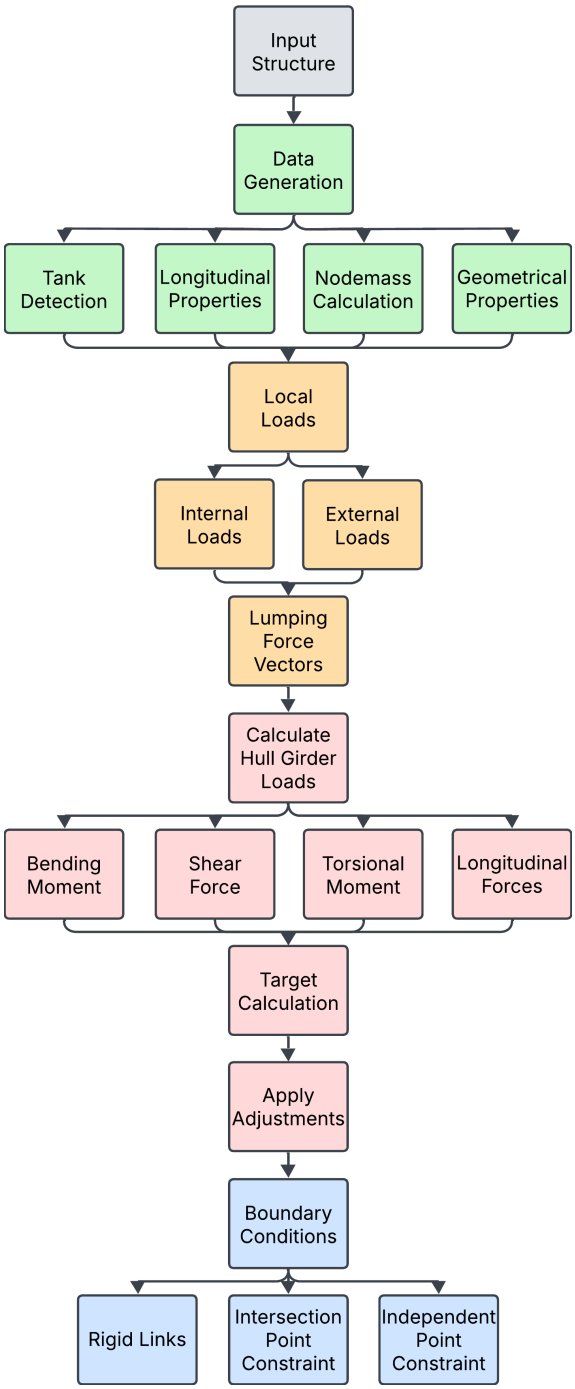


Figure 4.5: Overview of the software implementation plan.

5
Software Development

This chapter presents the methodology and process followed in the software implementation of the Common Structural Rules [2] within the scope of the thesis. The chapter details the iterative development approach, testing and validation strategies, and the full integration of the CSR-based calculation logic into MSC Apex through the use of its API and supporting Python libraries.

5.1 Iterative approach

In order to not have to tackle input structure, CSR calculations, and MSC Apex API structure all at once, an iterative approach was used to successively implement more of the final code. As a first iteration, the user inputs were all hardcoded, and the required models were all manually exported to do everything outside of MSC Apex. This involved a greater emphasis on utilizing pyNastran to read and write BDF files to end up with a final file including all loads and boundary conditions. During the second iteration most of the code was restructured in a way to allow for user friendly input and swapping out pyNastran and BDF files to utilizing the MSC Apex API.

5.2 Developing Validation tests

To ensure accuracy throughout the implementation, validation routines were developed to compare intermediate results against expectations derived from the CSR FE software. These routines leveraged pandas and NumPy to parse and compare values from output CSV and Excel files.

Load comparisons were performed both for local pressures and integrated hull girder forces at each longitudinal station. An error detection method was also introduced, which flagged any node or element with deviations beyond a defined tolerance. These outliers could then be visualized directly in MSC Apex through API driven highlighting. This mechanism greatly facilitated debugging by allowing the user to visually trace calculation errors back to specific geometric features or load anomalies.

5.3 Final Code implementation

The final implementation is modularized into an input/output interface and three primary processing modules: data generation, local load calculation, and load integration. These modules reflected the structure and logic of the CSR, ensuring compliance with the standards while maintaining flexibility for future adjustments.

5.3.1 Input & output interface

To maintain user familiarity and compatibility with existing workflows, the input and output formats were based on Microsoft Excel. Input sheets were structured to mirror those of CSR FE, with individual sheets for principal particulars, web frame definitions, load cases and more. Data exchange with Excel was handled using Pandas, providing efficient reading and writing capabilities as well as data consistency checks.

5.3.2 Data generation

The data generation module was the first module to be developed, as it was the module which reads all user input data and calculates all intermediary results for use in subsequent calculations. This approach was intended to yield the possibility for the user to change the calculated intermediary parameters before going into load calculation.

5.3.2.1 Longitudinal Stations

For many of the later calculations, the x location of the model's longitudinal stations was required. The frame locations were derived from user input, where the frame numbers and frame spacings were specified. These inputs made it possible to calculate the x coordinate of all frames, and by extension, also the location of the web frames and bulkheads. This was since the user inputs specify the location of the web frames and bulkheads as existing on a specific frame.

5.3.2.2 Interpolation of Factors

In order to get intermediate values for correction factors used for most subsequent calculations, linear interpolation was used. This process involved inputting all x and y values for the values supplied in the CSR, and then interpolating them using the SciPy module interp1d to get the resulting value.

5.3.2.3 Nodemass calculation

There also needed to be calculation of the node mass, in order to include the effects on hull girder loads stemming from the strucural steel weight. This was done by extracting the PSHELL properties from the model elements using the MSC Apex API. The PSHELL contains a reference to the MAT associated card, which among other properties, describes the density of the applied material. The density could

then be multiplied by the element surface area and thickness to get the total mass of every element.

5.3.2.4 Crosssection properties

To accurately calculate and apply the forces and moments to the midship model as well as to define the boundary conditions, it was necessary, as a preliminary step, to identify the relevant nodes for these operations. Following the CSR framework, the locations where loads and boundary conditions are typically applied correspond to the longitudinal positions of the web frames. These web frame positions were obtained from user input, and a methodical search was conducted using the MSC Apex API's ProximitySearch functionality to determine the associated nodes.

The ProximitySearch tool was employed to define thin transverse search volumes at each specified web frame x-coordinate. These volumetric boxes were configured to span the full width and height (y and z directions) of the model, while a finite tolerance was introduced in the x-direction to accommodate potential mesh irregularities. Within each of these bounding volumes, ProximitySearch identified all intersecting elements, from which the corresponding node IDs were extracted. This node list then served as the basis for further analysis at each cross-section.

Subsequent processing of these nodes involved assigning them to their respective elements and evaluating their contribution to key sectional properties, including net sectional area, moments of inertia, and neutral axis location. In alignment with CSR requirements, only longitudinal elements were considered in this analysis, as they are the structural members contributing to hull girder stiffness in the longitudinal direction. To ensure compliance, a filtering check was implemented to exclude any elements whose surface normals had an x-component exceeding a defined tolerance threshold. This meant that the elements of the hull, girders, stringers, deck, and more were included (since they are longitudinal), while transversal structural members such as the webframes were excluded.

To compute the net sectional area, the lengths of the element edges were first determined by calculating the distance between the y and z coordinates of each element's end nodes using the Pythagorean theorem. These lengths were then combined with the corresponding material thickness values retrieved via the MSC Apex API. By multiplying the edge length by the assigned thickness, the net sectional area contribution of each individual element could be accurately calculated. Also added to each element's net sectional area was the cross-sectional area of eventual stiffeners associated with the element. These could also be acquired through the MSC Apex API. An additional consideration was required at intermediate web frame stations to avoid double-counting structural contributions from elements on both sides of the section. For this reason, the calculated net sectional area for these frames was halved to accurately reflect the effective cross-sectional contribution. Following the area calculation, the centroidal position (neutral axis) of each cross-section was determined. This was done by computing the first moments of area about the y and z axes, summing the products of each element's net sectional area and its respective

y and z coordinates. These sums were then divided by the total net area of the cross-section to yield the neutral axis coordinates in the transverse plane.

With the neutral axis established, the moments of inertia about the y and z axes were calculated by summing the second moment contributions of each element. Each contribution was computed as the product of the element's net sectional area and the square of its distance from the corresponding neutral axis. This provided the transverse and vertical moments of inertia for each cross-section.

The output from these calculations was structured into several intermediary CSV files. Each cross-section was assigned its own file containing all relevant nodes and their associated properties. A final summary file was then generated, compiling the sectional properties of all web frames across the model.

5.3.2.5 Tank Detection

The implementation of internal load calculations, as prescribed by the CSR, necessitated an initial step of identifying the boundaries of each cargo tank within the midship finite element model. As the internal loads must be applied specifically to the tank boundary surfaces, it was essential to isolate these surfaces reliably and accurately. Since the initial goal was to retain the possibility of processing orphan meshes, a raytracing-based detection method was adopted to extract tank boundaries directly from the mesh (see figure 5.1).

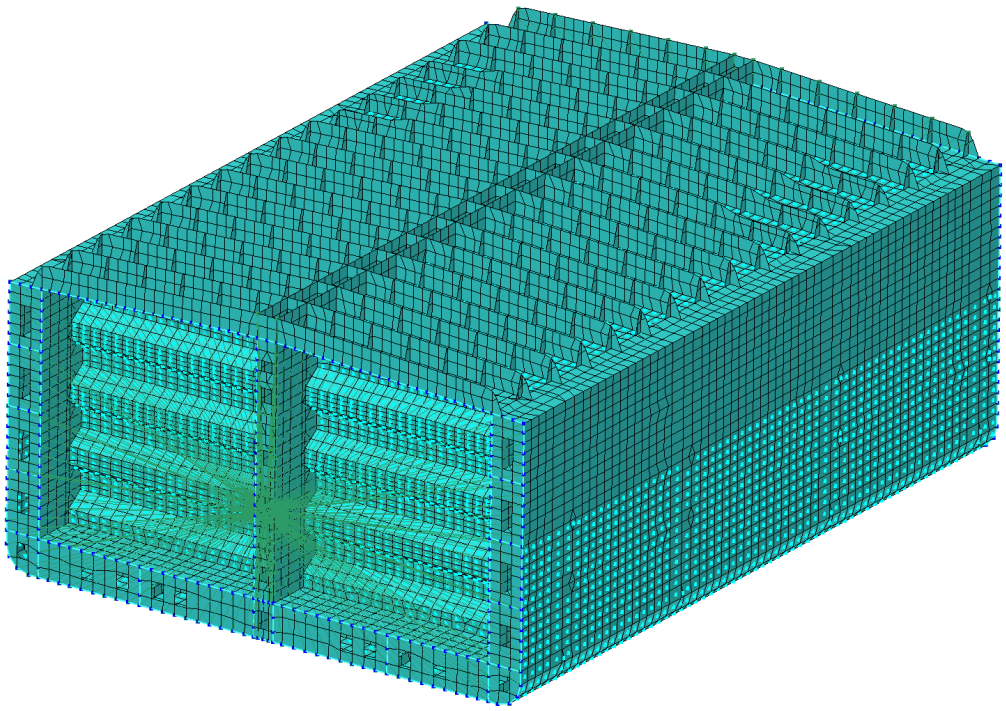


Figure 5.1: Midship FE mesh.

Using user-defined input parameters, including the moulded breadth and height of the model, as well as the longitudinal location of transverse bulkheads—an internal reference point was established for each tank. From this central seed point, the extreme x, y, and z coordinates of the tank were estimated, thereby defining an approximate spatial extent. Six secondary points were then derived between the internal reference point and each of the six bounding directions. These secondary points served as ray origins, chosen to ensure that even elements oriented nearly tangentially to the tank interior (such as those in corrugated bulkheads) could be intersected by at least one ray.

Prior to executing the raytracing, the model was converted into a triangular surface mesh using the Trimesh library. This was necessary as the raytracing engine required triangular geometry to function correctly. To ensure comprehensive surface coverage, 50,000 rays were emitted from each secondary point, uniformly distributed over a unit sphere. This dense ray distribution minimized the risk of overlooking any small or irregular surface segments. Each triangular hit element was mapped back to its corresponding element ID in the original BDF model, thereby preserving traceability to the structural data. For each of the six tanks, a dedicated BDF file was created containing only the elements detected as part of the internal boundary. A secondary pass was then performed on these files to address potential gaps in the boundary mesh. This was achieved by exploiting the relational data between the GRID and CQUAD4 entries in the BDF file, enabling the patching of missing connections where necessary.

Once clean, watertight boundary meshes were obtained (an example of which can be seen in figure 5.2, the geometric volume and center of gravity for each tank were calculated using Trimesh's built in volumetric analysis functionality. These properties were essential for computing the internal hydrostatic and hydrodynamic pressures.

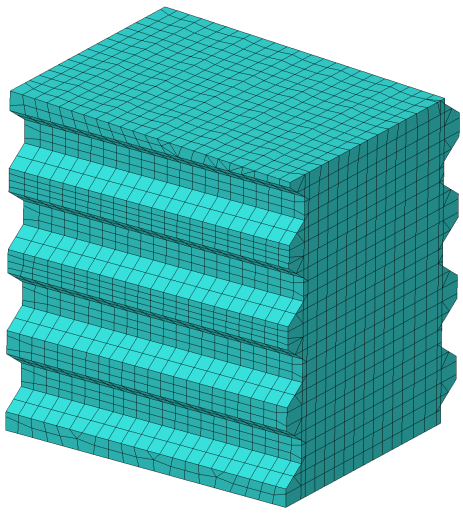


Figure 5.2: Tank boundary extracted from the midship

However, due to the high number of rays and several tanks involved, the Trimesh raytracing engine proved computationally expensive and time-consuming. To improve performance, the raytracing backend was later replaced with PyEmbree. This substitution resulted in a significant reduction in computation time.

In parallel with determining the tank volume and center of gravity, the software also required the calculation of ship motions and accelerations at each tank's center. These accelerations—expressed in the x, y, and z directions—were later used in the local pressure calculations. Furthermore, each tank required the identification of a reference point, typically located on the tank's upper surface, from which peak dynamic pressures would be evaluated. This reference node was found by extracting the main deck from the global BDF file and cross-referencing it with the previously identified tank boundary mesh. At each intersecting node, the CSR defined formula for vertical acceleration response, denoted as Vj, was evaluated, and the node with the highest resulting value was selected as the tank's reference point (see figure: 5.3.

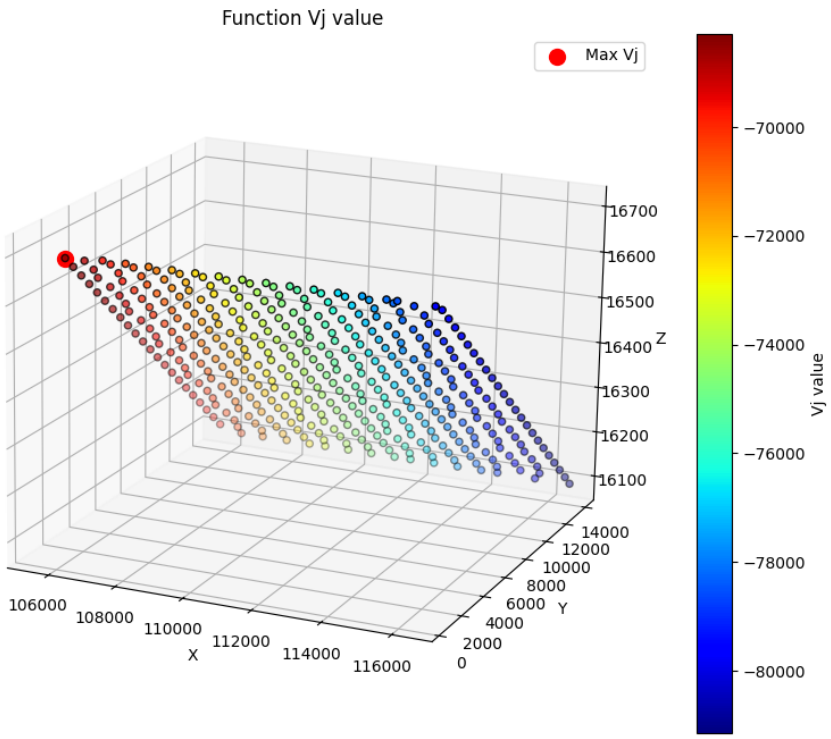


Figure 5.3: Plot showing the nodes of the top boundary and calculated value of Vj

5.3.2.6 Boundary conditions

To apply the boundary conditions to the model, the previously computed cross-sectional properties were leveraged. Per CSR, the boundary conditions at the fore and aft ends of the midship FE model are applied using a combination of rigid links, constraint beams, and a central independent point positioned along the neutral axis. For each end of the model, a rigid link is established between the relevant longitudinal nodes and the independent point, which acts as the reference for applying translational and rotational constraints depending on whether the point is located at the fore or aft end (see figure: 5.4).

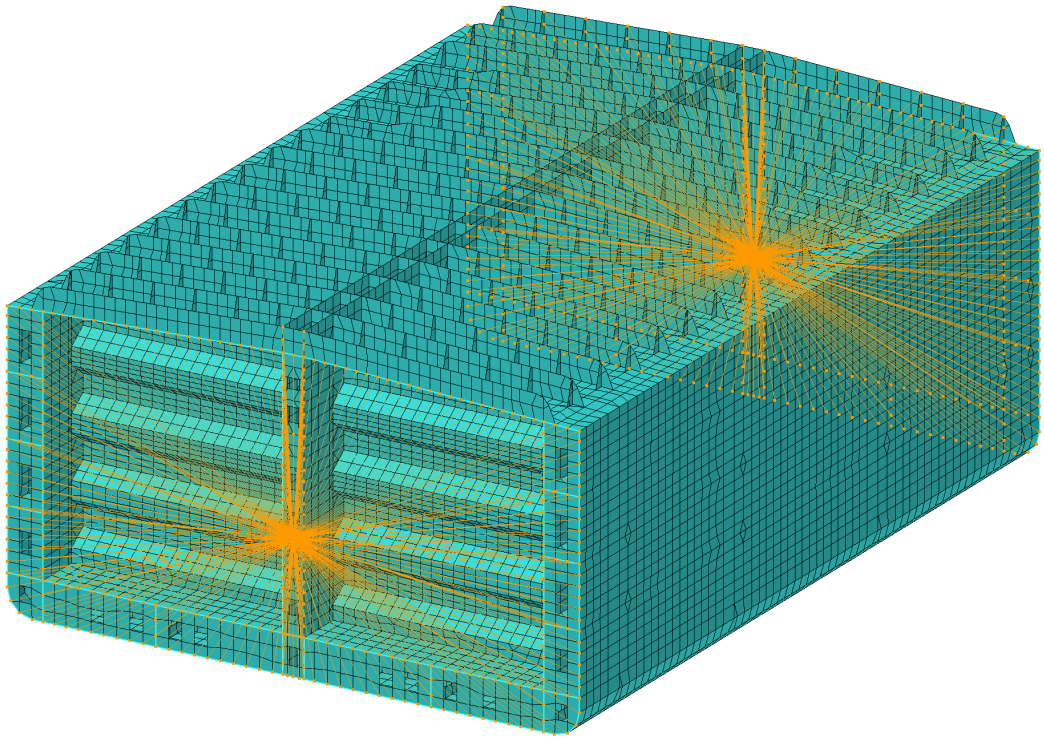


Figure 5.4: Rigid links (Highlighted in orange) automatically created in MSC Apex.

In addition to the independent point, constraints were applied at the intersection point between the centerline and inner bottom plating, which is fixed in translation to provide additional stability (see figure 5.6 and 5.5). To find the intersection point, the MSC Apex ProximitySearch functionality was once again utilized. Here it was used to find the node closest to an initial guess, at x value equal to the model ends, y equal to zero, and finally z equal to the minimum z value of the tanks. End constraint beams are also implemented along the longitudinal members at both ends of the model. These beams are assigned idealized stiffness properties as defined by CSR: one twenty-fifth of the vertical moment of inertia of the hull girder and one eightieth of the total net sectional area at the model ends. These structural idealisations ensure that the global response of the midship model reflects the hull

girder behavior intended by CSR, and that the deflection patterns and resulting stresses remain physically representative during load application. The rigid links, point constraints, and beams were applied to the midship model using the MSC Apex API.

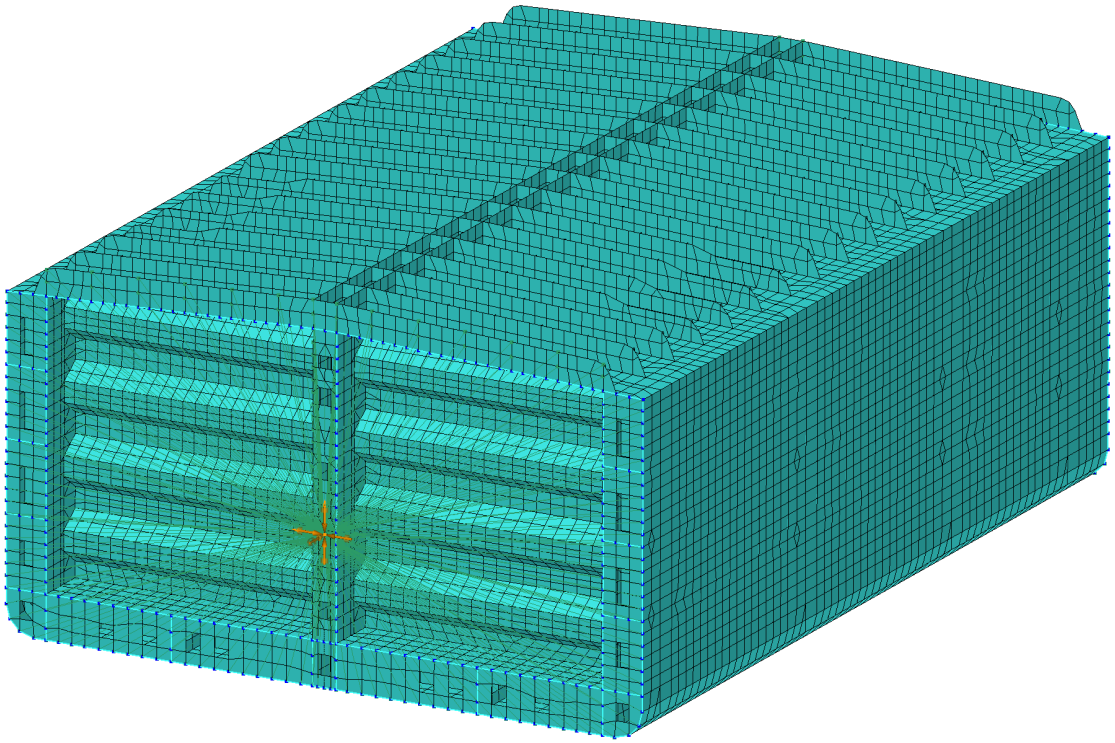


Figure 5.5: Aft point constraint automatically applied to the independent point

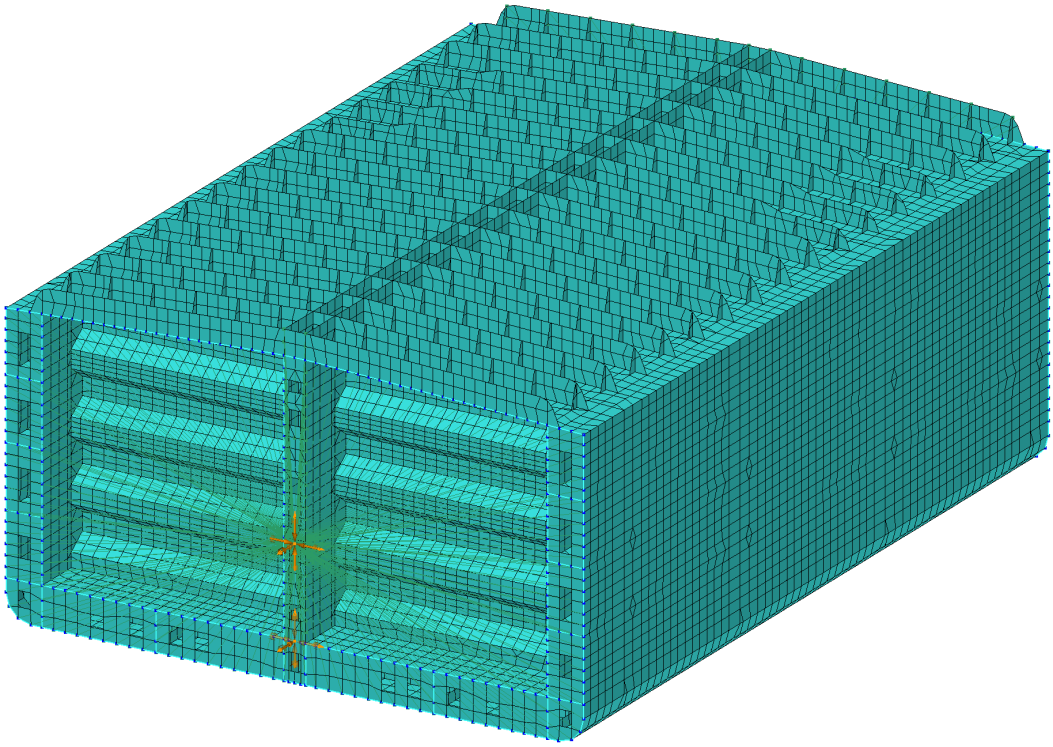


Figure 5.6: Fore point constraints automatically applied to the independent point and intersection point.

5.3.3 Moulded Breadth at the waterline

Bx, Moulded breadth at the waterline in meters at the considered cross section, is a measure that can either be input or calculated. For the project, this was implemented as an automatically detected measure. This was done by simply conducting a search among all nodes whose z coordinate aligned with the waterline coordinates (draught), and finding the largest y coordinate among those at the considered cross section. The measurement is used when calculating the external pressure that acts on the hull.

5.3.4 Local loads

With the tank centre of gravity and reference coordinates established, the internal static and hydrodynamic pressures were calculated using the acceleration data defined for each load case. This process followed the CSR formulations for internal loads, where pressure magnitudes are dependent on tank geometry, filling level, ship motions, and accelerations. The calculation was performed iteratively across all specified load cases. For each load case, the software looped through the corresponding tank elements extracted previously into dedicated BDF files and calculated the pressures at the nodal level.

Because each load case defines which tanks are filled or empty, a conditional check

was included to skip any tanks marked as empty in the loading condition. For the tanks included in a given case, nodal pressure values were calculated and stored along with their corresponding node IDs in a Python list structure. These values were subsequently inserted into MSC Apex using the API, applying the internal pressure loads directly to the relevant elements of the tank boundary as a pressure field together with the external pressure.

External pressures were handled similarly. The outer shell of the midship model was isolated and exported as a BDF file to allow for efficient reuse during each load case loop. Pressure magnitudes were then computed based on CSR equations for external hydrostatic and hydrodynamic loads, which depend on the longitudinal and vertical positions of each element, wave profile, and still water conditions.

A crucial step in the application of both internal and external pressures involved ensuring correct pressure vector orientation. External pressures must act inward, toward the hull centerline, whereas internal pressures must act outward from each respective tank center. However, since the original element normals in the model were oriented uniformly, typically in the positive x-direction, this led to incorrect pressure directions during initial implementation. To resolve this, a vector was constructed from the centroid of each element to the relevant tank centre of gravity (for internal loads) or the hull centre (for external loads). The true element normal was extracted using the MSC Apex API, and the dot product between the two vectors was calculated. If the dot product was negative, indicating the vectors were opposed, the pressure direction was inverted to ensure correct physical orientation. This adjustment ensured that all pressures, both internal and external, were applied in the correct direction relative to the geometry of the model, consistent with CSR requirements. The automatically applied pressure field inside MSC Apex for loadcase 1 can be seen in figure 5.7 below. Note how five of the tanks are colored (indicating pressure) while one is yellow (indicating no pressure), this is because the specific tank was noted as empty in the associated load pattern for load case 1.

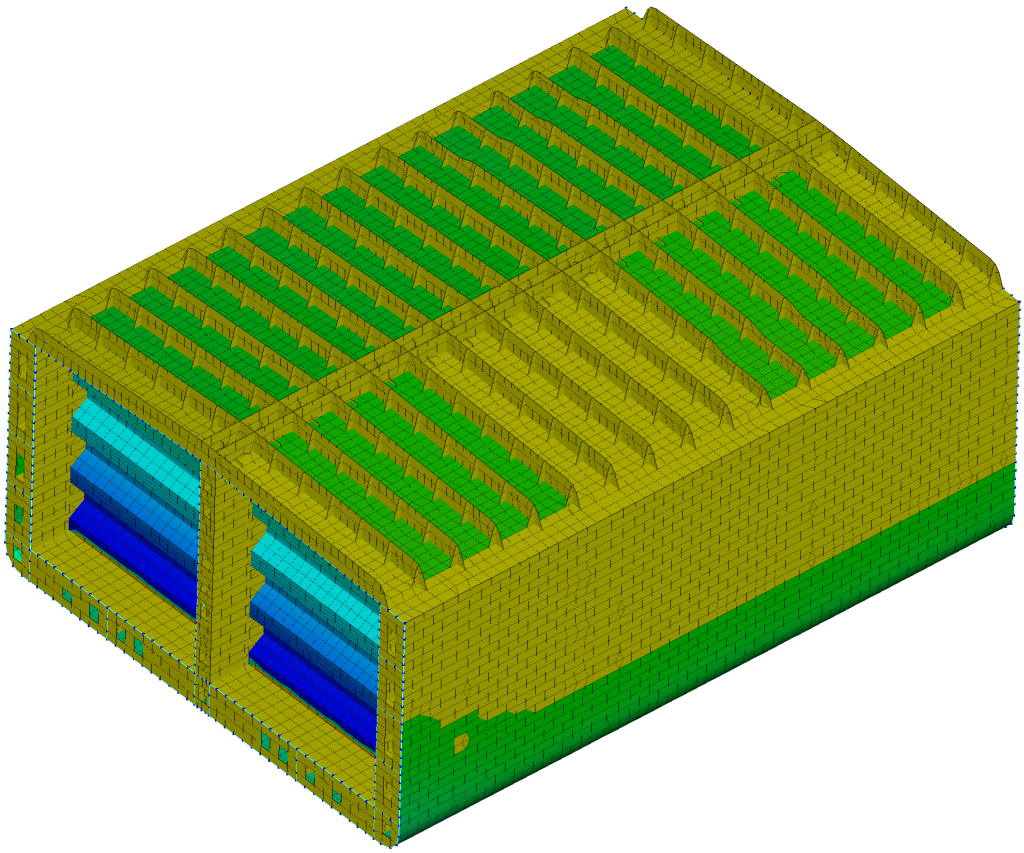


Figure 5.7: Pressure field for load case 1. Yellow elements indicate no pressure applied.

5.3.5 Load integration

The first step in the load integration process was to calculate the resultant shear forces, bending moments, and torsional moments generated by the applied local loads. Following the CSR procedure, these loads were vectorized and subsequently "lumped" to discrete longitudinal stations along the ship's length. These stations corresponded to the x-coordinates of the web frames in the midship model, as defined by user input.

For each load component, the force vector originating at an element's centroid was translated to the nearest web frame location along the x-axis. This transformation preserved the directional characteristics of the force while allowing the aggregation of loads at defined sectional locations. In addition to the forces stemming from the local loads, the forces from the earlier node mass calculation were also included in the vertical forces by multiplying each mass vector by the gravitational constant. The result of this operation was a structured set of lumped forces and moments for each station, comprising longitudinal, transverse, and vertical force components as well as torsional moments about the longitudinal axis.

Once all relevant loads were lumped, the distribution was used as input to the standard elastic line formulation. In this approach, the midship model was idealized as a simply supported beam, with supports positioned at the aft and fore ends. This simplification allowed the use of classic beam theory to calculate vertical reaction forces and derive shear force and bending moment distributions along the model's longitudinal axis. The shear force and bending moment distribution along the ship's length for loadcase 1 can be seen below in figure 5.8 and 5.9. As can be seen, loadcase one is a "sagging" scenario, a "hogging" scenario can be seen in figure 5.10.

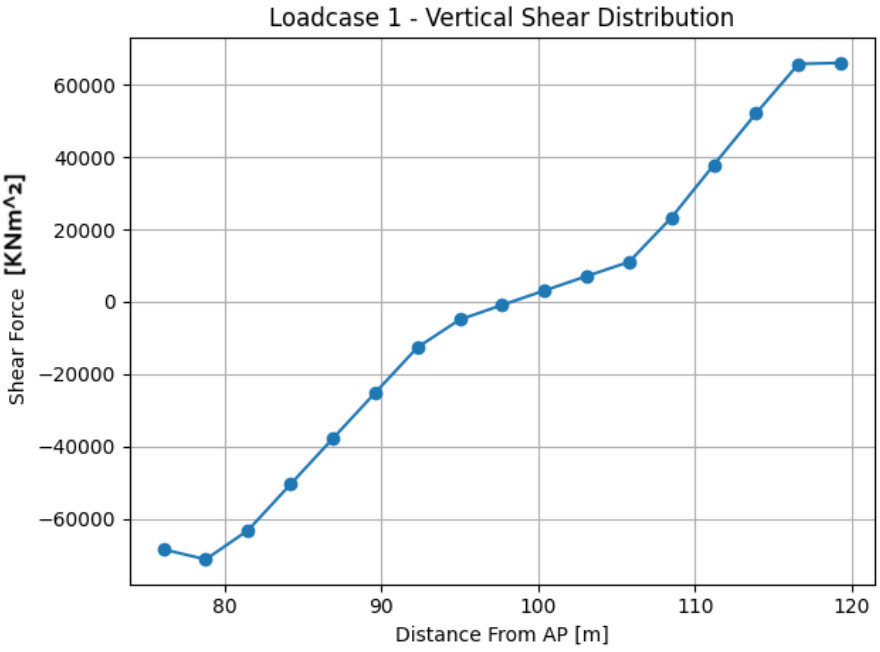


Figure 5.8: Shear force distribution for loadcase 1

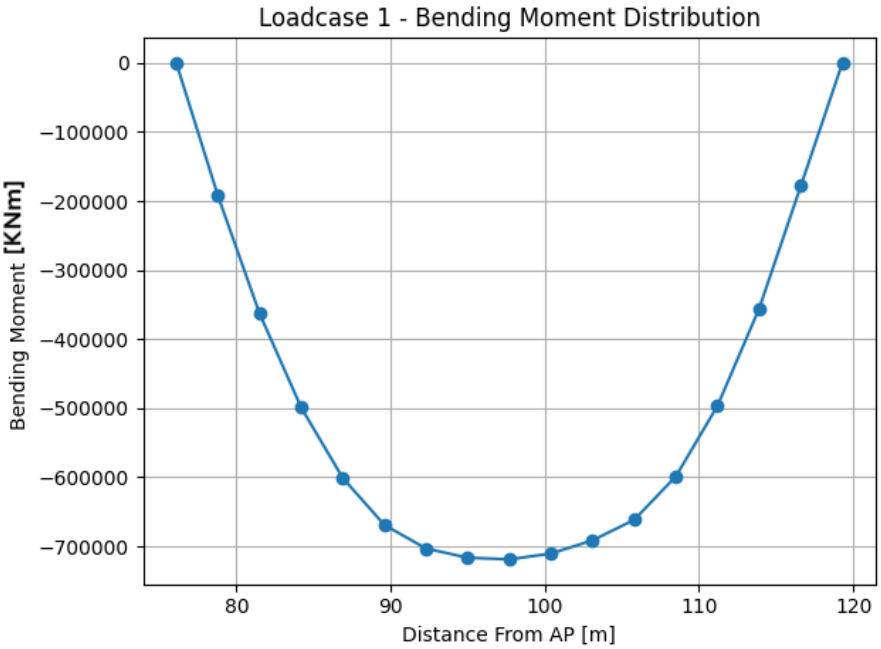


Figure 5.9: Vertical Bending moment distribution for loadcase 1

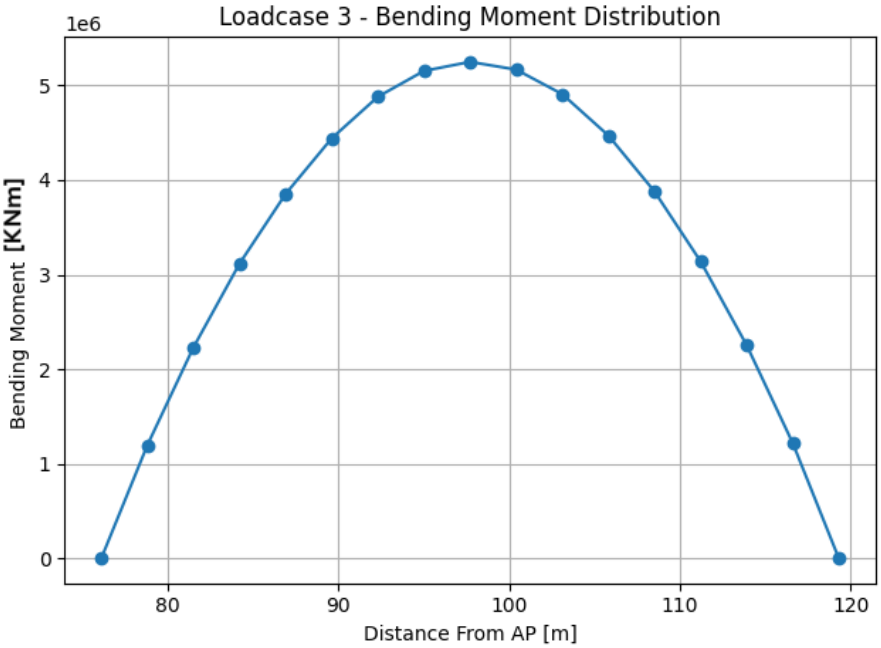


Figure 5.10: Vertical Bending moment distribution for load case 3

5.3.5.1 Target Calculation

To assess whether the midship model meets the structural adequacy criteria of the Common Structural Rules, is is necessary to calculate the target values for the vertical and horizontal shear forces, vertical and horizontal bending moments, and, where applicable, the torsional moment at each longitudinal station along the model. These targets serve as the benchmark that the loads should match, often requiring adjustment. As an example, the target vertical bending moment was calculated at each longitudinal station and load case by summing the still water bending moment and the wave-induced vertical bending moment.

5.3.5.2 Adjusting Hull Girder Loads

The first step of adjusting the integrated hull girder loads to the target values was to adjust the shear force. This adjustment was conducted with the condition that the shear force targets were exceeded at the target location. The shear force can be adjusted using several methods, and the target values are to be reached at the transverse bulkheads on either side of mid hold. Method 1 was applied when the vertical shear force exceeded the target at only one of the two transverse bulkheads, and the other bulkhead was within permissible limits after adjustment. This method involved calculating a single vertical bending moment, My, which was then applied at either the fore or aft end of the model. The location for applying this moment depended on which bulkhead exceeded the target value, and the direction of hogging or sagging in the load case. The magnitude of My, was calculated using CSR formulae, which take into account the longitudinal distance between bulkheads, the shear force imbalance, and reaction forces at the supports.

The process of implementing method M2 for shear force adjustment was more involved. Method M2 is executed when the condition arises that the shear force on both aft and fore bulkhead exceeds target values, or if the target values are exceeded after adjustment using method M1 on the non-adjusted location. In order to adjust the shear force when method M2 was required, all the longitudinal elements of each individual web frame had to be used for calculation. This is since method M2 requires adjustment on the longitudinal stations along the length of the ship, in addition to the moment My applied to either end of the model, similar to method M1. To adjust the intermediate frames between the aft and fore end of the ship, a vertical load was calculated for each longitudinal station. This vertical load had to be applied by following the shear flow distribution of the respective cross section of the longitudinal station. To calculate the shear flow for each cross section, the moment of inertia and first moment about the neutral axis of the cumulative section area starting from the open end had to be calculated as well. The result is a vertical force Fj, which was applied to each node of the longitudinal elements of each intermediate cross section along the midship model.

After the required shear force adjustments, My and Fj had been calculated, the bending moment adjustment could be calculated. The first step was to calculate the peak value of the bending moment along the ship, which involved taking the

extreme value of the bending moment caused by local loads and the moment caused by shear force adjustment, along with the moment line load. The lineload was taken as zero, unless method M2 for shear force adjustment had been used, in which case the vertical loads applied to the intermediate web frames of the model caused bending moments which needed to be taken into account. Using the peak value and the target value yielded the final needed vertical bending moment adjustment, which includes both shear force and bending moment adjustments. A similar procedure was also implemented to reach the required horizontal bending moment adjustment. The vertical bending moment (see figure 5.9), adjusted to meet the target value, can be seen below in figure 5.11

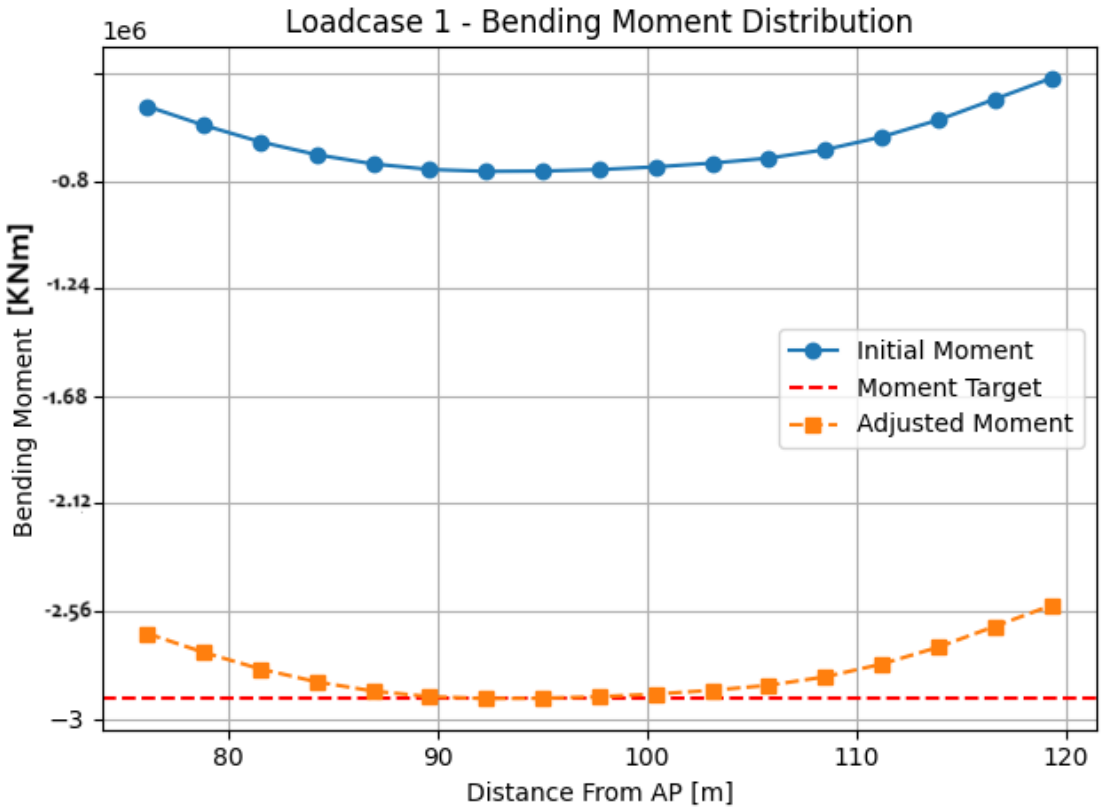


Figure 5.11: Adjusted Vertical Bending moment distribution for loadcase 1

The longitudinal forces for each webframe where added to get total longitudinal force, which needed to be equalized by adding opposing nodal forces to the aft end of the midship model (the side unconstrained in the x direction).
Finally, the bending moment adjustments could be applied to the midship model. This was done using the MSC Apex API to create forces on each of the nodes of the longitudinal elements on the aft and fore ends of the midship. This force was calculated according to the CSR formulae, taking net sectional area, neutral axis, and moment of inertia into account. The calculated forces to adjust both vertical and horizontal bending moment were added together with longitudinal force adjustment to reach the final applied force vectors. An example of the hull girder adjustments

using the shear force adjustment method M1 can be seen in figure 5.12, and method M2 in 5.13

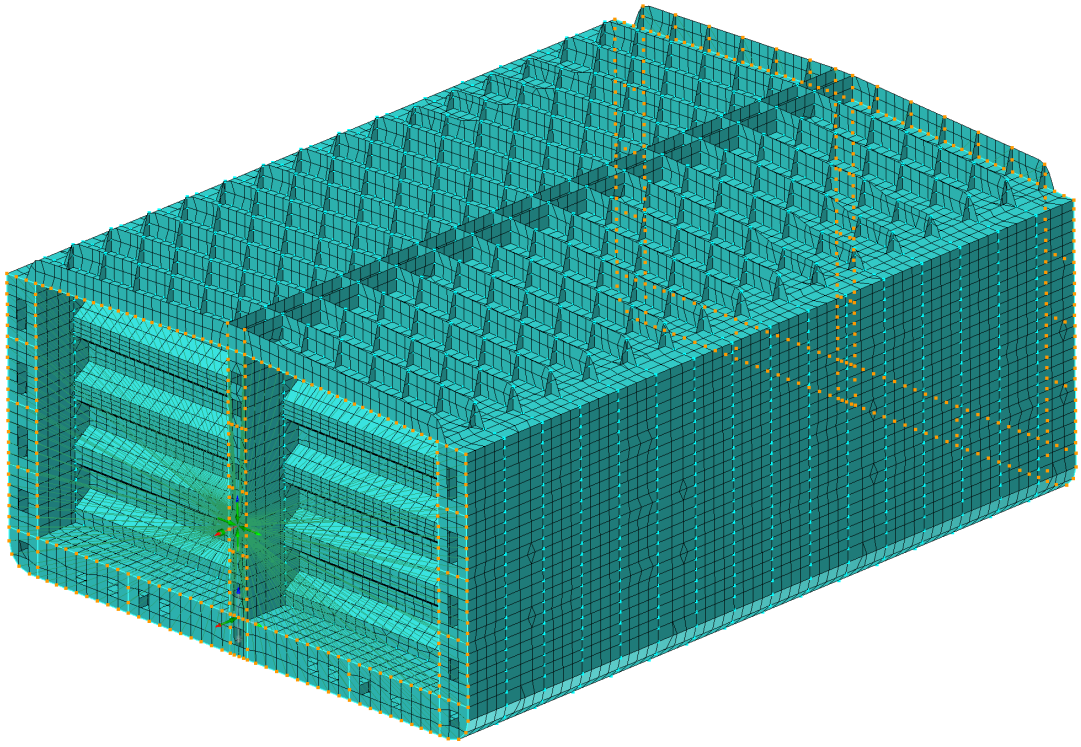


Figure 5.12: Nodal forces (highlighted in orange) applied to model ends to adjust hull girder loads in method M1.

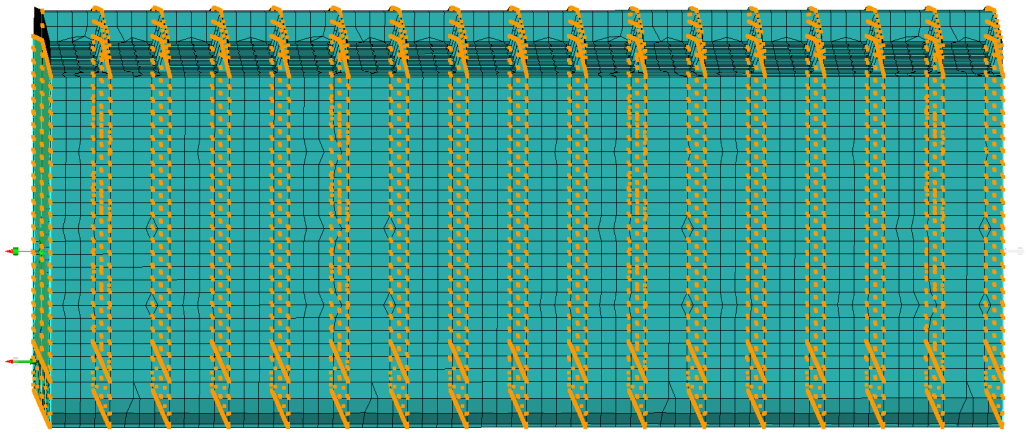


Figure 5.13: Nodal forces are applied at each webframe to adjust hull girder loads in method M2

From the lumped forces, there was also the total torsional moment at each webframe, which was used together with the target value (for most loadcases zero) to get the adjustment, with the addition of a moment that arises due to the horizontal reactionary force from the elastic line equation. This adjustment was applied to the model using the MSC Apex API to add a simple torsional moment to the neutral axis free point on either end of the midship model, as seen in figure 5.14.

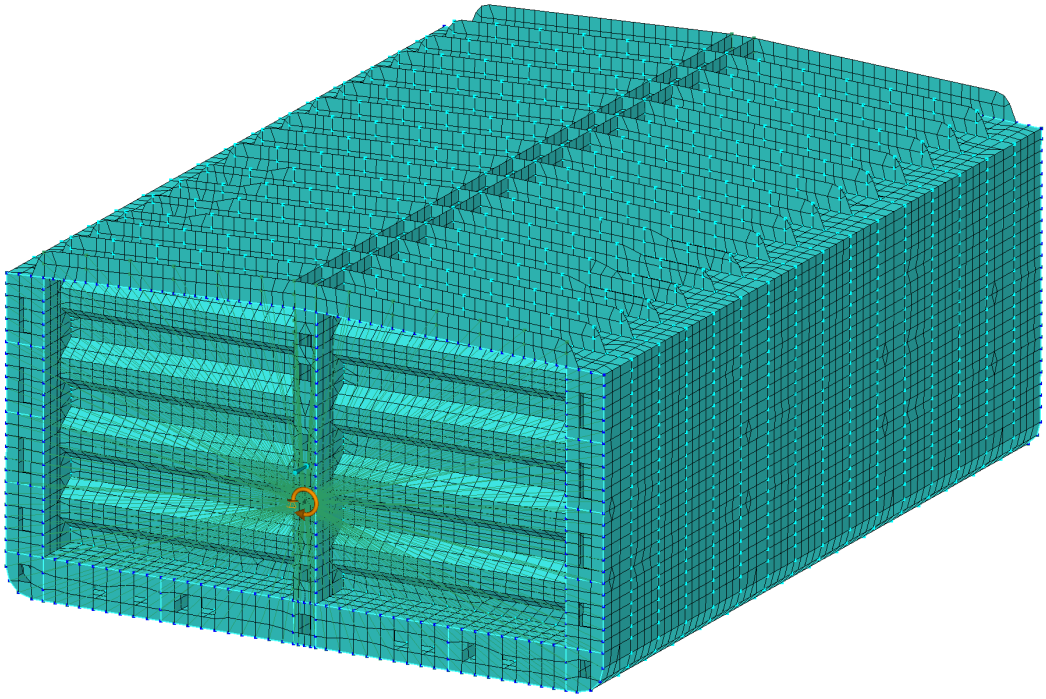


Figure 5.14: Torsional moment is automatically applied to the independent point in MSC Apex

6
Results

This chapter details the outcomes of the developed load-application framework and its impact on hull-structure analysis workflows. It begins by summarizing the software implementation, then illustrates how the new system streamlines and flexibly adapts existing processes.

6.1 Software implementation summary

The final deliverable is a fully automated load-application framework integrated into MSC Apex via its Python API. Inputs (principal particulars, webframe definitions, load cases, etc.) are managed through structured Excel files. A data-generation module parses these inputs, interpolates CSR correction factors, computes nodal mass from PSHELL/MAT properties, and extracts cross-sectional properties for boundary-condition setup. Subsequent modules calculate local pressure fields and integrate them into hull-girder forces and moments, applying both via MSC Apex's API to generate constrained FE models ready for Nastran analysis. The workflow mirrors existing DFC input conventions, smoothing the path to a future DFC graphical interface that will consolidate all input and output within a single window.

6.2 Streamlined workflow

One of the primary achievements of the developed software is the transition from MSC Patran, used by, for example, CSR FE, to MSC Apex, significantly improving workflow efficiency and flexibility. MSC Apex provides a modern, intuitive interface that simplifies tasks traditionally associated with maritime structural analysis, such as geometry preparation, mesh manipulation, and visualization. By adopting MSC Apex, the new workflow reduces the complexity of model preparation and enhances overall usability for designers who typically prefer intuitive and interactive modeling tools.

Additionally, this workflow incorporates interfaces already familiar to designers, specifically Excel spreadsheets and MSC Apex. Excel is commonly used in industry for managing input data, and its inclusion as a user interface significantly reduces the learning curve for new users. Designers can easily manage complex data inputs, modify values, and document project parameters within familiar spreadsheet templates. These Excel templates facilitate consistent and repeatable analyses, allowing

users to easily reuse and adapt settings across multiple projects, thereby streamlining the overall project initiation phase.

An important advantage introduced in this new workflow is the complete integration within MSC Apex, removing the need for manual export, import, or intermediate data handling steps between multiple software packages. As illustrated in Figure 6.1, the entire process from initial model definition to final structural assessment occurs seamlessly within a single FE software environment. This integration eliminates common sources of error and significantly reduces time spent managing data files, thereby enhancing productivity and reducing overall analysis lead time.

Furthermore, the developed software provides enhanced flexibility by enabling users to easily adjust all load combination factors, load scenarios, and load patterns directly from the Excel input interface. Users can also modify intermediate calculation parameters, such as vessel accelerations, before executing the load calculation routines. This functionality empowers designers to perform parametric studies rapidly, compare alternative design scenarios efficiently, and refine designs iteratively without the overhead typically associated with complex manual recalculations or software interactions.

Lastly, despite the shift towards more geometry-driven workflows, the capability to work with orphan meshes—mesh models lacking explicit geometry is fully maintained. MSC Apex's robust meshing and geometry reconstruction features ensure compatibility with legacy models or externally sourced mesh data. This flexibility is essential for supporting a wide range of real-world scenarios encountered in maritime design, including analyses based on imported models or older, geometryless meshes, ensuring broader applicability and usability across multiple project contexts.

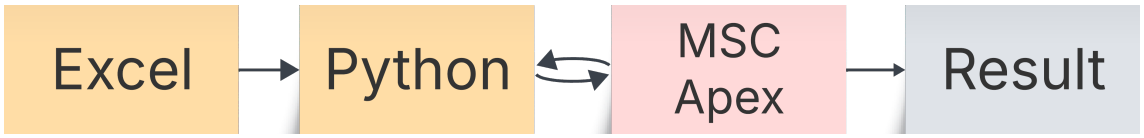


Figure 6.1: The new workflow with the developed software.

7
Discussion

This chapter addresses the outcomes, validations, strengths, limitations, and future potential associated with the software implementation for maritime structural analysis developed in this project. The discussion includes integration considerations for DFC for Maritime, validation approaches, key strengths and limitations identified, and opportunities for future advancements.

7.1 Implementation into DFC for Maritime

While implementation of the developed software into DFC for Maritime is not within the scope of this project, Accentus Aero has already began the process. After integration, all data will be handled using only 2 interfaces, MSC Apex and DFC for Maritime, as can be seen in figure 7.1. Here two interfaces is thought of as two open windows, as opposed to MSC Apex in combination with multiple different instances of Microsoft Excel.

Figure 7.1: The workflow after implementing code into DFC For Maritime.

As Accentus Aero intends to create DFC for Maritime as a commercial software, just as they have done with DFC (aerospace), work remains to implement all functionality. Below is a set of figures exemplifying the DFC interface with one of the user input Excel files integrated into DFC for Maritime (see figure: 7.2 and 7.3).

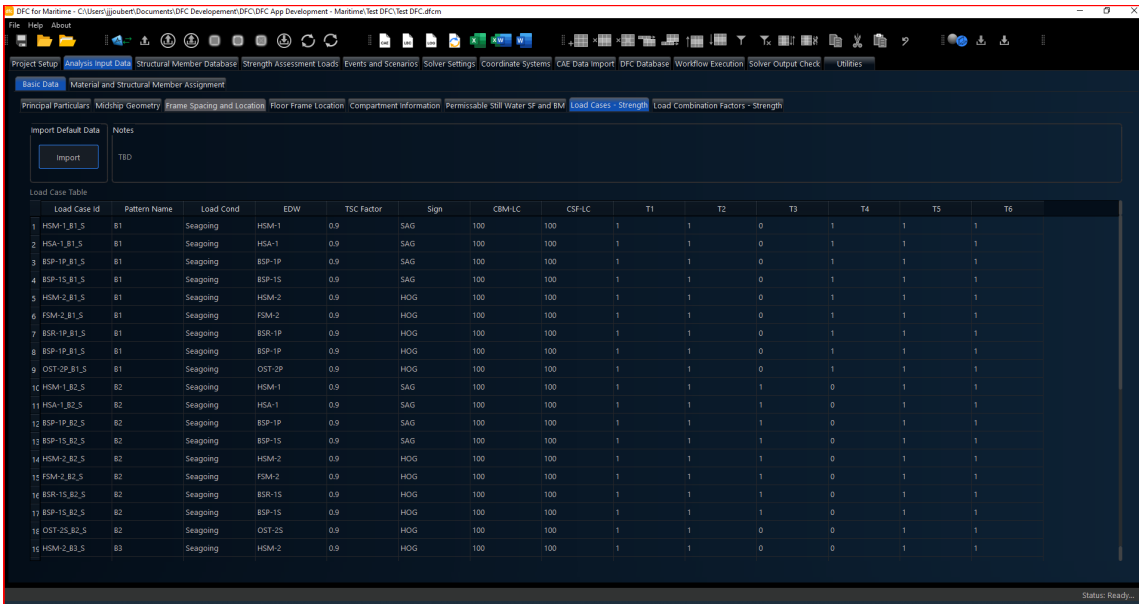


Figure 7.2: The loadcase table integrated into DFC For Maritime.

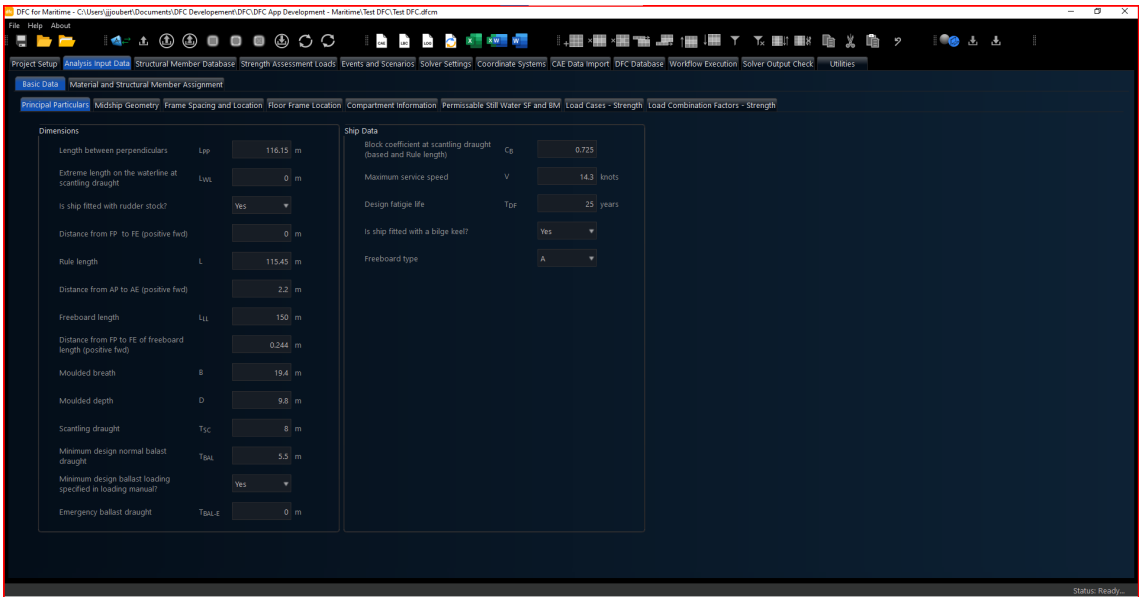


Figure 7.3: The principal particulars user input table integrated into DFC For Maritime.

7.2 Validation Using the Validation Square

Applying the Validation Square framework provided a structured evaluation:

Internal Validation of the Method: Code was verified against Mathcad prototypes to ensure faithful CSR formula implementation (TSV). Inductive reasoning confirmed that midship compartments chosen exercise all algorithmic paths (TPV) .

External Validation of the Method: Automated outputs were benchmarked against commercial CSR FE software on a separate set of hull models, thus confirming cross-platform consistency.

Internal Validation of the Results: Repetitive runs on identical inputs yielded no variance, demonstrating deterministic behavior (ESV).

External Validation of the Results: Pilot integration with existing DFC interface and stakeholder review validated real world applicability (EPV), though further trials across diverse vessel classes are needed before deployment.

7.3 Strengths and Limitations

A key strength of the method is its integration of commercial tools with custom Python automation, balancing user flexibility with structural fidelity. The raytracing-based boundary detection, in particular, represents an innovative approach to identifying compartment boundaries in corrugated or irregular geometries. Likewise, the use of PyNastran and the MSC Apex API proved critical in seamlessly bridging solver input and output.

However, limitations remain. The method currently assumes well-defined compartments and a relatively clean mesh, which may not be the case in all industrial scenarios. Sensitivity to mesh quality and geometric complexity remains a concern. Another limitation lies in the reprocessing that has to be done if the model is remeshed, as the element and node ID:s will change. In the final DFC integration, it is logical to move to a more geometry-based approach for the data generation module, as this will process quicker and generally be less complicated than raytracing. This is possible due to DFC having an integrated tool to convert orphan meshes into geometry, retaining the flexibility of working on mesh-only models.

7.4 Opportunities for Future Work

The final software code remains somewhat unoptimized. This is due to certain MSC Apex API commands, however, there is very likely a better alternative to those specific API commands. For example, applying the pressure field, instead of running the API for that, the pressures could be put in a .csv file and then imported into MSC Apex via the API. This would likely be more efficient, but remains to be tested. For further optimizations, certain nodal force applications could likely be done using other methods than those specified in the CSR, while still yielding the same result. For example, instead of calculating the nodal forces that applies to either end of the model to adjust shear force and bending moment, a simple moment could be added at the neutral axis similar to the torsional moment. Changes like these would reduce the complexity of the software and maybe speed up processing further. However, as it is no longer done exactly per the CSR, more documentation would likely have to supply the software to show that the result remains equal.

The limited timeframe of the project meant that only oil tanker midships in seagoing conditions were included in the software. However, modularity was prioritized, and hence it should be straightforward to integrate bulk carriers, fore and aft-ships, as well as all scenarios. Most of the code for this would be the same, with differing factor values. Similarly, the idea is, of course, that the software should include all rule sets, not only CSR. This inclusion would also likely be easy as the rule sets have converged over time, and again, mostly differ in factor values.

Many of the input values that go into the developed software are taken from non-FE software, such as the permissible still water bending moment and permissible still water shear force. These are crucial for calculation and generally are factors that dimension much of the vessel. As a future inclusion, this could likely be calculated in the DFC software automatically; however, it would likely mean that the entire ship model might be required to calculate it, not only the mid-ship. Other input values include factors such as the Shear Center and block coefficient, which are also taken from another software but can be calculated based on the cross-section geometry of the model. Even simple principal particulars, such as the input values for moulded breadth and height, could be algorithmically extracted from the model automatically instead.

Finally, in order to prevent users from altering the software in any way that is not intended, the code should be compiled. This prevents miscalculations due to user error and protects the integrity of the code. This would need further research, but as an initial though the CPython module could be used to compile the scripts.

7.5 Summary of the Discussion

Integrating the developed software with DFC for Maritime streamlines maritime structural analysis by combining MSC Apex's modeling strengths and DFC's data management. Validation confirmed correctness and potential usability, though broader applicability remains untested. Future work includes software optimization, broader vessel applicability, integration of diverse classification rules, and automation of key calculations within DFC.

8

Conclusion

In this project, the goal was to simplify and unify the way ship structures are analyzed for compliance with the IACS Common Structural Rules by building an automated finite element load applicator within MSC Apex. What began as a survey of disparate classification society tools and FKAB Marine Designs' existing processes quickly revealed both the opportunities and challenges of a single, cohesive environment. Through prototyping in Mathcad, the core formulae and interpolated dynamic-load factors where validated. Then those insights were translated into a modular Python architecture that reads Excel inputs, generates tank data and sectional properties, and applies the full suite of local and global CSR load cases directly to the model via the MSC Apex API.

The result is a designer-centric system where every step: preprocessing, load application, boundary condition setup takes place inside MSC Apex and Excel, eliminating file exchanges and giving engineers the freedom to explore "what-if" scenarios simply by tweaking spreadsheet parameters. Validation against established benchmarks confirmed that the framework not only reproduces CSR-mandated bending, shear, torsion, and local pressure loads accurately but also does so deterministically, ensuring repeatability across midship models.

Although the current implementation focuses on oiltanker midship sections under seagoing conditions, its underlying architecture is agnostic to vessel type or rule set. Looking ahead, extending support to bulk carriers, fore- and aft-ship geometries, and alternative societies such as Bureau Veritas or Lloyd's Register is straightforward. Integrating accidental and fatigue load analyses will further broaden the applicability of the framework, while a future full integration to the Design For Compliance interface promises seamless compliance reporting.

By embedding CSR computations and model setup directly in MSC Apex, this thesis demonstrates a clear path toward faster, more transparent structural analyses in ship design. The framework not only meets todays compliance demands but also lays a foundation for next generation tools, empowering naval architects to push the boundaries of efficiency and innovation in maritime engineering.

Bibliography

[1] Gustaver, M. (2024). *A Chalmers University of Technology Master's thesis template for LaTeX*. Unpublished manuscript, Chalmers University of Technology.

[2] IACS (2024). *Common Structural Rules for Bulk Carriers and Oil Tankers*. International Association of Classification Societies.

[3] Rörup, J. (2017). *FE-based strength analysis of ship structures for a more advanced class approval*. Master's thesis, Chalmers University of Technology.

[4] Blanke, M. (2015). *Simulation of Ship Motion in Seaway*. Technical University of Denmark.

[5] Rawson, K.J. (2001). *Basic Ship Theory*. 5th ed. Butterworth-Heinemann.

[6] Bai, Y. (2015). *Marine Structural Design*. 2nd ed. Butterworth-Heinemann.

[7] Corporate User Publications Group (1989). *The Digital Guide to Software Development*. Digital Equipment Corporation.

[8] Hexagon (2022). *MSC Nastran 2022.4 - Linear Static Analysis User's Guide*. Hexagon AB.

[9] Pedersen, K. (2000). *The Validation Square - Validating Design Methods*. Technical University of Denmark.

[10] Hexagon (2022). *MSC Apex - Unified CAE Environment for Virtual Product Development*. Available at: https://hexagon.com/products/product-groups/computer-aided-engineering-software/msc-apex [Accessed 2 May 2025].

[11] IACS (2025). *Common Structural Rules*. Available at: https://iacs.org.uk/publications/common-structural-rules/ [Accessed 2 May 2025].

[12] CSR Software (2016). *CSR Software products to simplify meeting the CSR with reliability and consistency*. Available at: https://www.commonstructuralrulessoftware.com/uploads/CSRS%20Brochure%20v0.3.pdf [Accessed 15 May 2025].

[13] Accentus Aero (2025). *DFC Software*. Available at: https://www.accentusaero.co.za/ [Accessed 15 May 2025].

[14] FKAB (2025). *About us*. Available at: https://www.fkab.com/ [Accessed 15 May 2025].

[15] Ulrich, K.T. & Eppinger, S.D. (2015). *Product Design and Development*. 6th ed. McGraw-Hill Education. Available at: https://industri.fatek.unpatti.ac.id/wp-content/uploads/2019/03/202-Product-Design-and-Development-Karl-T.-Ulrich-Steven-D.-Eppinger-Edisi-6-2015.pdf [Accessed 15 May 2025].

[16] Kay, T. L. & Greenberg, D. P. (1986). *Ray Tracing in Computer Graphics*. Available at: https://www.researchgate.net/publication/366663130_Ray_Tracing_in_Computer_Graphics [Accessed 1 May 2025].

[17] Masse, M. (2011). *REST API Design Rulebook*. 1st ed. O'Reilly Media.
[18] Van Rossum, G. & Drake, F. L. (2009). *Python 3 Reference Manual*.
[19] Shampine, D. (2024). *pyNastran: Python Interfaces for Nastran Files*. Available at: `https://github.com/SteveDoyle2/pyNastran` [Accessed 15 May 2025].
[20] McKinney, W. (2012). *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*. 1st ed. O'Reilly Media.
[21] Virtanen, P. et al. (2020). *SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods*.
[22] Dawson-Haggerty, D. et al. (2024). *trimesh: A Python Library for Loading and Using Triangular Meshes*. Available at: `https://github.com/mikedh/trimesh` [Accessed 15 May 2025].
[23] Wald, I., Johnson, G., Amstutz, J., Brownlee, C., Knoll, A., Jeffers, J., Gunther, J. & Navrátil, P. (2014). *Embree: A Kernel Framework for Efficient CPU Ray Tracing. ACM Transactions on Graphics*.
[24] PTC Inc. (2023). *Mathcad Prime 9.0 User's Guide*. PTC Inc. Available at: `https://www.ptc.com/en/products/mathcad` [Accessed 15 May 2025].
[25] Eyres,D.J, Bruce, G.J (2023). *Ship Construction*. 7th ed. [Accessed 19 May 2025]
[26] Tupper, E.C. (1996). *Introduction to Naval Architecture*. 3rd ed. Butterworth-Heinemann. Available at: `https://home.hvl.no/ansatte/gste/ftp/MarinLab_files/Litteratur/IntroductionToNavalArchitecture_Tupper1996.pdf` [Accessed 15 May 2025].

A

Appendix 1

Stakeholder	Role/Responsibility	Interest (What They Want)	Influence/Power	Engagement Strategy
Project Lead (Student) (e.g., Jesper Aronsson)	- Oversees day-to-day project execution - Coordinates tasks, timeline, reporting - Primary developer/researcher	- Successful completion of project - Academic success - Personal skill development	High (primary executor of tasks and key decision-maker for daily operations)	- Frequent communication with collaborators (weekly updates) - Close monitoring of progress - Agile adjustments to scope as needed
Examiner (Academic Advisor) (e.g., Gauti Asbjörnsson)	- Ensures academic standards - Reviews methodology and deliverables - Provides final approval for thesis	- Quality research methodology - Compliance with academic requirements - Timely and coherent thesis deliverables	Medium (strong academic authority but less direct control over industry content)	- Periodic milestone reviews - Formal check-ins (progress reports, drafts, final review)
Supervisor (TBD)	- Advises on project scope and feasibility - Provides technical or managerial guidance	- Project aligns with expected outcomes - Timely, coherent execution - Clear scope management	Medium (can shape project direction; final sign-off typically required)	- Ongoing dialogue - Support with risk mitigation and resource allocation
FKAB Marine Design Contact: Aron Jalkesen	- Industry partner/ customer requiring a unified FEA tool - Provides domain expertise - Offers real-world testing scenarios	- More efficient, cost-effective FEA workflows - Compatibility with classification requirements - Potential to reduce multiple-software overhead	High (critical end-user and testing partner; real-world feedback can steer development)	- Detailed requirement gathering workshops - Frequent demonstrations and feedback sessions
Hexagon AB Contact: Stefan Tynelius	- Owner/developer of MSC Apex - Provides technical documentation,API support	- Showcases MSC Apex's extensibility - Expands market adoption of Apex-based solutions	High (controls the core FEA platform; ability to influence technical feasibility)	- Close collaboration on API usage - Timely updates on integration issues - Joint debugging of API challenges
Accentus Aero Contact: JJ Joubert	- Developer of DFC (Design For Compliance) software - Provides codebase insights and structure	- Demonstrates adaptability of DFC to new industries - Preserves software integrity and brand - Gains new market opportunities (marine sector)	High (provides critical code and knowledge base for adaptation; strong technical partner)	- Collaborative code reviews - Shared repository access - Regular sync on feature alignment and test plans

Stakeholder	Role/Responsibility	Interest (What They Want)	Influence/Power	Engagement Strategy
Classification Societies	- Competing software - Provide the Common Structural Rules (CSR)	- Strict adherence to rules and guidelines - Proper validation of FEA solutions - Sell their own software	Small (Changes applied to rulesets annually)	- Keep up with ruleset updates
End Users / Marine Engineers	- Operate and manage the new FEA tool in daily ship design tasks	- Intuitive, efficient software with robust features - Compatibility with existing workflows - Clear documentation	Medium (their adoption determines ultimate success, but they have limited direct control)	- Beta-testing and user feedback - Training sessions and user manuals - Iterative UI/UX improvements based on input
Shipping Industry / Regulatory Bodies	- Broader sector benefiting from improved FEA workflow - Potentially invests in new solutions for more efficient ship design	- Cost-effective, more standardized processes - Potential compliance streamlining	Low (indirect but large-scale influence if tool gains market traction)	

Figure A.1: Stakeholder Analysis

DEPARTMENT OF SOME SUBJECT OR TECHNOLOGY
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY