2026

WSC 2026
Simulation Challenge
Tech Document

NUS National University of Singapore
Centre of Excellence in Modelling and Simulation for Next Generation Ports
Industrial Systems Engineering and Management
College of Design and Engineering

University of Exeter
Engineering

SUMMARY

The Simulation Challenge at the Winter Simulation Conference (WSC), originally launched in 2022 as the "Case Study Competition," continues to showcase the crucial role of simulation in bridging academia and industry. This initiative fosters interdisciplinary collaboration, sparks innovation, and addresses real-world challenges through advanced simulation technologies.

In 2026, aligning with the WSC theme "Simulation for Climate Resilience," the Challenge will focus on managing climate-induced disruptions in complex maritime logistics systems. As global trade increasingly depends on resilient liner-shipping networks, participants are encouraged to develop dynamic, resilience-oriented decision policies to address extreme climate uncertainty. This competition offers a unique opportunity to design adaptive solutions that could enhance system adaptability and shape the future of maritime logistics.

This year's Challenge continues the theme of maritime logistics, with an emphasis on liner-shipping networks exposed to stochastic climate disruptions, covering vessel deployment, port-level operations, and cargo routing. The competition promotes the development, optimization, and evaluation of adaptive decision policies, fostering innovation and effective solutions for resilient maritime logistics.

Participants are invited to design, optimize, and evaluate adaptive decision algorithms with the aim of minimizing transport delays and improving system adaptability under climate-induced disruptions. A foundational simulation model has been built using discrete-event modeling methods and implemented in Python with the o2despy framework, an object-oriented discrete-event simulation framework. Participants are required to build, design, and implement their own strategy algorithms within the standardized simulation environment, competing against other teams. Participants are expected to embed their custom algorithms into specific components of the model, applying their skills in optimization, simulation-based evaluation, and large-scale search to improve maritime logistics performance. The teams that achieve the best overall performance will have the best chance of winning the competition.

This document is intended to provide participants with detailed descriptions of the simulation model structure, along with guidance on file downloads and submission procedures. Through this Challenge, we aim to inspire innovative solutions that can make a lasting impact on the future of the maritime industry.

CONTENTS

Overview

1.1 Problem Description

The 2026 Simulation Challenge continues its focus on maritime logistics, addressing a complex and dynamic problem in liner shipping networks under climate-induced disruptions. This year's challenge centers on improving the interactions and operations across a maritime logistics system, spanning vessel deployment, port-level operations, and cargo routing. Participants are invited to propose innovative and effective solutions that enhance system adaptability and reduce transport delays through simulation and optimization.

Maritime logistics systems are increasingly exposed to operational uncertainty caused by extreme climate events, stochastic disruptions, port congestion, and cascading delays across service routes. These challenges can result in vessel delays, cargo misrouting, capacity imbalances, and reduced reliability of shipping services. This year, the primary objective of the Challenge is to improve the resilience and efficiency of the maritime logistics network by developing smarter and more adaptive decision policies.

To address these challenges, the competition introduces a standardized discrete-event simulation model of a liner-shipping network. The model focuses on key decision areas including vessel deployment, service rerouting, port-level operational decisions, and adaptive cargo routing. These decisions are typically affected by uncertain demand, finite vessel capacity, port conditions, and climate-induced disruptions, leading to complex trade-offs in delay reduction and resource utilization. Participants are encouraged to rethink these operations using innovative data-driven and optimization-based approaches. The model supports exploration of advanced techniques beyond rule-based policies, including optimization algorithms, simulation-based search, and machine learning methods. This opens the door to leverage cutting-edge technologies to significantly improve maritime logistics resilience and performance.

This competition is not only about devising winning strategies but also about fostering learning and discovery in the field of resilient maritime logistics. We hope participants will gain a deep understanding of the complexities involved in liner-shipping operations while exploring creative and efficient solutions to

real-world climate-related challenges. The 2026 WSC Simulation Challenge offers a unique opportunity to apply theoretical knowledge in a practical setting, fostering personal growth and potentially driving transformative change in the maritime industry.

We look forward to seeing the unique perspectives and innovative ideas each team will bring. Join us in shaping the future of maritime logistics, apply your skills, embrace the challenge, and enjoy the process of pioneering resilient maritime operations.

Good luck and welcome to the exciting journey of the 2026 WSC Simulation Challenge!

1.2 Competition Rules

As shown in Figure 1, the given model contains data and source code according to the following three building blocks of information flow:

(A) Input: Example simulation scenarios with specific parameters settings.

(B) Interface: This is where participants may modify the code to implement their own decision algorithms for port response, shipping-line response, and cargo-owner response. The provided interface includes four customizable methods: "select_vessel_for_berth", "create_alternative_service_routes", "assign_associated_bookings", and "adjust_bookings_before_cargo_handling". These methods allow participants to decide which vessel should receive berth priority at a congested port, whether to create alternative service routes and reassign existing vessels, how to assign the initial booking chain for newly generated shipments, and how to replan shipment bookings before cargo handling.

(C) Output: Performance indicators to evaluate the efficiency, reliability, and resilience of maritime logistics operations under climate-induced disruptions.

In this contest, you are expected to rewrite and replace the existing decision modules (in Block B), to maximize the archived performance of the system (in Block C) under different scenarios (from Block A).

You can generate the logic rules in Part (B) in various ways, including but not limited to:

a. Writing rule-based scripts or heuristic algorithms embedded in decision events,
b. Embedding this simulation model into some external optimization search algorithms,

c. Developing a machine learning model to identify and conclude the best rule parameters and embedding them in the decision events.

You **only** need to provide **Part (B)** of the program code (with any required data).

- **Please Note:** Other source codes (except those for Part B) will not be evaluated or run by the assessors.

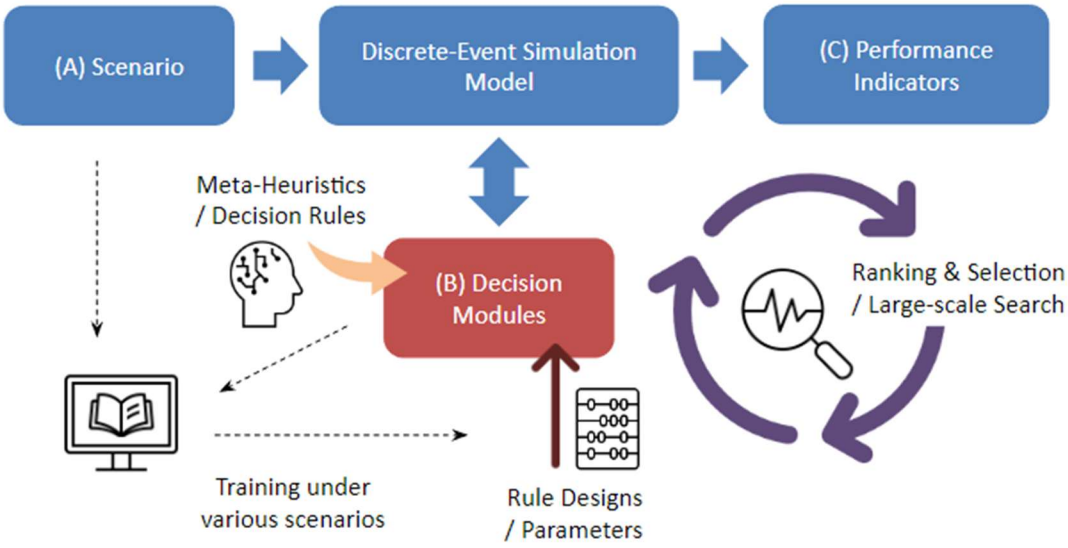


Figure 1 – Information flow diagram.

1.3 General Conditions of the Challenge

Your data and program code will be embedded in the discrete-event simulation model (that we have provided in advance). It will overwrite the corresponding original code, then it will be compiled and generates an executable simulation program. Your program will run under a variety of scenarios and random seeds. The winner will be the one whose model generates the top average performance index for each case.

For further instructions regarding file downloads and programming please check Chapter 2. Simulation model structures and the available data are explained in Chapter 3. The GUI is introduced in Chapter 4. Finally, the submission guidelines and detailed assessment criteria for this competition can be found in Chapter 5.

User Guidance

2.1 Install Python (Windows as the example operating system)

1) Open the official Python download page: https://www.python.org/downloads/
2) Download the Windows installer, version 3.11 or later.
3) Run the installer.
4) On the first installer screen, select Add python.exe to PATH.
5) Click Install Now to complete the installation.

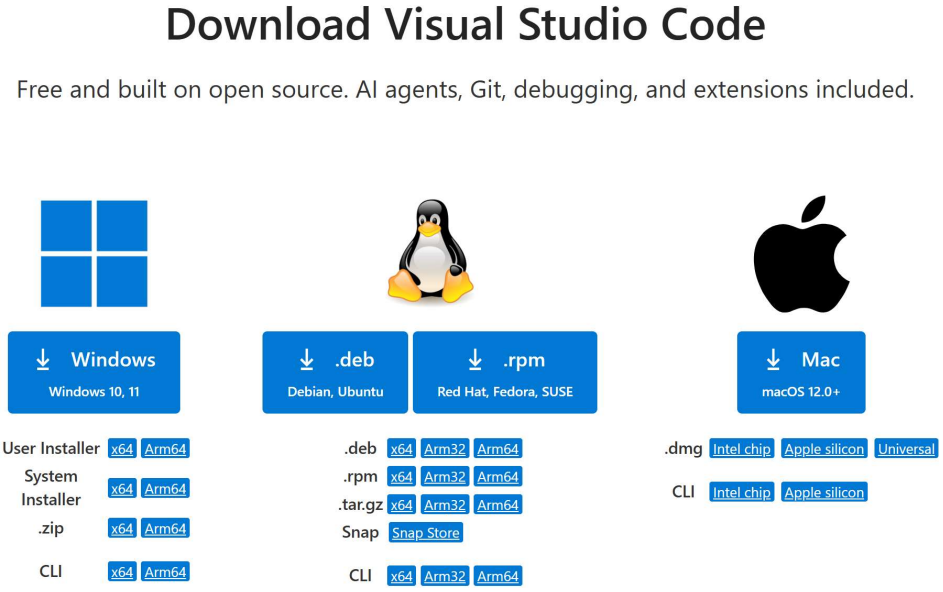
After installation, open PowerShell and check the installed versions:

```
python --version
pip --version
```

If Python and pip version numbers are displayed, Python is installed correctly.

2.2 Download Visual Studio Code

1) Go to website https://code.visualstudio.com/download and download the installer for your operating system.



2) Install on Windows.

Double-click the downloaded .exe installer and follow the setup wizard. During installation, it is recommended to select the following options:

- Add to PATH
- Add "Open with Code" action to Windows Explorer file context menu
- Add "Open with Code" action to Windows Explorer directory context menu

After installation, open Command Prompt or PowerShell and run:

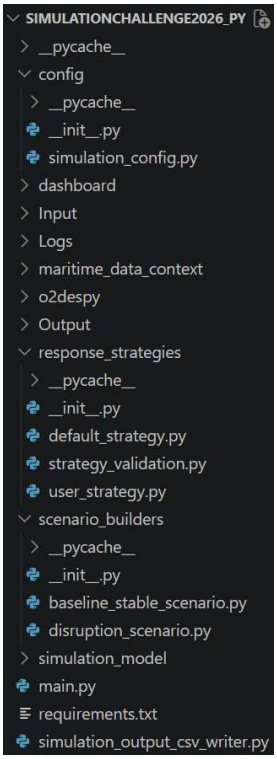
```
code --version
```

If a version number is displayed, VS Code has been installed successfully.

3) Open VS Code and go to Extensions.
4) Search for and install the Python extension published by Microsoft.
   The Python extension allows VS Code to detect interpreters, run Python files, debug code, and show Python-related diagnostics.

2.3 Source Code (Python)

1) After registration and joining in a team, you will receive the zip package of source code by email: "**SimulationChallenge2026_Py.zip**".
2) Decompose zip package.
3) Source code structure of '**SimulationChallenge2026_Py**'.



Note： scenario_builders folder includes scenario files, response_strategies folder includes default_strategy.py and user_strategy.py files and config folder includes simulation_config.py file.

2.4 Open the Project Folder

1) In VS Code, click: File -> Open Folder

Then select your project folder to start working.

2) In VS Code, click: Terminal -> New Terminal

If the terminal does not start in the project folder, use the ‘cd’ command, for example, ‘cd C:\Users\admin\Downloads\SimulationChallenge2026_Py’.

2.5 Create a Virtual Environment

A virtual environment keeps this project's Python packages separate from other Python projects.

```
python -m venv .venv
```

Activate the virtual environment:

```
.\.venv\Scripts\Activate.ps1
```

After activation, the terminal prompt should start with (.venv).

If PowerShell blocks script execution, run this command once:

```
Set-ExecutionPolicy -Scope CurrentUser RemoteSigned
```

Then close and reopen the VS Code terminal and activate the virtual environment again.

2.6 Select the Python Interpreter

1) Press Ctrl+Shift+P.
2) Search for and select Python: Select Interpreter.
3) Select the interpreter in the project virtual environment: .venv\Scripts\python.exe.

This ensures that VS Code runs and debugs the project using its own Python environment.

2.7 Install Project Dependencies

Make sure the virtual environment is active, then run:

```
python -m pip install --upgrade pip
python -m pip install -r requirements.txt
```

This installs the project's third-party packages and installs the local o2despy package in editable mode.

2.8 Run the Simulation

The main entry point for this project is main.py. Run it from the VS Code terminal:

```
python main.py
```

Discrete-Event Simulation Model

3.1 Entities

There are ten entities involved in this model, Port, Berth, Vessel, Vessel Class, Leg, Segment, Service Route, Demand, Shipment and Booking. Each is represented by a class and has its own set of attributes. Figure 2 presents an Entity Relationship Diagram (ERD) that provides an overview of the entities, their attributes, and the relationships among them.

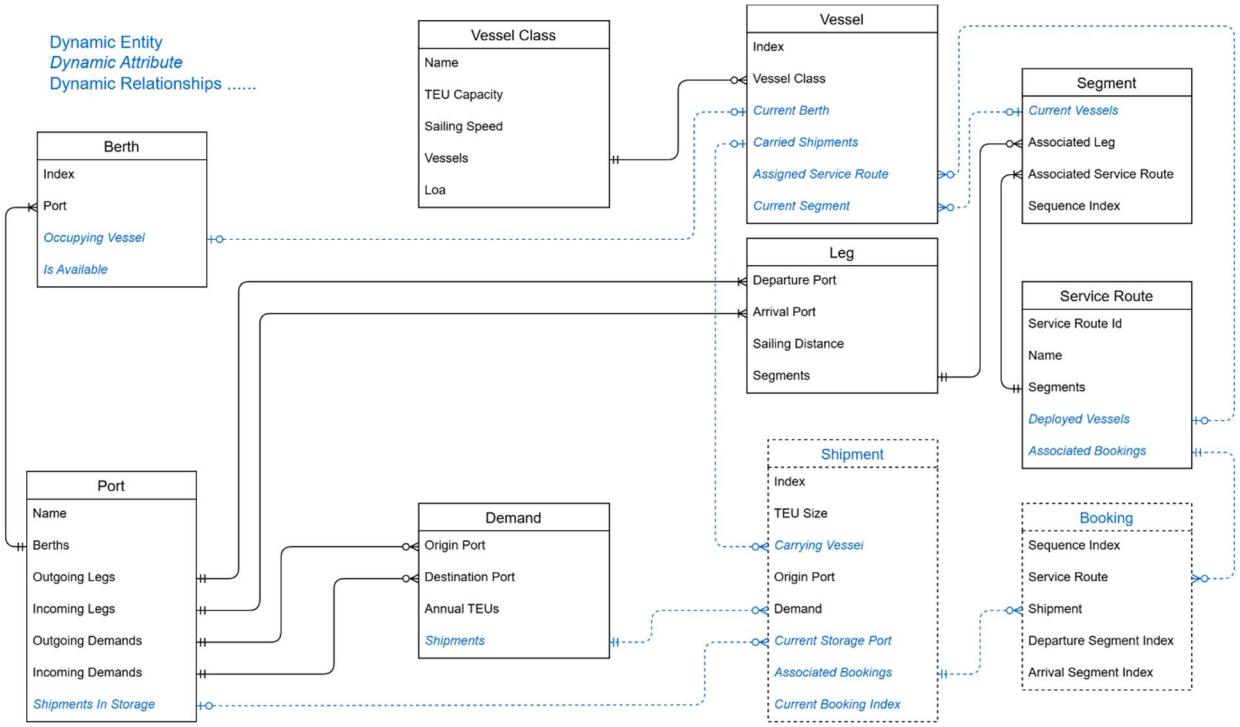


Figure 2 - Entity Relationship Diagram (ERD)

Note:

1) Static attribute: entity attributes that do not change overtime. Static attributes are owned by the class itself (in black color).
2) Dynamic attribute (italic): entity attributes that change overtime (in blue color).

3) The attributes shown in this ERD are the baseline attributes of each entity under the normal simulation setup. When a disruption occurs and response strategies are applied, additional runtime attributes or relationships may be introduced or updated, such as alternative service routes, new route segments, reassigned vessels, and regenerated shipment bookings. Therefore, the diagram represents the core/base entity structure rather than all possible disruption-induced runtime states.

3.1.1 Entity Relationship

1. A Port has one to many Berths.

   A Berth corresponds to one and only one Port.

2. A Port has one to many Outgoing Legs.

   An Outgoing Leg corresponds to one and only one Departure Port.

3. A Port has one to many Incoming Legs.

   An Incoming Leg corresponds to one and only one Arrival Port.

4. A Port has zero to many Outgoing Demands.

   An Outgoing Demand corresponds to one and only one Origin Port.

5. A Port has zero to many Incoming Demands.

   An Incoming Demand corresponds to one and only one Destination Port.

6. A Port can have zero to many Shipments in Storage.

   A Shipment corresponds to zero to one Current Storage Port.

7. A Vessel Class has zero to many Vessels.

   A Vessel corresponds to one and only one Vessel Class.

8. A Berth has zero to one Occupying Vessel.

   A Vessel corresponds to zero to one Current Berth.

9. A Vessel has zero to many Carried Shipments.

   A Shipment corresponds to zero to one Carrying Vessel.

10. A Vessel has zero to one Current Segment.

    A Segment can have zero to many Current Vessels.

11. A Service Route has zero to many Deployed Vessels.

    A Vessel corresponds to zero to one Assigned Service Route.

12. A Leg has zero to many Segments.

    A Segment corresponds to one and only one Associated Leg.

13. A Service Route has one to many Segments.

    A Segment corresponds to one and only one Associated Service Route.

14. A Demand has zero to many Shipments.

    A Shipment corresponds to one and only one Demand.

15. A Shipment has zero to many Associated Bookings.

    A Booking corresponds to one and only one Shipment.

16. A Service Route has zero to many Associated Bookings.

    A Booking corresponds to one and only one Service Route.

3.1.2 Entity Attributes and Definitions

The attributes of each entity and their definitions are explained in detail below.

Table 1 summarizes whether each ERD entity is static, dynamic, or partially dynamic.

Table 1 - Entity Classification

Entity	Classification	Description
Berth	Static Entity	Represents a quay berth at a port. Its occupancy and availability are dynamic.
Port	Static Entity	Represents a node in the maritime network. Stored shipments are dynamic.
Vessel Class	Static Entity	Defines the capacity and physical characteristics of vessels.
Vessel	Static Entity	Vessel instances are initialized from the route plan. Their location and assignments can change.
Leg	Static Entity	Represents a physical port-to-port sailing connection. Disruption can change its sailing multiplier.
Segment	Static + Dynamic Entity	Baseline segments are static. Alternative-route segments may be created during disruption response.

Service Route	Static + Dynamic Entity	Baseline service routes are static. Alternative service routes may be created by strategy.
Demand	Static Entity	Represents annual OD demand. Its generated shipment collection changes over time.
Shipment	Dynamic Entity	Generated during simulation from a demand.
Booking	Dynamic Entity	Created when a shipment is assigned to route segments; may be regenerated during rerouting.

Table 2 summarizes the attributes of Berth.

Table 2 - Attributes of Berth

Attribute Name	Type	Description	Nature
index	int	Identifier of the berth within its port.	Static Attribute
port	Port	Associated port of the berth.	Static Relationship
occupying_vessel	Vessel	Vessel currently occupying the berth.	Dynamic Relationship
is_available	bool	Whether the berth is available. Disruption can close or reopen the berth.	Dynamic Attribute

Table 3 summarizes the attributes of Port.

Table 3 - Attributes of Port

Attribute Name	Type	Description	Nature
name	string	Name of the port.	Static Attribute
berths	List<Berth>	Berths located at the port.	Static Relationship
outgoing_legs	List<Leg>	Legs departing from the port.	Static Reverse Relationship
incoming_legs	List<Leg>	Legs arriving at the port.	Static Reverse Relationship
outgoing_demands	List<Demand>	Demands whose origin is this port. Currently maintained but not used by simulation logic.	Static Reverse Relationship

incoming_demands	List<Demand>	Demands whose destination is this port. Currently maintained but not used by simulation logic.	Static Reverse Relationship
shipments_in_storage	List<Shipment>	Shipments currently stored at the port.	Dynamic Relationship

Table 4 summarizes the attributes of Vessel Class.

Table 4 - Attributes of Vessel Class

Attribute Name	Type	Description	Nature
name	string	Name of the vessel class.	Static Attribute
teu_capacity	int	TEU capacity of vessels in this class.	Static Attribute
sailing_speed	float	Sailing speed of vessels in this class.	Static Attribute
loa	float	Length overall of vessels in this class.	Static Attribute
vessels	List<Vessel>	Vessels belonging to this class.	Static Reverse Relationship

Table 5 summarizes the attributes of Vessel.

Table 5 - Attributes of Vessel

Attribute Name	Type	Description	Nature
index	int	Identifier of the vessel.	Static Attribute
vessel_class	VesselClass	Class defining vessel capacity and speed.	Static Relationship
assigned_service_route	ServiceRoute	Service route assigned to the vessel; may change during disruption response.	Dynamic Relationship
current_berth	Berth	Current berth of the vessel, if berthed.	Dynamic Relationship
carried_shipments	List<Shipment>	Shipments currently carried by the vessel.	Dynamic Relationship
current_segment	Segment	Current sailing segment of the vessel.	Dynamic Relationship

Table 6 summarizes the attributes of Leg.

Table 6 - Attributes of Leg

Attribute Name	Type	Description	Nature
departure_port	Port	Port from which the leg departs.	Static Relationship
arrival_port	Port	Port at which the leg arrives.	Static Relationship
sailing_distance	float	Sailing distance of the leg.	Static Attribute
segments	List	Service-route segments using this physical leg.	Static Reverse Relationship

Table 7 summarizes the attributes of Segment.

Table 7 - Attributes of Segment

Attribute Name	Type	Description	Nature
sequence_index	int	Order of the segment within its service route.	Static Attribute
associated_leg	Leg	Physical leg represented by this route segment.	Static Relationship
associated_service_route	ServiceRoute	Service route that owns this segment.	Static Relationship
current_vessels	List<Vessel>	Vessels currently on the segment.	Dynamic Relationship

Table 8 summarizes the attributes of Service Route.

Table 8 - Attributes of Service Route

Attribute Name	Type	Description	Nature
service_route_id	string	Identifier of the service route.	Static Attribute
name	string	Name of the service route.	Static Attribute
segments	List	Ordered segments that form the route.	Static Relationship
deployed_vessels	List<Vessel>	Vessels deployed to the route. This can change when vessels are reassigned.	Dynamic Relationship
associated_bookings	List<Booking>	Bookings assigned to the route.	Dynamic Relationship

Table 9 summarizes the attributes of Demand.

Table 9 - Attributes of Demand

Attribute Name	Type	Description	Nature
origin_port	Port	Origin port of the OD demand.	Static Relationship
destination_port	Port	Destination port of the OD demand.	Static Relationship
annual_teus	int	Annual TEU demand volume.	Static Attribute
shipments	List<Shipment>	Shipments generated from this demand.	Dynamic Relationship

Table 10 summarizes the attributes of Shipment.

Table 10 - Attributes of Shipment

Attribute Name	Type	Description	Nature
index	int	Identifier of the shipment.	Static Attribute
teu_size	int	TEU size of the shipment.	Static Attribute
carrying_vessel	Vessel	Vessel currently carrying the shipment.	Dynamic Relationship
origin_port	Port	Origin port of the shipment; derived from its demand in code.	Derived Relationship
demand	Demand	Demand from which the shipment was generated.	Static Relationship
current_storage_port	Port	Port where the shipment is currently stored, if not onboard.	Dynamic Relationship
associated_bookings	List<Booking>	Bookings assigned to this shipment.	Dynamic Relationship
current_booking_index	int	Current booking sequence index for the shipment.	Dynamic Attribute

Table 11 summarizes the attributes of Booking.

Table 11 - Attributes of Booking

Attribute Name	Type	Description	Nature
sequence_index	int	Order of the booking in the shipment's booking chain.	Static Attribute
service_route	ServiceRoute	Service route used by the booking.	Dynamic Relationship
shipment	Shipment	Shipment associated with the booking.	Static Relationship

departure_segment_index	int	Departure segment sequence index.	Static Attribute
arrival_segment_index	int	Arrival segment sequence index.	Static Attribute

3.2 Event Flow Diagram

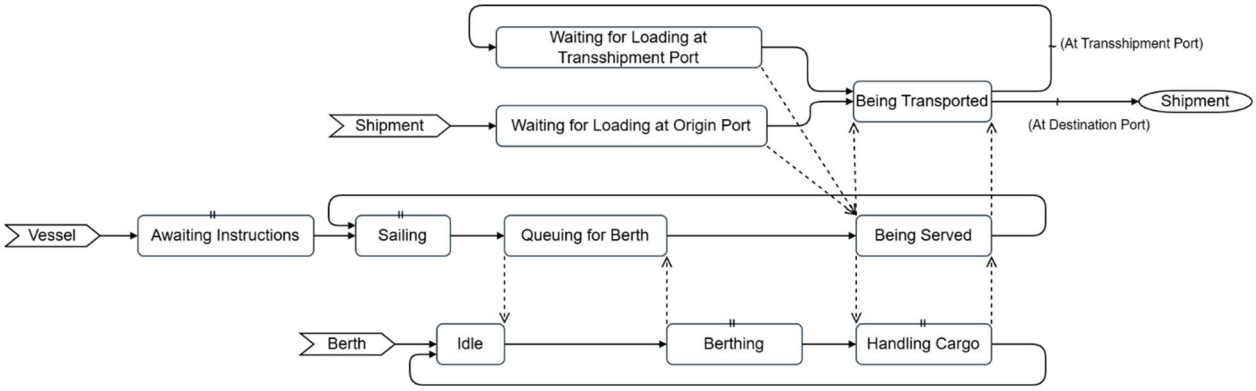


Figure 3 - Event Flow Diagram for the Model

The Event Flow Diagram (EFD), see Figure 3, provides a clear representation of the activity flow, including entities, signal sources, and more. To facilitate a better understanding of the EFD, the process is broken down into the following key relationships:

- shipment and vessel
- shipment and transshipment port
- shipment and destination port
- vessel and berth
- berth availability and vessel berthing
- vessel service and cargo handling
- vessel sailing and port departure

The diagram shows that shipments wait for loading either at the origin port or at a transshipment port before being transported by vessels. Vessels first await instructions, then sail to the port, queue for an available berth, berth when resources become available, and are served through cargo-handling activities. The completion of cargo handling triggers the continuation of vessel movement, while shipments either proceed to another transshipment stage or complete their journey upon arrival at the destination port.

GUI Introduction

The GUI is a visual dashboard for the port network simulation. It reads simulation output CSV files and presents the operational status of the system through summary cards, charts, maps, heatmaps, and time-based performance figures. The interface allows users to quickly inspect port congestion, service route utilization, vessel states, cargo flow, and transport performance during the simulation.

The key measurement indicator shown in the GUI is Cumulative Resilience Loss. This indicator measures the accumulated resilience loss of the current scenario compared with the baseline scenario. A lower value indicates that the system experiences less disruption impact, recovers faster, and has better resilience. This indicator will be used as the sole basis for scoring and ranking submissions.

4.1 Data Input

The GUI loads simulation results from the Output folder by default. If the browser blocks automatic local CSV loading, the user can click Upload Output Folder and manually select the simulation output folder. The dashboard mainly uses the following output files:

- Port_Waiting_Statistics.csv: port-level waiting TEU and vessels waiting for berth.
- Service_Route_Utilization.csv: service route capacity, carried TEU, and utilization.
- Average_Origin_Waiting_TEU_By_OD.csv: waiting cargo by origin-destination pair at origin ports.
- Average_In_Transit_TEU_By_OD.csv: average cargo volume currently being transported by origin-destination pair.
- Cumulative_Completed_TEU_By_OD.csv: cumulative completed cargo volume by origin-destination pair.
- Average_Vessel_State_Counts.csv: average vessel counts by operational state.
- ATT_By_Statistics_Interval.csv: period-based average transport time in the current scenario.
- Baseline_ATT_By_Statistics_Interval.csv: period-based average transport time in the baseline scenario.

4.2 Interface Modules

- Summary Cards

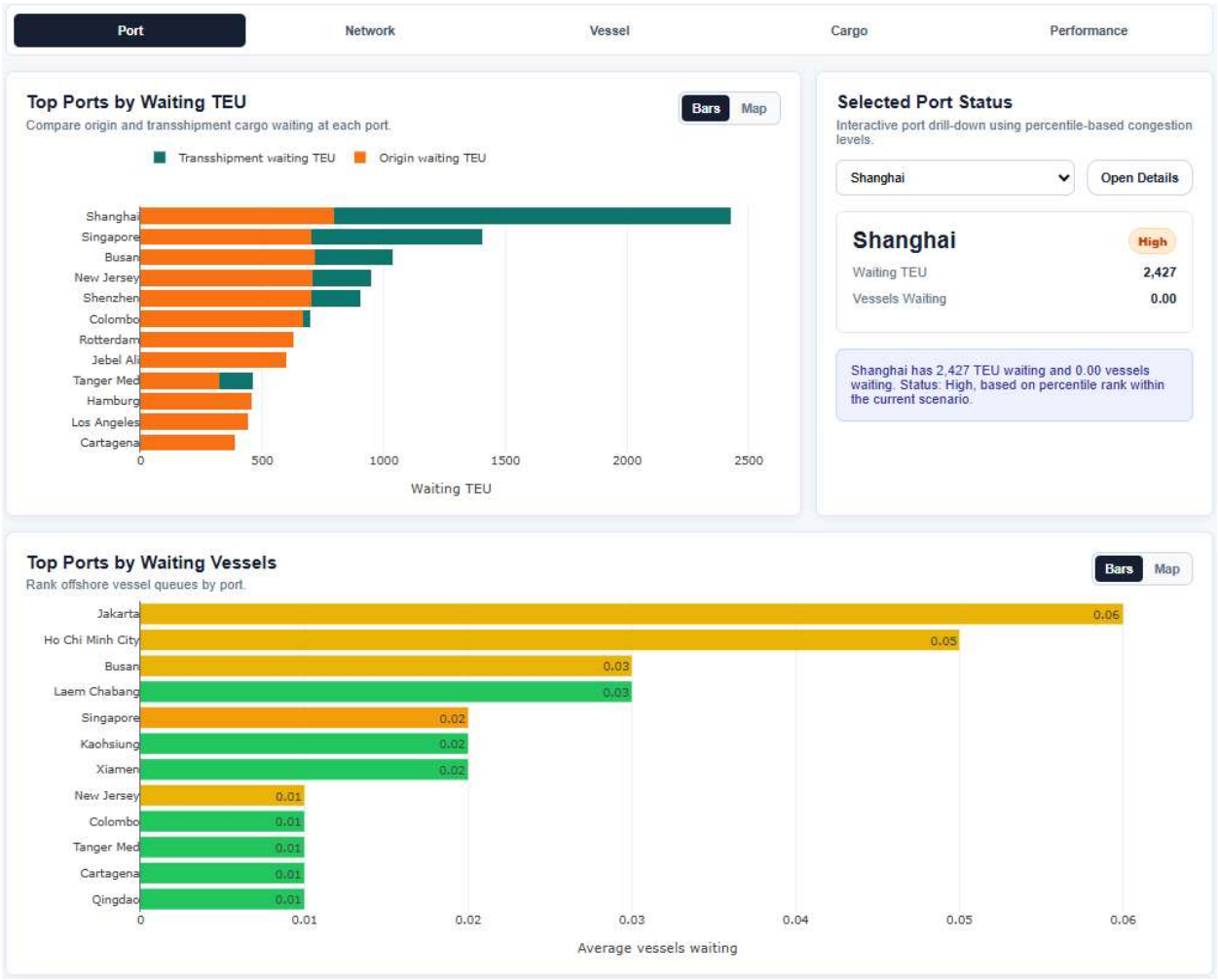
The top of the GUI contains five summary cards for quickly reviewing the current simulation scenario:

- Main Risk Port: the port with the highest combined congestion score based on waiting TEU and vessels waiting.
- Route Utilization: the average utilization of the service route network.
- Vessels Waiting: the average number of vessels waiting for berth.
- Total Waiting TEU: the total cargo volume waiting at origin and transshipment ports.
- Cumulative Resilience Loss: the accumulated resilience loss during the simulation and the main measurement indicator in the GUI.

The **Upload Output Folder** button is used to manually load simulation output data into the GUI. In general, after the program finishes running, the browser will automatically open the GUI webpage and load the generated data. If the webpage does not open automatically, users can open it manually by double-clicking index.html in the dashboard folder. If the data is not loaded automatically, users can click the **Upload Output Folder** button and select the entire Output folder to upload the simulation results into the GUI.

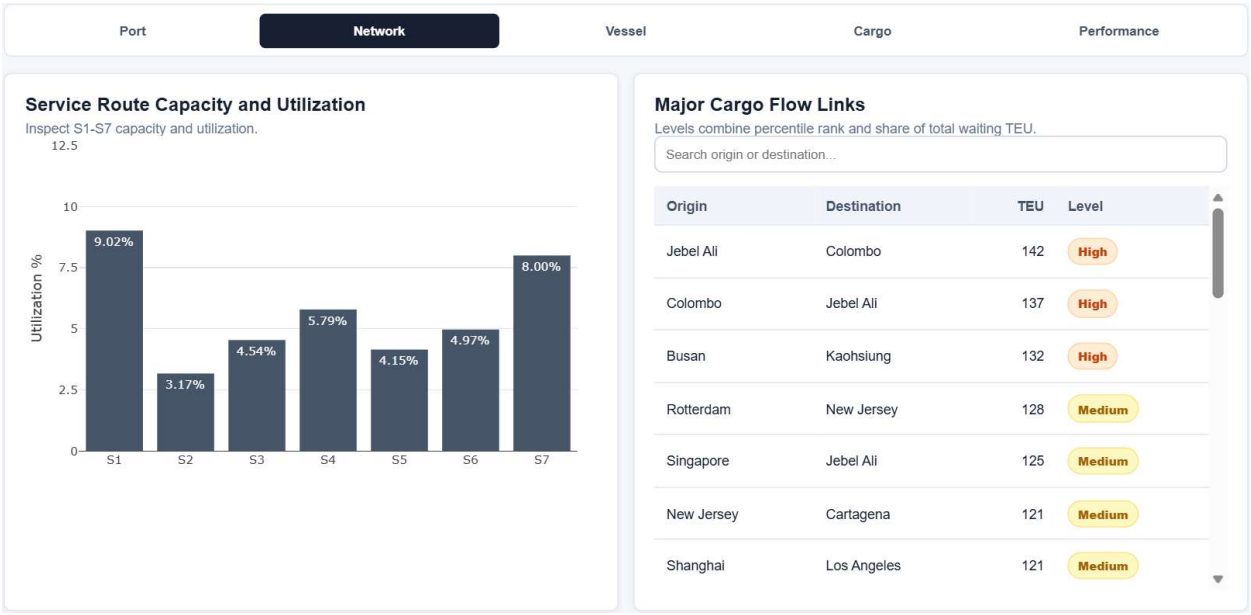
- Port Page

The Port page focuses on port-level congestion. It shows the top ports by waiting TEU and the top ports by vessels waiting. Users can switch between bar chart and map views. They can also select an individual port to inspect waiting cargo, vessels waiting, and congestion level.



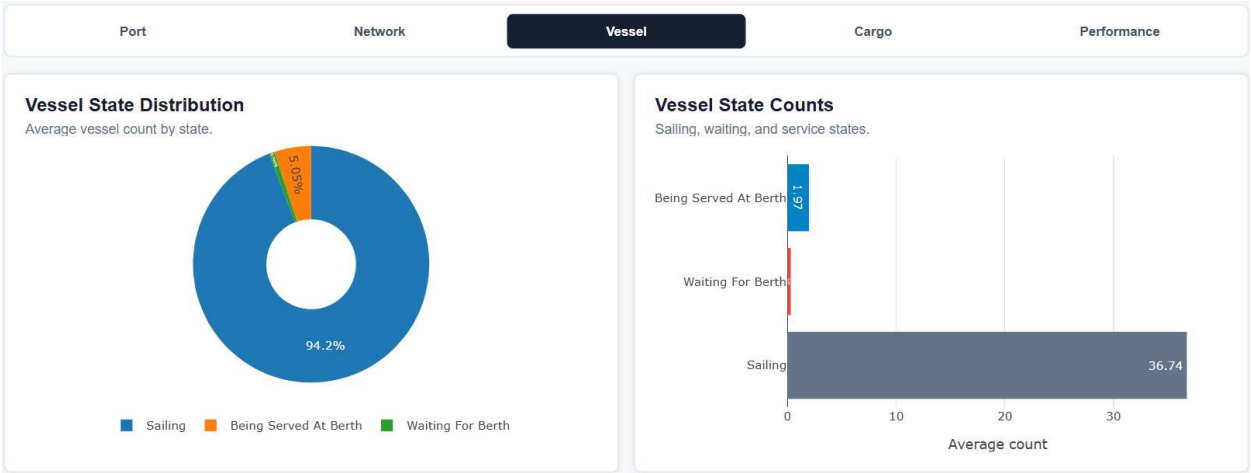
- Network Page

The Network page analyzes service routes and major cargo flow links. The route chart compares route capacity, carried TEU, and utilization. The cargo flow table lists important origin-destination pairs by waiting TEU, helping users identify high-pressure transport relationships.



- Vessel Page

The Vessel page presents the distribution of vessel states, such as sailing, waiting for berth, and being served. This page helps determine whether a disruption leads to longer berth queues or reduced service activity.

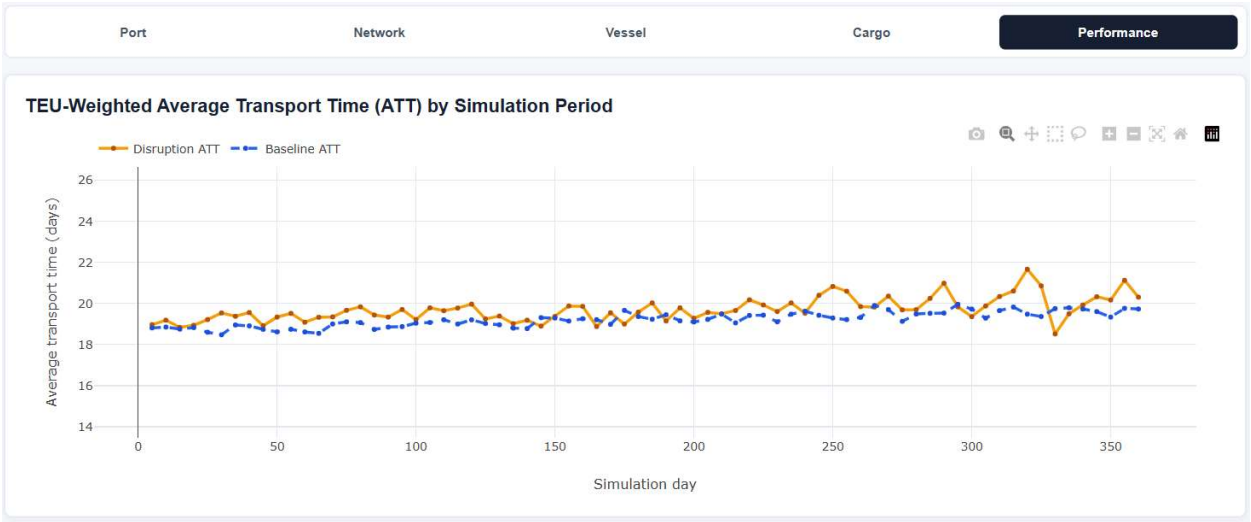


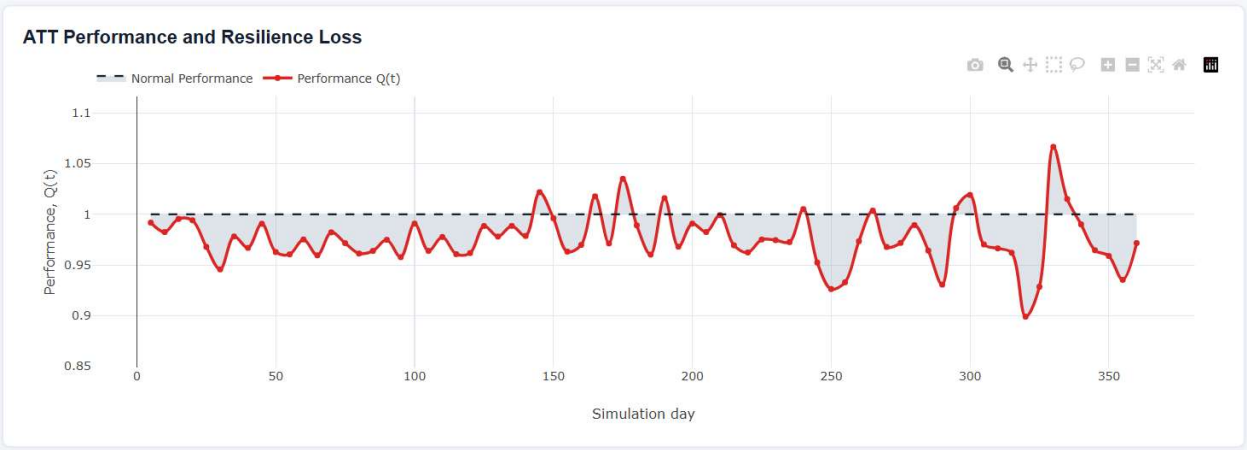
- Cargo Page

The Cargo page visualizes cargo status through origin-destination heatmaps. Users can switch among Waiting, In Transition, and Completed views to inspect waiting cargo, cargo being transported, and cumulative completed cargo.

- Performance Page

The Performance page analyzes system performance over time. It includes period-based ATT, resilience loss, and the comparison between baseline and scenario transport performance. This page is the main place for explaining the changes in Cumulative Resilience Loss.





4.3 Cumulative Resilience Loss Indicator

Cumulative Resilience Loss measures the accumulated performance loss of the current scenario compared with the baseline scenario. The GUI calculates this metric based on period-based ATT, where ATT represents the TEU-weighted average transport time.

For each reporting period, the GUI first calculates the ATT performance ratio:

ATT performance ratio = Baseline ATT / Scenario ATT

Then it calculates the resilience loss for that period:

Period resilience loss = (1 - ATT performance ratio) * Number of days in the period

Finally, all period losses are accumulated:

Cumulative Resilience Loss = Sum of period resilience losses

If the scenario ATT is higher than the baseline ATT, the ATT performance ratio decreases and the period resilience loss increases. If the system gradually recovers and ATT moves closer to the baseline level, the additional loss in later periods becomes smaller. Therefore, this metric reflects both the severity and the duration of disruption impact.

Evaluation

5.1 File Submission Format

1) Zip package of **xxx** folder

   Note: Any modification should be put under the **response_strategies** folder, including newly created files for algorithms purposes and explanatory documents if necessary. Changes made in other folders will not be considered.

2) Save your code and name it in the following format: **Team Name_Round Number** (e.g. **SealTeam_Round1.zip**)

5.2 File Submission Method

Email address for submission: wsc2026simchallenge@gmail.com

5.3 Evaluation Criteria

The criteria used for evaluation in this simulation challenge are outlined as follows:

1) The strategy proposed by each competitor will be implemented to simulate both the given scenario and the hidden scenario in the last round.
2) "**Cumulative Resilience Loss**" shown on the web-based GUI is the sole criterion for evaluating system performance.
3) In the evaluation stage, multiple random seeds will be used to calculate the average performance for each round. Only codes that run successfully will be eligible for scoring.
4) Weightage and score of each round will be computed as follows:

Weightage Table

Round Number	Round 1	Round 2	Hidden Round
Weightage	20%	30%	50%

The full of Each round has a maximum score of 100 points. For the top 10 teams, scores will decrease in increments of 5 based on their ranking (i.e. the first-place team will receive 100 points, the second-place team 95 points, etc.). All teams ranked below the top 10 will receive a fixed score of 50.