Evaluating Carrier Assignment and Relocation Strategies in Autonomous Pod-based Railway Systems Using Discrete-Event Simulation

Thesis Project

by

Aditya Pavadad

Msc. Transportation , Infrastructure and Logistics (TIL)

Student number: 6004695

Thesis committee: Dr.ir. M. Saeednia, Dr.Patrick Stokkink , Ir. N.D. (Nina) Versluis

TU Delft

Abstract

Autonomous pod-based railway systems represent a promising innovation in freight transport, combining the flexibility of modular vehicle concepts with the efficiency of rail-based logistics. Their success, however, depends on effectively managing the assignment and relocation of carriers under dynamic and uncertain operating conditions. Static optimization approaches such as Mixed-Integer Linear Programming (MILP) can design efficient baseline schedules, but these often prove fragile when confronted with real-world uncertainties such as delays, carrier breakdowns, and structural disruptions. To address this gap, this thesis develops a hybrid decision-making framework that integrates MILP-based planning with a Discrete-Event Simulation (DES) environment, enabling dynamic re-optimization and real-time resilience analysis.

The framework operates through an event-driven feedback loop: initial assignments are optimised using MILP, executed within the DES, and re-optimised whenever disruptions occur. This coupling allows for continuous adjustment of plans to reflect the real-time state of the system. The methodology was validated through two case studies: a simplified Toy Case to verify the model logic and a large-scale Randstad network to evaluate system robustness against cascading disruptions. The analysis incorporated a range of scenarios, including probabilistic delays, deterministic breakdowns, arc removals, transport unit (TU) insertions at varying time brackets, and the impact of delivery-window flexibility.

Results demonstrate that re-optimisation is highly effective in recovering service levels after disruptions, significantly improving fulfillment rates and resource utilisation compared to static schedules. The system showed strong adaptability to sudden demand surges and carrier failures by reallocating idle resources, while also highlighting resilience thresholds in cases of severe network degradation. Time-window flexibility was found to play a dual role: it improved overall fulfillment but introduced delays, suggesting trade-offs that need to be balanced by operators and policymakers. Carrier utilisation and empty travel metrics further revealed how resilience is achieved at the cost of increased repositioning.

This study contributes both theoretically and practically. It establishes an integrated simulation–optimisation framework that advances research on disruption management in autonomous rail systems, and it provides operational insights on fleet pre-positioning, flexible service design, and the role of redundancy in network infrastructure. The findings emphasise that digital re-planning capabilities are necessary for future autonomous freight systems, while industrial and policy implications include incentivising early bookings, setting flexibility standards, and ensuring investment in redundant routing capacity. Overall, the proposed framework provides a powerful tool for designing, testing, and improving resilient autonomous pod-based rail networks, bridging the gap between strategic planning and operational execution under uncertainty.

Contents

Abstract i

1 Introduction 1
- 1.1 Research question, aims and objectives . . . 2
- 1.2 Conceptual flowchart . . . 4
- 1.3 Experimental Set-up . . . 5
- 1.4 Results, Outcome and Relevance . . . 5

2 State of the art 7
- 2.1 Factors influencing carrier-TU matching. . . . 8
- 2.2 Repositioning of empty carriers . . . 9
- 2.3 Literature gaps and contributions . . . 11

3 System Dynamics 12
- 3.1 Simulation Flow Process . . . 12
- 3.2 Introduction to State-Event Modelling in TU–CU Systems . . . 13
  - 3.2.1 Transport Unit (TU) States . . . 14
  - 3.2.2 Carrier Unit (CU) States . . . 16
- 3.3 System events and transitions . . . 18

4 Methodology 24
- 4.1 Case Studies . . . 24
  - 4.1.1 Toy Case . . . 24
  - 4.1.2 Randstad Case Study . . . 25
- 4.2 Discrete Event Simulation . . . 27
  - 4.2.1 Inputs from MILP . . . 27
  - 4.2.2 Simulation Algorithm . . . 29
- 4.3 Disruption Implementation . . . 32
  - 4.3.1 Toy Case . . . 32
  - 4.3.2 Randstad Case Study . . . 33
  - 4.3.3 A General Framework for Sequential Disruption Analysis . . . 34

5 Results & Discussion 37
- 5.1 Key Performance Indicator (KPI) Selection . . . 37
  - 5.1.1 Category 1: Service Effectiveness & Quality . . . 37
  - 5.1.2 Category 2: Resource Efficiency . . . 38
  - 5.1.3 Category 3: System-Specific Performance . . . 38
  - 5.1.4 Discussion . . . 39
- 5.2 Toy Case . . . 39
  - 5.2.1 KPI Summary . . . 40
  - 5.2.2 Aggregate Analysis (200 Runs) . . . 42
- 5.3 Randstad Case Study . . . 44
  - 5.3.1 TU Addition . . . 44
  - 5.3.2 CU Breakdown . . . 53
  - 5.3.3 Arc Removal . . . 57

5.3.4 System Resilience Under Multiple Disruptions 60

6 Conclusion **65**
6.1 Industrial and Policy Insights 66
6.2 Contributions 67
6.3 Limitations and Future Research 67

References **68**

A Appendix **71**
A.1 Summary of Literature 72
A.2 Extract of DES - Master Example 83
A.3 Optimization Model Overview 83
A.3.1 Key Inputs 83
A.3.2 Key Outputs 83
A.4 KPIs for Toy Case 84
A.5 Master Example: Two-CU, Two-TU Scenario 85
A.5.1 Input Data 86
A.5.2 Event Timeline and State Transitions 86

1

Introduction

Intermodal transportation plays a pivotal role in modern logistics by significantly enhancing the efficiency and sustainability of freight distribution. It reduces transportation costs and CO2 emissions, effectively addressing global environmental concerns while maintaining economic efficiency. This system of freight transportation applies different types of vehicles to facilitate movement in an efficient manner while using a standardised transport unit that is unchanged until the goods reach their destination. This method optimises the handling transfers between trucks, trains, ships, and barges, reducing handling times, security risks, damage, and costs. It also integrates the benefits of different means of transport in a single undertaking for efficient and environmentally friendly transport, combining truck and rail services to reach areas without direct barge or rail terminals[1].

There are enormous benefits intermodal transport offers when compared to unimodal road systems, with savings exceeding 20 per cent and reductions in CO2 emissions reaching 57 per cent [2]. This strategy strengthens the efficacy and sustainability not only of freight forwarder processes but also of the entire international supply chain by lowering the need for local emission-reducing measures using intermodal strategies to change transport mode. The rail-freight mode of transportation has seen a downward trend in the last ten years, mainly because of issues like lower flexibility and reliability when compared to its main competitor, the road sector.

This underscores the importance of shifting from reliance on road transport to embracing intermodal transportation systems. As a result, there have been several innovations aimed at improving the utilisation of existing railway systems. One such innovation is the development of autonomous wagons, commonly referred to as ”pods,” which significantly enhance the flexibility and efficiency of rail transport[3].

A “Pod” is a modular vehicle concept characterised by its detachable capsule(Transport unit) and carrier architecture. Pods are essentially modular automobiles with an integrated transport unit and a carrier that can work independently in a larger rail network, as depicted in Fig. 1.1. This system enables Pods to actively join or detach from platoons to increase the flexibility and efficiency of operations. This concept is the core of the Pods4Rail project, which combines different methods of transport in a single unit for prompt transfer. The unit can be delivered by 1) Rail, 2) Road and potentially Rope. The issue of managing the assignment and rescheduling of carriers can be solved through heuristics, which reduces operating costs, increases infrastructure and vehicle utilisation, and represents a new stage in the development of the systems of rail logistics[4].

The optimisation of assigning carriers to (TU) transport units in autonomous pod railway systems is one of the major problems in the development of shared mobility services. Autonomous pods, which integrate the advantages of rail transport and vehicle automation, require drastic coordination in time (e.g., pickup/drop-off times) and space (e.g., predefined geographic railway networks), and on top of that, the repositioning of empty carriers to serve future assignments. To achieve this goal, this work

will formulate an optimisation model embedded into a discrete event simulation (DES) that reduces the costs of operations and guarantees their timely execution. The proposed solution will be demonstrated through simulation, paving the way for significant contributions to both academic research and practical implementation in the transportation sector. This proposal presents enhanced methodologies along with the appropriate simulation framework and the desired outcomes intended toward increased benefits in the use of autonomous pod railway systems.

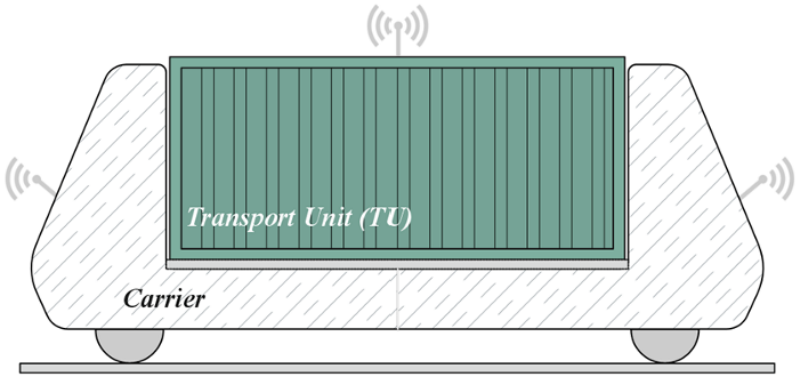

Figure 1.1: Conceptual image of a Pod with one Transport Unit (TU) coupled with a Carrier.

This thesis is structured into seven chapters. Chapter 1 introduces the project and integration of discrete-event simulation (DES) with a MILP model to optimise carrier assignment in autonomous pod-based rail systems. Chapter 2 reviews relevant literature and identifies key research gaps. 3 presents the DES framework, including state-event modelling of carrier and transport units. Chapter 4 details the system implementation and case studies. Chapter 4 explores disruption scenarios and re-optimisation strategies. Chapter 5 presents performance results using key indicators, and Chapter 6 concludes with findings, limitations, and future directions.

1.1. Research question, aims and objectives

This project employs a hybrid approach to address the challenges of optimally assigning carriers to Transport Units (TUs) within an autonomous pod-based railway system, considering operational, spatial, and temporal constraints. It combines the development of an optimisation model with subsequent simulation.

Theoretical Basis of the Work:
As noted in the chapter 1, the focus of the research is that an autonomous rail system is likely to achieve greater productivity along with lower expenditure through effective assignment and relocation of carriers. So as to validate this claim, this research will attempt to use a hybrid approach whereby some static demand cases that prevail in such systems will be treated with a mixture of deterministic and probabilistic models. This approach comprises advanced optimisation algorithms such as Greedy, Tabu Search, and Iterated Local Search Algorithms, which are known to constrain the spatial and temporal status of carriers. The research will also make use of the in-hand simulation software for in-depth scenario analysis and performance metrics assessment. All these techniques make it possible to study the efficiency of the system using a different range of working conditions under different operational case scenarios.

Main Research Question

How can discrete-event simulation (DES) effectively evaluate and enhance the operational performance of autonomous pod-based railway systems, considering predefined carrier assignments and relocation strategies, dynamic demand conditions, operational constraints, and potential disruptions?

Sub-Questions

1. What are the key factors influencing the matching of carriers to TUs in an autonomous pod railway system?
2. How can a discrete-event simulation (DES) model be structured and implemented to accurately capture carrier assignment and relocation dynamics in autonomous pod-based railway systems under static demand scenarios and operational constraints?
3. How robust are carrier assignment and relocation strategies under simulated disruptions (e.g., carrier delays, cancellations, operational interruptions), and how can DES identify opportunities for improvement?

1.2. Conceptual flowchart

To address the challenges of carrier assignment and relocation in autonomous pod-based railway systems, a structured research methodology was developed. The approach combines literature research, data preparation, and identification of key operational factors with the development of both optimisation and simulation models. A Discrete Event Simulation (DES) framework was integrated with a Mixed-Integer Linear Programming (MILP) model to capture dynamic system behaviour and enable re-optimisation during disruptions. The methodology also includes code implementation, performance evaluation using relevant KPIs, and systematic reporting of findings. Together, these steps provide a robust foundation for analysing system resilience and informing both theoretical and practical advancements.

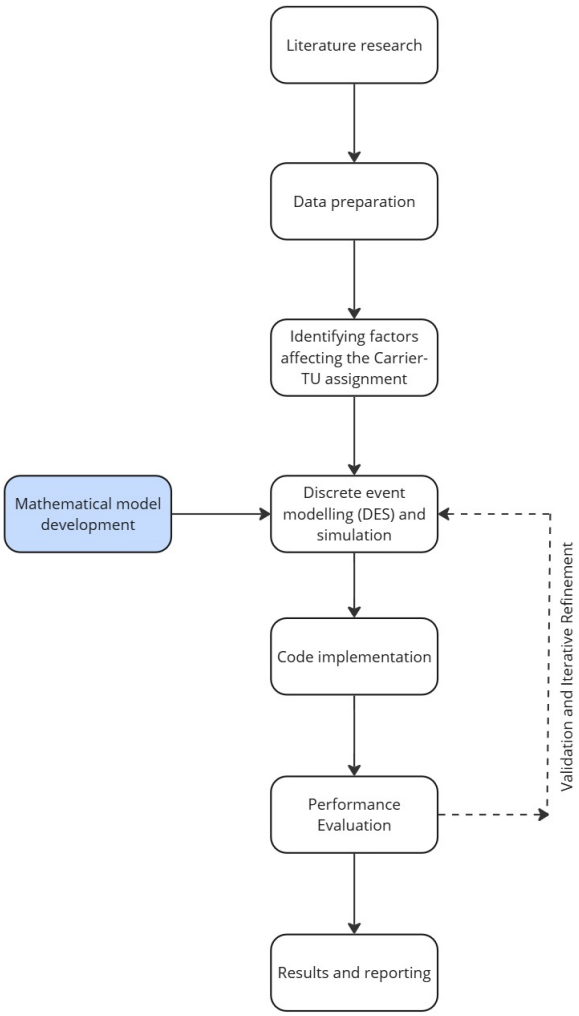

Figure 1.2: Project methodology

1. **Literature Research:** A comprehensive review of existing literature on carrier assignment and relocation strategies in similar transportation systems will be conducted to build a solid foundation.
2. **Data Preparation:** Synthetic data representing static demand within a railway system will be generated, reflecting various operational scenarios, and suitable case studies will be selected.
3. **Identifying Factors Affecting the Carrier-TU Assignment:** To determine the key factors that influence the matching of carriers to TUs, such as temporal constraints (e.g., pickup/drop-off

times), spatial constraints (e.g., station locations), and operational constraints (e.g., carrier availability).

4. **Discrete Event Modelling (DES) and Simulation:** To simulate the dynamic behaviour of the autonomous pod-based railway system and evaluate the performance of the optimisation models [5]. This includes building a DES model that captures events such as pod arrivals, TU pickups/drop-offs, carrier assignments, etc. The DES model integrates the optimisation models to dynamically adjust carrier assignments based on real-time data. A simulation framework that allows testing operational scenarios. Discrete Event Simulation (DES) effectively models macro-level changes in a system, such as the start and end of operations, while omitting continuous micro-level changes, making it suitable for large-scale systems. Additionally, DES enables explicit representation of synchronisation (where multiple conditions must be met for an event to occur) and parallelism (where separate system dynamics evolve independently for a period) [6].
5. **Mathematical Model Development:** This involves taking inputs from an optimisation model (MILP) that incorporates the identified factors and constraints. This model aims to minimise operational costs, reduce turnaround times, and ensure high service reliability. This model gives the assignment details that go as input to a discrete event simulation.
6. **Code Implementation:** The developed models will be implemented in a Python programming environment using SimPy. This includes the development of algorithms capable of finding efficient solutions to the carrier-TU assignment problems.
7. **Performance Evaluation:** The performance of the implemented models will be evaluated against key metrics such as efficiency, service reliability, utilisation, etc. This involves comparing the outcomes of the simulation scenarios with existing work carried out in this field [4]
8. **Results and Reporting:** Findings from the simulations and algorithmic studies will be documented in a detailed report, providing insights into the efficiency of the proposed solutions and their impact on the system's performance.

1.3. Experimental Set-up

This thesis aims to develop a discrete-event simulation (DES) environment in Python and integrate it with the existing MILP optimisation model. This combined framework enables the study of how the system behaves under different disruptions and how it can adapt through re-optimisation.

The DES will be implemented using SimPy to capture dynamic processes such as TU arrivals, CU assignments, and disruption events (e.g., delays, breakdowns, arc closures). The MILP, formulated in Python and solved using Gurobi, provides optimal carrier–TU assignments under baseline conditions. Disruption snapshots from the DES will trigger re-optimisations, allowing the MILP to adapt the plan to the updated system state, while the development of the MILP model is outside the scope of this project, its outputs are integral to the research as they provide the essential CU-TU assignments.

Performance will be evaluated using key metrics such as fulfillment rate, delivery delays, carrier utilisation, and platooning efficiency, with results analysed and visualised through Python libraries (Pandas, NumPy, Matplotlib). .

1.4. Results, Outcome and Relevance

In this investigation, the main data components consist of dynamic demand scenarios, carrier capacities, locations of transport units, and the operational time horizon. The experiments manipulate factors such as carrier distribution strategies, disruption types, and scheduling policies to evaluate system robustness and adaptability.

The expected results go beyond cost and efficiency improvements: the combined MILP–DES framework is designed to reveal how carrier assignments evolve when disruptions occur and how re-optimisation can restore or improve system performance. Anticipated outcomes include quantifiable improvements in fulfillment rates, delivery reliability, and resource utilisation under different disruption settings.

By validating the approach through disruption scenarios (e.g., CU breakdowns, arc closures), this work demonstrates not only how the system can adapt in real time, but also how such methods can inform broader design and operational decisions in autonomous pod-based railway systems. Given its integration into the Pods4Rail project, the results are directly relevant for advancing both the theoretical understanding and practical implementation of resilient, efficient, and scalable rail logistics systems.

Autonomous rail pod-based transport systems are becoming an essential component of modern transportation networks. Improving their operational efficiency is crucial, as it can lead to significantly lower operational costs and enhanced reliability of transport services in the existing infrastructure. Given its integration into the broader pods4rail project [3], the findings from this thesis could directly influence and improve the practical implementation of these systems.

2

State of the art

Within the broader context of the Pods4Rail [3] project, significant advancements have been made in rail-based, intermodal freight transport systems. Liao, Han, and Saeednia [4] have explored ways to enhance system flexibility through modular vehicle routing, focusing on the integration of autonomous wagons in railway environments. Their research addresses key operational challenges like platooning and routing efficiency, demonstrating potential reductions in transportation costs and improvements in capacity utilization. Furthermore, contributed to a second study that introduced a heuristic framework for scheduling these modular vehicles, optimizing makespan and enhancing railway capacity. These collaborative efforts underscore the project's commitment to developing integrated and efficient transportation solutions [7].

Building on the foundation laid by previous research within the Pods4Rail [3] project, the focus now shifts to addressing specific challenges associated with autonomous pod railway systems. While prior studies have optimized broad aspects of intermodal transportation, this research seeks to delve deeper into the nuanced dynamics of carrier-TU interactions within this innovative framework. The ensuing discussion will explore the adaptation of established models from related fields to enhance the efficiency and effectiveness of pod-based rail systems.

The optimization of resource allocation/assignment in transportation systems has been widely studied, particularly in the context of ride-sharing, autonomous vehicles, railway logistics , car-pooling algorithms etc. While there is limited research addressing autonomous pod railway systems, existing studies on related topics provide valuable insights into the key factors, optimization methods, simulation and evaluation methods that can be adapted to address the challenges in this domain.

To clarify the relationship between the components of related problems (e.g., car/ride-sharing and railway logistics) and the elements of the autonomous pod-based rail system, Table 2.1 provides a mapping of these components.

Table 2.1: Mapping Components to Pod Context

Related Problem	**Component in Literature**	**Equivalent in Pod Context**
Car/Ride-Sharing	Car/Vehicle	**Carrier**
	User/Passenger	**Transport Unit (TU)**
	Station/Pickup-Drop-off	**Station**
Railway Logistics	Rolling Stock (Train Unit)	**Carrier**
	Passenger/Goods	**Transport Unit (TU)**
	Station/Depot	**Station**
Empty Rolling Stock	Empty Train Units	**Empty Carriers**

2.1. Factors influencing carrier-TU matching.

The matching of carriers to TUs is influenced by many factors, such as temporal constraints such as wait times, operational costs, the vessels being demanded, and the fleet size (which, in our case, refers to the number of carriers) that affect the matching of carriers to TUs [8], [9] . Research in the area of ride-sharing systems [10], [11], [12] has identified time windows, vehicle travel distances, and availability as one of the critical factors in optimizing matching.

Tafreshian et al. (2020) [10] did a comprehensive review of ride-matching algorithms in peer-to-peer (P2P) rideshare systems which they classify as one to one, one to many, and many to many matching. This discussion of temporal and spatial constraints aligns closely with the challenges in autonomous pod railway systems, where carriers must navigate on fixed railway networks and adhere to strict time windows. Wu et al. (2008) [11] presented decentralized P2P shared ride systems with an explicit need for spatial constraints and geospatial matching of riders and drivers. Ma et al. (2019) [13] introduced a new heuristic algorithm to solve the P2P ridesharing match problem with the new recursive techniques to pair riders on the based on feasibility constraints and preference lists.

In addition, this study highlights the employment of complex meta heuristics, nearest neighbour dispatch strategies, and insertion algorithms as provided by [9], with emphasis on the FPSO's (Firefly Particle Swarm Optimization) capacity to quickly converge for robust solution quality and efficient carrier distribution in preset scenarios with demand. Trip-vehicle assignment problems are in the focus of Bei and Zhang (2018) [12], where they develop a two-phase algorithm using minimum weight perfect matchings that guarantees no vehicle bears more than one request. This encapsulated approach achieves impressive minimizing of travel distances and operational expenses and provides a solution framework that can be adapted to the assignment of rail carriers to TUs, provided that each one can only move a single TU at a given timestamp. Developments in this work has revealed the importance of the strategic location design of the service area together with the composition of the vehicle fleet to not exceed the emission of CO2 and at the same time serve the customer satisfactorily [14].

In following up this conversation, the paper "A Fuzzy Approach to the Vehicle Assignment Problem" by Milosavljevic (1996) [15] is an example of how fuzzy logic is utilized within vehicle assignment decisions. This approach incorporates fuzzy set theory to manage the degree of unknowns and subjective parameters involved in vehicle assignment, which is highly useful in the autonomous pod railway systems. By simulating dispatcher logic and dealing with the fuzziness of operational constraints such as the number of vehicles and the timing of trips, this approach could considerably increase the efficiency of carrier-TU assignments. Fuzzy logic helps in addressing real world problems of matching carriers to TUs on the rails with the greatest possible economy within the given time and space parameters per-

taining to specialized resources.

2.2. Repositioning of empty carriers

When managing autonomous pod systems on rail, efficient relocation strategies prevail because they guarantee adequate availability of carriers, ensuring that the service coverage levels are consistently high while attempting to reduce the operational costs. Effective relocation not only increases the service level provided, but also assists in the distribution of the carriers across the network, which is essential in reducing the idle times and unnecessary transit.

In railway operation, optimizing rolling stock circulation is crucial for resource efficiency and demand fulfillment. Alfieri et al.'s [16] paper employs an integer multicommodity flow model with transition graphs to optimize train unit circulation, minimizing costs and maintaining operational compliance. Peeters and Kroon's [17] study enhances this by using a branch-and-price approach to efficiently allocate train units across lines, reducing seat shortages and unnecessary operations. Similarly, Canca et al.'s [18] work proposes a mixed integer programming model tailored for Rapid Transit Systems, focusing on minimizing train empty movements and equilibrating maintenance needs. Adapted to the context of this project, where the train units represent carriers and the demand comprises the TUs, these methodologies could optimize carrier assignments and relocations, ensuring timely and efficient transitions between road and rail, while facilitating effective platoon formations and carrier repositioning. This approach is ideal for managing the complexities of dynamic carrier relocation and operational planning in the rail-based autonomous pod system.

While railway operation literature provides valuable insights for managing rolling stock and optimizing schedules, it typically focuses on broad, network-wide optimizations and might overlook the detailed needs of individual pod assignments in autonomous systems. This oversight leads us to delve into car-sharing and ride-sharing literature. These areas excel in micro-managing resources and offer adaptable strategies for precise resource allocation, which are essential for meeting the specific operational challenges faced by autonomous rail-based systems. In transitioning to car-sharing and ride-matching literature, useful parallels are drawn: vehicles are analogous to carriers, and the riders' demand corresponds to the Transport Units (TUs).

A well-known example of such efforts is provided by (Illgen , Höck & Alfian) [19], [20], who in their systematic review on VReP (Vehicle Relocation Problem) in one way car sharing networks tried to capture the complexity of such logistic problems. Their research supports the effectiveness of mixed-integer programming for such relocation problems, in addition to simulation models, and multistage methods for relocation management. It was noted, however, that the best results in predictive remote vehicle relocation with the vehicle's historic location use were obtained out of the multistage methods.

Such methods are especially important for adjusting to the shifting requirements of a rail-based system where carriers have to be efficiently redistributed in light of real-time requirements and bounds. The review gives contributes towards the multistage approaches that combine optimization and simulation, presenting a framework for real time control and decision making that is necessary for the effective handling of carrier relocation in the autonomous pod transport systems (Illgen & Hock,2019)[19]. To build on this, Clemente et al. [21], further elaborate on the concept in the study "The Vehicle Relocation Problem in Car Sharing Systems: Modeling and Simulation in a Petri Net Framework", where user-based relocations strategies are modeled and simulated using Timed Petri Nets (TPN) in focusing on user motives and incentives for load balancing in car-sharing systems. While these those modalities were devised for dynamic demand, the methods are seamlessly applied for static, predetermined

demand in autonomous pod based rail systems through preset carrier station allocation and movement based on a control schedule. By TPN bounding the relocation paths and snapshot demand conditions, the model provides the desired optimal condition of carrier position and operational efficiency.

Scheduled incentives can be strategically used to manage carrier utilization, enhancing service reliability and efficiency in a rail context . This integration from car-sharing systems into rail transport demonstrates how tailored adaptations of dynamic models to static systems can significantly improve resource utilization and system performance,on a broader scope study is being done on automated systems to perform the rellocation activity [8]. Further building on these foundational strategies, the paper by Weikl, Bogenberger & Cepolina et al. (2012) investigates different relocation strategies for car sharing, specifically focusing on operator-based methods and a two-step algorithm that combines predictive offline planning with adaptive real-time adjustments [8], [22]. This method is particularly relevant for our project on autonomous pod-based rail systems, where similar strategies can be adapted to optimize the positioning of carriers. By applying a similar two-step approach, historical data can be utilised to proactively plan carrier deployment and incorporate real-time data to make ongoing adjustments. This guarantees that carriers are placed in strategic locations ahead of the anticipated fixed demand which improves system reliability and efficiency without needing any manual effort from users. This also highlights the carriers' management aspect that is necessary for strategic efficiency and the way in which the techniques devised for car sharing can be more easily applied to complex and larger transportation system railways. Algorithms like the greedy ,Tabu and Iterated Local Search Algorithm search have been have been analyzed by several researchers including Ait - Ouahmed, Josselin, Zhou, and Lai [23], [24] for their works studied in exploration of optimal vehicle distribution strategies for car-sharing systems. In their work, they developed and tested these algorithms to efficiently address vehicle imbalances across network stations. The greedy algorithm quickly identifies cost-effective moves by selecting the most immediate beneficial relocation based on current vehicle excess or deficiency, focusing on achieving short-term balance. In contrast, the Tabu search algorithm explores the solution space more thoroughly by accepting worst solutions to escape local optima, all while avoiding previously explored solutions through its Tabu list, leading to a potentially better long-term distribution solutions , whereas iterated local search algorithm is within a rolling-horizon framework for dynamically optimizing the carrier placements based on fixed schedules . In another study, Kek and Alfian [20], [25] proposed two critical techniques for moving cars in a car sharing system: The Shortest Time Technique, which prioritizes rapid relocation to reduce operational downtime, and the Inventory Balancing Technique, aimed at maintaining an optimal level of vehicle availability by redistributing vehicles based on station inventory levels. These strategies ensure efficient use of resources and enhance service availability across the network.The effects of the relocation time period, or when to move vehicles, were also examined by Ganjar Alfian et al. They found that periodical relocation—that is, relocation that occurs every six hours—had lower relocation costs than static relocation, which is the immediate relocation that occurs when a station net flow reaches or falls below a certain threshold [26].

In the matter of our project on carrier-TU assignment, these algorithms can be instrumental in managing the relocation of carriers to efficiently fulfill Transportation Unit (TU) matching. For example, a greedy algorithm could be used during periods of predictable/static demand to quickly allocate carriers to stations with imminent TU arrivals, ensuring rapid response and minimal waiting times. Meanwhile, the Tabu search could be employed for long-term strategic planning, optimizing the distribution of carriers across the network to anticipate and adjust to varying demands throughout the day or in response to special events.

Further details on the literature review are given in Table A.1 in Appendix Chapter A.

2.3. Literature gaps and contributions

Although previous studies on ride-sharing, car-sharing, and railway logistics have made valuable contributions, this research aims to cover new gaps that have emerged. The absence of autonomous pod-based research is mainly caused by the focus on existing road-based systems (e.g. ride-sharing and car-sharing) or traditional railway logistics. These systems raise novel issues like carrier-TU assignment and relocation in a hybrid road-rail scenario. This work aims at autonomous pod-based rail systems by creating optimization models and algorithms that consider the special conditions of these systems. This research integrates road and rail operations interaction which is often overlooked in literature. Also, while many studies address dynamic demand scenarios in a real-time environment like ride-sharing or car-sharing, there is little attention paid towards static demand situations within railway systems. Scheduled rail operations give rise to static demand, but the topic of empty carrier relocation to satisfy future demand is under-researched. Current strategies for relocation are largely user-incentive driven, which do not work for systems that move TUs.This project focuses on dynamic demand scenarios and develops system-driven relocation strategies that do not rely on user incentives, ensuring efficient resource utilization and high service reliability.

Moreover, although the effect of platooning (multiple pods traveling together) is multi-faceted for a lot of road-based systems, it is nearly non-existent for modular pod systems. Platooning has an impact on carrier availability and scheduling, but this has not been sufficiently tackled in the context of carrier TU assignment and relocation. Even though platooning is not the primary objective of this study, it is recognized as a tertiary component that could impact the carrier TU assignments. The developed models incorporate platooning in a way that guarantees carrier availabilities as well as scheduling is optimally set even when several pods move simultaneously.

To finish, the use of Rolling Horizon is not novel (supply chain management, automotive sharing), but its usage for carrier TU assignment and relocation in flexible modular pod rail systems is new. Problems with dynamic decision making combined with uncertainty and time sensitivity of the given task at hand can greatly benefit from the implementation of Rolling Horizon, yet this area remains undiscovered. This study will implement Rolling Horizon as a primary tool to tackle the carrier TU assignment and relocation problem. Rolling Horizon allows the user to continuously re-optimize by dividing the problem into smaller time windows, in doing so the system stays agile and efficient.This approach is integrated with MILP and DES, providing a robust framework for dynamic decision-making and evaluate the system behavior with different scenarios. Together, these contributions address the gaps in the literature and position this research as a significant advancement in the field of autonomous pod-based rail systems.

3

System Dynamics

This chapter will provide a comprehensive overview of the system's mechanics. First, the end-to-end flow of a transport request will be visually illustrated, both with and without the consideration of platooning, to provide a high-level understanding of the process. Following this, delving into the specifics of the state-event model systematically defines every possible operational state for both the Transport Units (TUs) and the Carrier Units (CUs). Finally, we will detail the discrete events that trigger transitions between these states, such as arrivals, departures, and loading operations. Together, these components create a complete and precise blueprint of the system's behaviour, which is essential for the implementation of the simulation model discussed in the subsequent chapters.

3.1. Simulation Flow Process

Figure 3.1 illustrates the TU-CU matching process without considering platooning. It captures the basic point-to-point transport of TUs, where each TU is loaded onto a CU at the pick-up station, travels solo through the network, and is finally unloaded at the destination. This approach assumes a direct, ungrouped transit flow, where each TU is managed individually without the efficiency benefits of platooning.

When a transport request (TU) enters the system, it first arrives at its designated pick-up station and remains there until a carrier unit (CU) becomes available. The CU then drives empty to that station (if it is not already positioned), waits for the TU to be ready, and initiates the loading operation. Once loading completes, the combined CU–TU pair departs immediately and traverses the network along the predefined route toward the delivery station. Travel occurs without interruption until the pair reaches the drop-off point, where the TU remains on board while unloading is performed. Upon completion of unloading, the TU exits the system, and the CU departs the station empty. The empty CU may then either proceed to the next pick-up station, reposition itself in anticipation of future requests, or enter maintenance if required. This basic sequence—arrival, wait, load, in-transit movement, unload, and empty departure—is shown in Figure 3.1 and captures the baseline operational logic without any coordinated grouping of CUs.

Figure 3.2 extends the baseline sequence by allowing carrier units to travel in platoons both when empty and when carrying a transport unit. After completing a loading or unloading operation, each CU checks whether it should form or join a platoon before its next departure. If platooning is desired (platoon stations are predetermined and known from the MILP output as explained in Chapter 4), a CU will either wait at its current station for other CUs to arrive or move to a predefined station. Once at least two CUs converge—whether they are empty or loaded—they merge into a single platoon, synchronise their departure time and speed, and traverse the network together along the same route. The platoon remains intact until individual CUs reach a station where one of two events occurs: (1) a loaded CU

arrives at its delivery station and unloads its TU, or (2) an empty CU arrives at a designated split station (either to pick up a TU or to further form a platoon). At that moment, the group dissolves: unloaded CUs become available for new tasks or maintenance, and loaded CUs (if still in transit) continue to their final delivery station alone. By enabling both empty and loaded platooning, this enhancement preserves the core steps of arrival, loading, transit, unloading, and empty movement while improving resource utilisation and travel efficiency.

Without Platooning:-

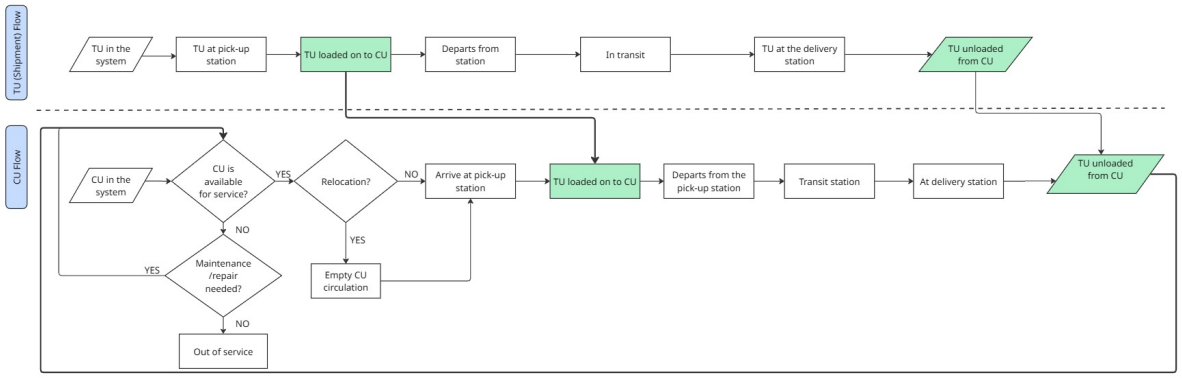

Figure 3.1: Simulation flow process

With Platooning:-

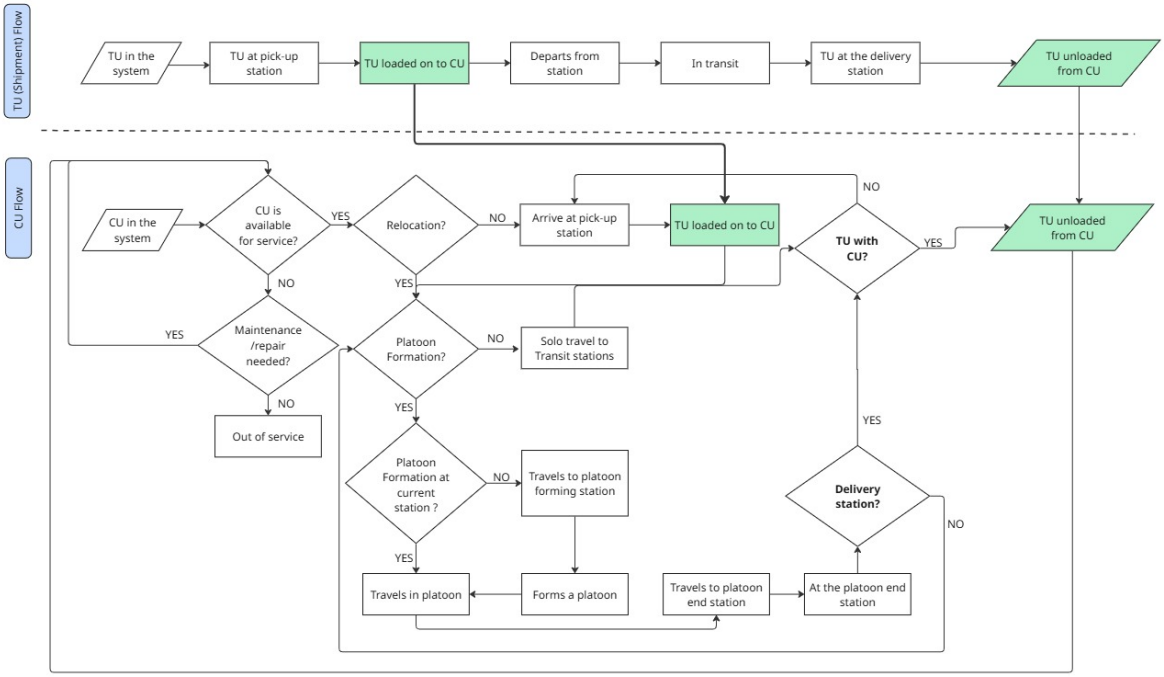

Figure 3.2: Simulation flow process with platooning

3.2. Introduction to State-Event Modelling in TU–CU Systems

This section introduces the core “states” and “events” that drive our discrete-event simulation of transport units (TUs) and carrier units (CUs). A state describes what a unit is doing at a given moment—for example, waiting at a station, carrying a load, or travelling between stops—while an event is the exact

instant that causes a unit to change from one state to another (e.g., a TU arriving at a platform, a CU beginning its journey, or two CUs joining a platoon). By enumerating every possible state for each TU and CU and specifying the events that trigger each transition, the simulation can step through the sequence of loadings, departures, platoon formations, arrivals, and unloadings in precise chronological order. This clear, event-driven structure makes it straightforward to record, trace, and analyse every action in the system.

A discrete-event system is one in which state changes occur instantaneously at specific points in time, with nothing happening between those events. This contrasts with continuous systems, where states evolve continuously. Systems fitting this description can be analysed using discrete-event simulation [27].

Global system state At any point in time, the simulator keeps a single "snapshot" of the whole system. This snapshot includes:

- **Transport Units (TUs):** for each TU—its current state (T0–T4), location (station or arc), assigned CU (if any), planned route/next stop, and time-window data.
- **Carrier Units (CUs):** for each CU—its current state (C0–C8), location, assigned TU (if any), current itinerary step, platoon membership (if any), and remaining processing/travel time.
- **Station queues and capacities:** who is waiting (TUs/CUs), which load/unload operations are in service, and current occupancy vs. capacity at each station.
- **Active platoons:** which CUs are grouped, where they are (arc/station), and their planned split/merge stations.
- **Network availability:** which arcs/resources are open or closed (e.g., due to disruptions) and the travel/handling times currently in force.
- **Lock-ins (uninterruptible tasks):** ongoing activities that must finish before re-planning can change them (e.g., an in-progress arc traversal, an active load/unload, or a reserved station slot).
- **Outstanding requests:** TUs that have been released but not yet delivered, plus any future releases with their release times.
- **Event calendar:** the next scheduled completions (arrivals, departures, load/unload completions, platoon join/split, etc.).

Events $E1$–$E10$ update this snapshot (see Tables 3.3 and 3.4); when an event fires, the relevant records change state and the event calendar is refreshed. During a disruption, the simulation is paused and this snapshot is passed to the MILP (TU/CU states and positions, network availability, lock-ins, outstanding requests). The MILP produces an updated plan, and the simulation resumes while respecting the lock-ins.

3.2.1. Transport Unit (TU) States

In a discrete-event simulation, *states* represent the various conditions or "modes" that an entity/actor can occupy as it moves through the system. For a Transport Unit (TU), we define five distinct states, labelled `T0` through `T4`. Below is a detailed explanation of each TU state.

Table 3.1: Transport unit (TU) states

State Code	State	Description
T0	At Station Waiting	TU is waiting at its pickup station for loading.
T1	Loaded to CU	TU has been placed on the CU and is ready to depart.

State Code	State	Description
T2	In Transit	TU is being carried by the CU toward its destination.
T3	At Delivery Station	TU has arrived at the delivery station, awaiting unload.
T4	Delivered	TU has been unloaded, and its journey is complete.

T0: At Station Waiting

A TU in state `T0` is idle at its pick-up station, ready to be loaded onto a Carrier Unit (CU). It has not yet been loaded to any carrier unit (CU); it is simply *waiting* for its turn. Key points about `T0`:

- The TU has arrived (e.g., by truck, or has been placed at the station) but has not yet been picked up by a CU.
- It remains in `T0` until a CU becomes available at the pick-up station and the loading process begins.
- In simulation, the moment the TU arrives at the station (often driven by a scheduled arrival time), it transitions directly into `T0`.
- While in `T0`, the TU may incur a waiting cost or simply occupy space, but it is not *moving*.

T1: Loaded to CU

Once a CU has been allocated and the physical loading has taken place, the TU moves into state `T1`. At this point, the transport unit (TU) has been placed onto the carrier unit (CU) and is ready for departure. Key aspects of `T1`:

- The TU is now *on board* its assigned CU, but the departure has not yet occurred. It is in a "loaded, but stationary" condition.
- In simulation terms, there is typically a small loading-time delay that must occur before the TU transitions from `T0` to `T1`.
- During `T1`, the TU is effectively *locked* to its CU: it cannot be re-assigned (unless a CU fails, which will be discussed under disruptions in chapter 4) or moved until departure.

T2: In Transit

When the TU-CU departs the pick-up station with its TU on board, the TU enters state `T2`. At this moment, the TU is in motion along its planned route toward its destination. Important characteristics of `T2`:

- The TU is moving from its pick-up station towards its delivery station as part of the CU's journey.
- In simulation, a travel-time delay is usually specified for this transit segment. Only after that delay elapses does the TU reach the next state.
- While in `T2`, the TU cannot be unloaded or rerouted; it is bound to the CU's path until arrival at the TU delivery station.
- If the CU joins a platoon, the TU remains in `T2`, possibly adjusting its estimated arrival time.

T3: At Delivery Station

Once the CU carrying the TU arrives at the TU's designated delivery station, the TU transitions to state `T3`. In this state, the TU is physically at the destination but has not yet been unloaded from the CU. Details of `T3`:

- The TU has completed its transit and is now ready for the final unloading process.
- There may be a short *unloading time* simulated while the forklift or other equipment detaches the TU from the CU.

- The TU remains in `T3` until that unload action completes.
- While in `T3`, the TU occupies space at the delivery station and may incur local handling costs or delays.

T4: Delivered

When the unloading process finishes, the TU enters state `T4`, meaning the TU has been fully delivered and its journey is complete. Highlights of `T4`:

- `T4` represents *completion*; no further simulation actions will involve this TU.
- From a modelling perspective, we can record statistics (e.g., total transit time, waiting time) once the TU reaches `T4`.
- In this DES framework, a TU in `T4` is effectively removed from the active simulation, freeing up resources.
- Although the TU is "finished," its delivery may trigger downstream events in a larger supply chain (e.g., starting a next-mile delivery), but that is beyond our current scope.

3.2.2. Carrier Unit (CU) States

A Carrier Unit (CU) in this discrete-event simulation can occupy one of nine distinct states, labelled `C0` through `C8`. Each state corresponds to a specific operational condition of the carrier unit (CU), from being idle or carrying a Transport Unit (TU) to repositioning or undergoing maintenance. Below is a detailed description of each CU state.

Table 3.2: Carrier unit (CU) states

State Code	State	Description
C0	Idle and Unassigned	CU is free and awaiting assignment or repositioning.
C1	TU Loaded to CU	CU has a TU on board and is ready to depart.
C2	In Transit – Solo	CU is carrying a TU alone, without platoon partners.
C3	In Transit – Platoon	CU is carrying a TU as part of a platoon of CUs.
C4	At Delivery Station	CU has reached the TU's delivery station and awaits unloading.
C5	At Station, Waiting	CU is idle at a station, ready for pickup or platoon formation.
C6	Repositioning – Solo	CU is moving empty to another station on its own.
C7	Repositioning – Platoon	CU is moving empty as part of an empty-CU platoon.
C8	Out of Service	CU is unavailable (maintenance or downtime).

In the simulation model.

C0: Idle and Unassigned

- CU is free and not assigned to any tasks and is available for new tasks (pickup or empty repositioning).
- Remains idle until a scheduling decision allocates it to a TU or a repositioning order.

C1: TU Loaded to CU

- CU has just completed loading a TU and is ready to depart.
- Stationary at the station, with the TU on board.
- Dedicated exclusively to that TU; cannot perform other actions until departure.
- Awaits the departure event (E5) to transition into transit.

C2: In Transit – Solo

- CU is transporting a TU alone, covering each arc's specified travel time.
- Moves under independent control (no platoon partners).
- Remains in solo transit until either:
  - It reaches the TU's delivery station (then E8/E9), or
  - It merges into a platoon at the next station (E6).

C3: In Transit – Platoon

- CU carries a TU as part of a platoon (two or more CUs travelling together).
- Shares speed, routing, and departure timing with platoon partners.
- Cannot separate until a platoon-split event (E7) occurs at a station.
- Remains platooned until either dissolution or arrival at the TU's delivery station.

C4: At Delivery Station

- CU (with its TU on board) has reached the designated delivery station.
- Parked at the platform; TU awaiting the unloading process.
- In a transient "arrival" condition, not yet unloaded.
- Awaits the unload event (E9) to transition into station waiting.

C5: At Station, Waiting

- CU is idle at a station, either just after unloading or after empty arrival.
- Occupies one of the station's parking slots.
- Waits for its next task: loading a TU or empty repositioning.

C6: Repositioning – Solo

- CU travels empty from one station to another, following a planned route.
- Moves under independent control with no TU on board.
- Covers each arc's specified travel time, then transitions to `C5`.

C7: Repositioning – Platoon

- CU travels empty as part of a platoon of empty CUs.
- Remains in platoon until:
  - The platoon splits (E7), reverting this CU to solo repositioning or waiting.

C8: Out of Service

- CU is undergoing maintenance or otherwise unavailable.
- Cannot carry TUs, move, or accept assignments.
- Remains inactive until maintenance completes, then transitions to `C0`.

By defining these nine CU states C0 through C8, we capture all possible conditions of a Carrier Unit within the simulation. Each state encapsulates a clear operational role, from waiting for work (C0 or C5), to carrying TUs alone or in platoons (C1, C2, C3, C4), to moving empty (C6, C7), or being offline for maintenance (C8). The transitions between these states will be governed by discrete events, which are discussed in the following section.

In summary, these nine CU states (C0–C8) capture every phase of the carrier's lifecycle: from idle availability and loading, through solo or platoon transit with or without a TU, to empty repositioning and maintenance. Transitions between these states are driven by discrete events (e.g., loading, departure, arrival, platoon join/split), ensuring that the CU's behaviour is fully captured.

3.3. System events and transitions

Events are a crucial component of Discrete Event Simulation (DES), representing the discrete, instantaneous occurrences that drive state changes within a system, as shown in the transition diagrams in 3.3 and 3.4. In DES, events mark the precise moments when system elements transition from one state to another, such as the arrival of a TU at a station, the loading of a TU onto a CU, or the departure of a CU from a terminal. These events capture the cause-and-effect relationships that define the operational flow of a complex system. The events and the event-state transition are shown in table 3.3 and 3.4, respectively.

Table 3.3: List of Events in the TU–CU System

Event Code	Event	Description
E1	TU arrives at station	Transport Unit appears at its pickup station and begins waiting for a CU.
E2	CU becomes available	Carrier Unit completes maintenance or downtime and enters the idle pool.
E3	CU arrives at a station	Carrier Unit (empty or loaded) pulls into a station and transitions to waiting.
E4	TU is loaded onto a CU	Loading completes: TU moves on board, CU is ready to depart.
E5	TU–CU depart pickup station	Loaded CU (solo or platoon) leaves the pickup station with its TU.
E6	CU joins platoon	A solo CU merges into an existing platoon at a station.
E7	CU leaves platoon	A CU splits off from a platoon at a station, either to deliver or reposition solo.
E8	TU–CU arrive at delivery station	Loaded CU completes its final link and reaches the dropoff station of TU.
E9	TU unloaded from CU	Unloading completes: TU exits the CU, which becomes idle at that station.
E10	CU departs station empty	Empty CU (solo or platoon) departs a station to reposition elsewhere.

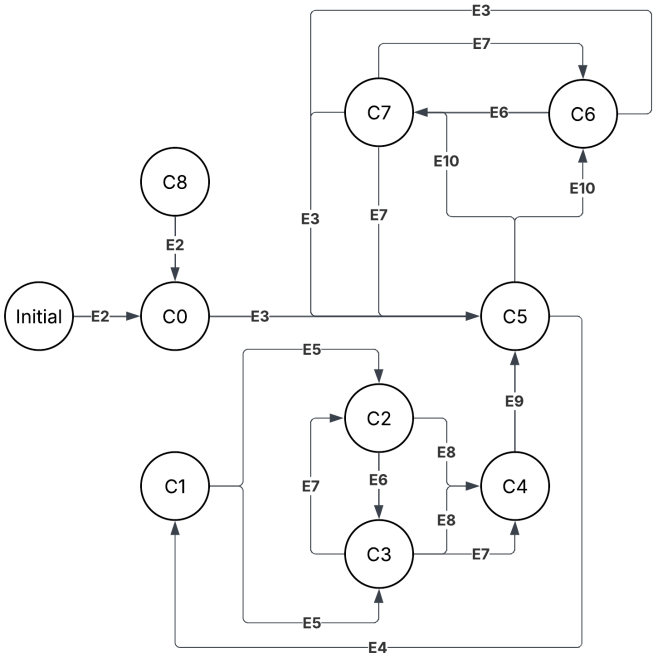

Figure 3.3: Carrier Unit (CU) Event-State Transition Diagram

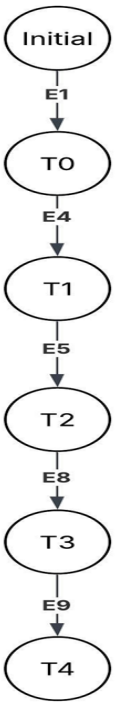

Figure 3.4: Transport Unit (TU) Event-State Transition Diagram

Table 3.4: Event–State Transition Matrix

Event	Event Name	Current State(s)	Next State(s)
E1	TU arrives at station	– (initial)	T0: At Station Waiting
E2	CU becomes available	– (initial) C8: Out of Service	C0: Idle and Unassigned
E3	CU arrives at station	C0: Idle and Unassigned	C5: At Station, Waiting
		C6: Repositioning – Solo	C5: At Station, Waiting
		C7: Repositioning – Platoon	C5: At Station, Waiting
E4	TU is being loaded onto a CU	T0: At Station Waiting C5: At Station, Waiting	T1: Loaded to CU C1: TU Loaded to CU
E5	TU–CU depart pickup station	T1: Loaded to CU C1: TU Loaded to CU	T2: In Transit C2: In Transit – Solo C3: In Transit – Platoon
E6	CU joins platoon	C2: In Transit – Solo C6: Repositioning – Solo	C3: In Transit – Platoon C7: Repositioning – Platoon
E7	CU leaves platoon	C3: In Transit – Platoon	C2: In Transit – Solo T3: At Delivery Station C4: At Delivery Station
		C7: Repositioning – Platoon	C6: Repositioning – Solo C5: At Station, Waiting
E8	TU–CU arrive delivery station	T2: In Transit C2: In Transit – Solo C3: In Transit – Platoon	T3: At Delivery Station C4: At Delivery Station
E9	TU is being unloaded from CU	T3: At Delivery Station C4: At Delivery Station	T4: Delivered C5: At Station, Waiting
E10	CU departs station empty	C5: At Station, Waiting	C6: Repositioning – Solo C7: Repositioning – Platoon

E1: TU Arrives at Station

Event `E1` represents the moment when a Transport Unit (TU) first appears at its pickup station. When `E1` occurs, the TU transitions from "not yet in the system" into the state `T0`, meaning "at station, waiting." No movement happens at `E1` itself it simply records that the TU has shown up and is now ready to be matched with a Carrier Unit (CU). In simulation, `E1` often happens at a pre-scheduled arrival time generated by the input data.

E2: CU Becomes Available

Event `E2` marks the instant when a Carrier Unit (CU) finishes any maintenance or downtime and becomes free to start a new task. Before this event, the CU may have been offline, undergoing repairs, or otherwise occupied. When `E2` triggers, the CU moves from "out of service" into an "idle and unassigned" condition. Conceptually, `E2` models activities such as completing a maintenance check, or finishing paperwork—any action that renders the CU ready to work. After `E2`, the CU can be assigned

to pick up a TU or to reposition itself to another station.

E3: CU Arrives at Station

Event E3 captures a moment when a CU (empty CU) pulls into a station. This can happen under three scenarios:

1. A CU that was idle (C0) moves to its first station at the start of the simulation or after becoming available.
2. A CU that has been repositioning alone (C6) finally reaches a station and is now waiting for its next assignment.
3. A group of empty CUs travelling together as a platoon (C7) pulls into a station, ready to split up or wait for further instructions.

When E3 occurs, the CU enters the "at station, waiting" state (C5). Although E3 happens at an instant in simulation time, it implies that the CU has spent the incurred travel time en route. Immediately after E3, the CU can either load a TU (if one is present) or prepare for another action.

E4: TU is being loaded onto a CU

Event E4 represents the instant when a CU and TU complete their loading process and become coupled. Prior to E4, the TU must be in the "at station, waiting" state (T0) and the CU must be in the "at station, waiting" state (C5). The loading process typically takes some measurable amount of time (e.g., forklift operations, securing straps, paperwork). In discrete-event simulation, that loading duration is modelled as a delay, and when that delay ends, E4 fires. As a result, the TU moves into T1 ("loaded to CU") and the CU moves into C1 ("TU loaded to CU"). No actual movement begins at E4; instead, it signifies that the handling and securing steps are complete.

E5: TU–CU Depart Pickup Station

Event E5 occurs at the exact moment the CU, now carrying its TU, begins its journey out of the pickup station. The TU transitions from T1 ("loaded to CU") into T2 ("in transit"), and the CU transitions from C1 ("TU loaded to CU") into either C2 ("in transit – solo") or C3 ("in transit – platoon"), depending on whether the CU travels alone or joins a platoon immediately. In either case, the departure step is instantaneous in simulation; the travel time along the route is represented by a separate delay.

E6: CU Joins Platoon

Event E6 designates the moment when a CU travelling alone (in state C2) or repositioning alone (in state C6) merges into an existing platoon at a station. Imagine two or more CUs aligning their departure times so they can travel together; that grouping is the platoon. At the station where the platoon forms, E6 sets each joining CU's state as "in transit – platoon" (C3) or "repositioning – platoon" (C7). While the platoon travels as a group, the individual CUs share the same speed and routing instructions. E6 is instantaneous, but its precondition is that the CUs all meet at the same station and are scheduled to travel to the same next destination.

E7: CU Leaves Platoon

Event E7 marks the instant when a CU splits off from a platoon at a station. Two states trigger E7:

- A CU traveling in a loaded platoon (C3) arrives at a station and separates In that case, the CU transitions into C4 ("at delivery station") while the TU moves into T3 ("at delivery station") if that station is the TU delivery station, otherwise the CU can continue a solo travel to the next station.
- A CU travelling in an empty platoon (C7) arrives at a station where it does not need to continue in that group. It can either remain at that station (C5) or continue solo repositioning (C6).

Because platoon splits only occur at stations, E7 does not represent travel time but a decision point that changes the CU's grouping. After E7, the CU either proceeds alone or waits for its next assignment.

E8: TU–CU Arrive Delivery Station

Event E8 indicates that a CU, with its TU on board, has completed its final transit link and physically reaches the TU's designated delivery station. The TU changes from T2 ("in transit") to T3 ("at delivery station"), and the CU moves from either C2 ("in transit – solo") or C3 ("in transit – platoon") into C4 ("at delivery station"). Although the transit time was consumed as a delay on the link, E8 itself is the instantaneous moment of arrival. No unloading has occurred yet, but the TU and CU are now co-located at the delivery platform.

E9: TU is being unloaded from CU

Event E9 captures the precise instant when the TU is unloaded at the delivery station. Prior to E9, the TU is in T3 ("at delivery station") and the CU is in C4 ("at delivery station"). The unloading action typically takes a finite amount of time (for example, forklift operations, paperwork). When that unloading delay concludes, E9 fires and places the TU into T4 ("delivered"), while the CU transitions to C5 ("at station, waiting"). In other words, E9 signifies completion of the delivery cycle and frees the CU for its next task.

E10: CU Departs Station Empty

Event E10 represents the moment when an empty CU (in state C5) leaves a station to reposition to another station without carrying a TU. Upon occurrence, the CU moves into either C6 ("repositioning – solo") or C7 ("repositioning – platoon"), depending on whether it travels alone or as part of an empty platoon. Again, the actual travel time is modeled as a subsequent delay; E10 itself is the instantaneous trigger indicating "ready-to-move" for an empty departure.

4

Methodology

This chapter presents the translation of the rail-based transport system's mathematical models and simulation into executable code. Although the core mixed-integer linear programme (MILP) is solved externally (further explained in section A.3), its solution is utilised as input to a discrete-event simulation (DES) designed to evaluate operational performance and to explore "what-if" scenarios—most notably the effects of vehicle platooning and random disruptions. An overview of the MILP model is provided in section A.3, where the key inputs and outputs of the optimisation model are explained.

In Section 4.1, a small "toy" network and the Randstad Case Study are introduced as the test environments. The DES architecture is then described: the principal classes (`TransportUnit` and `CarrierUnit`), the finite-state logic governing loading, departure, platoon formation and dissolution, and unloading events, together with the mechanisms for recording a timestamped trace of each state transition. A pseudocode overview is provided first, followed by a detailed discussion of the actual implementation.

Once the core DES framework has been established, methods for interpreting its outputs—time-indexed tables of TU and CU states—are examined in order to compute key performance metrics such as travel times, waiting times, and platoon utilisation. The chapter concludes with an extension of the simulation to incorporate real-world uncertainties: the introduction of random delays, cancellations, and equipment failures enables analysis of system resilience and robustness under disturbance.

4.1. Case Studies

This section lays out the experimental setting. We first introduce a minimal *Toy Case* used to verify model logic and analyse behaviour under controlled disruptions. We then describe the *Randstad* network used to test realism and scalability, including stations, initial fleet placement, and the baseline demand set. For both cases we define the evaluation protocol (baseline → disrupted → re-optimised), the disruption types and timing, and the KPI pack used to compare outcomes.

4.1.1. Toy Case

The MILP model, as explained in section A.3, receives as input a small "toy" rail network, carrier initial positions, and a set of transport requests. Table 4.1 summarises the network topology, station capacities, and carrier fleet; Table 4.2 lists each transport request.

Table 4.1: Rail network and carrier data

Stations	$\{1, 2, 3, 4\}$, each capacity = 2
Travel times (min)	$(1 \rightarrow 2,\ 4)$, $(1 \rightarrow 3,\ 3)$, $(1 \rightarrow 4,\ 5)$, $(2 \rightarrow 4,\ 3)$, $(3 \rightarrow 4,\ 4)$
Carriers (CUs)	IDs = {1,2,3}
Initial locations	CU 1@Station 1, CU 2@Station 2, CU 3@Station 3

Table 4.2: Transport requests

Req. ID	Origin	Dest.	Pickup window	Delivery window
1	4	2	[11,18]	[14,21]
2	1	3	[12,18]	[15,21]
3	3	2	[14,19]	[21,2]
4	1	4	[1,5]	[8,13]
5	2	4	[1,7]	[5,11]

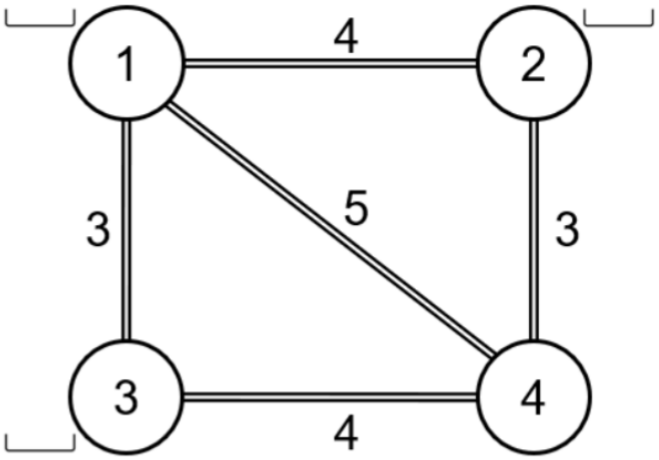

Figure 4.1: Toy network

Description: A four-station network (1–4) is considered, as shown in Figure 4.1, where each station has a capacity of up to two carrier units (CUs). Travel times between connected stations are expressed in minutes, and arcs are bidirectional. Three CUs are initially located at Stations 1, 2, and 3, respectively. Five transport requests are required to be satisfied within specified pickup and delivery windows, subject to the availability of carrier units in the system.

4.1.2. Randstad Case Study

The railway network employed for this analysis is represented by the existing system in the Randstad conurbation, Netherlands (see Figure 4.2). The key characteristics of the model are outlined as follows:

- **Geographical Scope:** The network is defined over an area of approximately 72 km by 58 km, bounded by the main stations of Den Haag (1), Amersfoort (10), Rotterdam (2), and Amsterdam (13).
- **Network Topology:** Both principal and secondary lines and stations are incorporated, resulting in a branched network structure.
- **Demand Nodes:** Nine major stations (1, 2, 3, 6, 7, 8, 9, 10, and 13) are designated as "transfer stations," representing the locations where transport demand originates and terminates.

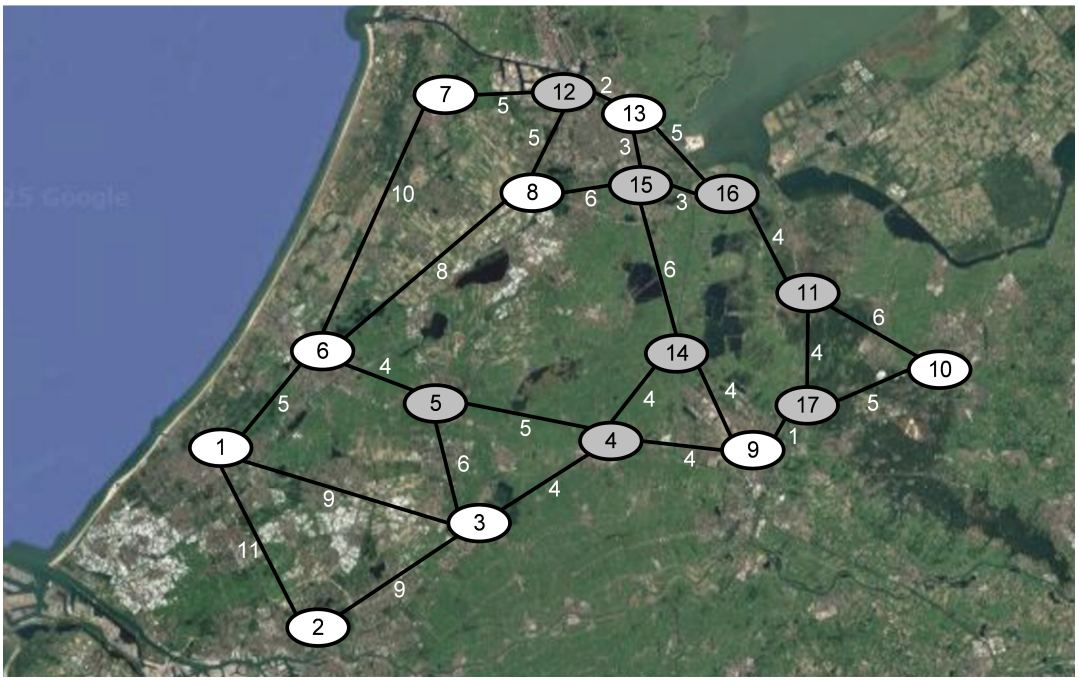

Figure 4.2: Case study network based on the Randstad area in the Netherlands with 17 stations (white: main, grey: routing) connected by arcs with weights representing the 'travelling time units'.

Table 4.3: Stations in Randstad Case Study: mode transfer stations (left) and other stations (right).

Mode transfer stations		**Other stations**	
Number	Name	Number	Name
1	Den Haag Central	4	Woerden
2	Rotterdam Central	5	Alphen aan den Rijn
3	Gouda	11	Hilversum
6	Leiden Central	12	Amsterdam Sloterdijk
7	Haarlem	14	Breukelen
8	Schiphol Airport	15	Duivendrecht
9	Utrecht Central	16	Weesp
10	Amersfoort Central	17	Utrecht Overvecht
13	Amsterdam Central		

Example scenario.
Considering ten carriers in the Randstad Case Study, the following initial carrier locations are obtained for a specific run, see Table 4.4.

Table 4.4: Initial locations of the carriers for the example scenario in the Randstad Case Study (generated with random seed 10).

Carrier	1	2	3	4	5	6	7	8	9	10
Station	1	9	10	1	6	10	7	3	13	8

Over a time period of 40 time units ($T = 40$), a total of 19 transport requests are generated. Their details are provided in Table 4.5.

Table 4.5: Transport requests for the example scenario in the Randstad Case Study (random seed 10 with T = 40).

Request	Pickup station	Delivery station	Pickup time window	Delivery time window
1	8	6	[13, 17]	[23, 29]
2	8	7	[13, 20]	[25, 34]
3	7	13	[5, 12]	[14, 22]
5	6	8	[19, 26]	[29, 38]
6	1	3	[7, 11]	[18, 24]
7	6	7	[17, 23]	[29, 37]
8	6	8	[21, 27]	[31, 39]
9	9	8	[2, 9]	[20, 31]
10	13	7	[7, 12]	[16, 22]
12	7	6	[17, 21]	[29, 35]
13	3	6	[18, 25]	[30, 39]
14	3	9	[24, 28]	[34, 40]
16	8	13	[13, 20]	[22, 30]
17	6	3	[4, 10]	[14, 20]
19	3	6	[11, 18]	[23, 32]

4.2. Discrete Event Simulation

This section explains how the MILP plan is executed in the discrete-event simulator and how re-optimisation is triggered. We map MILP outputs into simulator inputs (*tu_data* for transport requests and *cu_data* for carrier itineraries), outline the TU/CU processes and event logging (E1–E10), and specify the lock-in rules that protect in-progress actions during re-planning. We conclude with the rolling loop (simulate → detect disruption → re-optimise → resume) that underpins all experiments.

4.2.1. Inputs from MILP

Once the MILP returns a solution, the decision-variable values are converted into the two core inputs for the discrete-event simulation:

1. **Transport-Unit Table (`tu_data`):** For each request r with the binary variable $y_r = 1$, the fol-

lowing variables is extracted:

- *pickup time* p_r and *delivery time* d_r obtained from the continuous variables,
- the *assigned carrier*, identified as the c for which $x_{r,c} = 1$,
- the *route* node-sequence (e.g. [3,4,2]) specified in the input data,
- the *origin* (first node) and *destination* (last node) of the route,
- fixed *load* and *unload* durations, and
- the *total travel time*, computed as the sum of link travel times along the route.

2. **Carrier-Unit Itinerary (`cu_data`):** For each carrier c, the relocation-and-transport arc variables $zz[c, s_1, s_2, t]$ is scanned, and every $(s_1 \to s_2, t)$ with $zz = 1$ is collected. Sorting by departure time t produced the CU's time-sequence of moves. Together with the *initial station*, this sequence constituted the CU-itinerary employed in the DES.

To make this concrete, Table 4.6 and Table 4.7 summarise the two resulting data structures.

Table 4.6: Fields in the Transport-Unit table (`tu_data`)

Field	Type	Description
`tu_id`	integer	Unique request identifier
`pickup_time`	integer (time)	Scheduled departure time from origin station
`delivery_time`	integer (time)	Scheduled arrival time at destination station
`assigned_carrier`	integer	Carrier unit assigned to this transport
`route`	list of ints	Sequence of station IDs (e.g. [3,4,2])
`origin_station`	integer	First station on the route
`destination_station`	integer	Last station on the route
`total_travel_time`	integer	Sum of link travel times along `route`
`load_time`	integer	Fixed loading duration
`unload_time`	integer	Fixed unloading duration

Table 4.7: Fields in the Carrier-Unit table (`cu_data`)

Field	Type	Description
`cu_id`	integer	Unique carrier identifier.
`initial_location`	integer	Station ID where the CU starts.
`itinerary`	list of dicts	Ordered list of arcs; each arc-dict contains the fields `from_station`, `to_station`, `start_time` and `travel_time`.

With these two tables available, the DES engine is able to generate one process per TU (capturing pickup, loading, transit, and unloading events) and one process per CU (capturing relocation, transport, platooning, and drop-off events). In the subsequent section, the implementation of this event logic is presented in pseudocode.

4.2.2. Simulation Algorithm

In this section, the structure of the discrete-event simulation (DES) is outlined. A pseudocode sketch of the two core processes (one per transport unit and one per carrier unit) is first provided.

Walkthrough of Algorithm 1: The discrete-event simulation (DES) is initiated by creating separate processes for all transport units (TUs) and carrier units (CUs). Each TU is advanced along a straightforward timeline: arrival at the pickup station, loading, departure, travel to the delivery station, and unloading. In parallel, CUs are progressed through their itineraries arc by arc: before departure, an assigned TU and possible platoon membership are checked, determining whether departure occurs loaded or empty, and solo or in platoon. During travel, intermediate and arrival events are logged, unloading is performed at delivery stations, or repositioning is continued. The simulation proceeds event by event until all TUs and CUs have completed their processes, and a system-wide event trace is produced as output.

Algorithm 1 Overview of DES logic

Require: `tu_data`: list of transport-unit records
Require: `cu_data`: list of carrier-unit records
Require: `platoon_arcs`: set of $(s_1,\ s_2,\ t_0)$ where platooning occurs
Ensure: `trace` $\leftarrow$ []

procedure Main
  for all $tu \in$ `tu_data` **do**
    spawn TU_Process(tu)
  end for
  for all $cu \in$ `cu_data` **do**
    spawn CU_Process(cu)
  end for
  SimulateUntilAllDone
  return `trace`
end procedure

function `TU_Process`(`tu`)
  wait until `tu.pickup_time`
  log E1: "TU arrives at station"
  wait `tu.load_time`
  log E4: "load"
  log E5: "depart loaded"
  wait `tu.delivery_time - tu.pickup_time`
  log E8: "arrive delivery"
  wait `tu.unload_time`
  log E9: "unload"
end function

function `CU_Process`(`cu`)
  Initialise CU state
  log E2: "CU becomes available"
  log E3: "CU arrives at start"
  for all `arc` $\in$ `cu.itinerary` **do**
    $(s_1,\ s_2,\ t_0,\ dur) \leftarrow$ (from, to, start_time, travel_time)
    assignedTU $\leftarrow$ (is there a TU on $(s_1 \rightarrow s_2,\ t_0)$?)
    inPlatoon $\leftarrow (s_1,\ s_2,\ t_0) \in$ `platoon_arcs`
    wait until t_0
    if assignedTU **then**
      log E4: "load TU"
      log E5: "depart loaded" (solo or platoon if inPlatoon)
    else if not assignedTU **then**
      log E10: "depart empty" (solo or platoon if inPlatoon)
    end if
    wait dur
    if assignedTU and this is the TU's drop-off arc **then**
      log E8: "arrive delivery"
      log E9: "unload"
    else if inPlatoon and next arc not in platoon_arcs **then**
      log E7: "leave platoon" (solo or at station)
    else
      log E3: "arrive at station" (repositioning)
    end if
  end for
end function

Detailed Description

State-Event Tables. Both Transport Units (TUs) and Carrier Units (CUs) are represented as small finite-state machines. Each possible event (e.g., TU arrival, CU departure loaded, CU joining a platoon) triggers a deterministic state transition. Table 3.4 (already presented in Chapter 3) enumerates all states and events together with their codes. In the implementation, these event–state transitions are stored in the dictionaries `TU_TRANSITIONS` and `CU_TRANSITIONS`.

The `handle_event()` Routine: Advancement of a unit from one logical state to the next is managed through the function

```
handle_event(env, unit, event_code, trace, …)
```

This single helper function performs the following tasks:

- the unit's prior state is retrieved,
- the new state is determined via the appropriate transition map,
- the unit's current `state` and `location` are updated, and
- a timestamped record is appended to the global `trace` list.

Transport-Unit Processes: Each TU is progressed through the following sequence:

1. *Arrival* occurs at the pickup station at time `pickup_time` (event E1).
2. A fixed `load_time` is then observed, after which the *load* event (E4) is logged and immediate departure loaded (E5) takes place.
3. *Travel* continues until the `delivery_time`, followed by logging of the *arrival* event (E8) and execution of *unloading* (E9).

Carrier-Unit Processes: Each CU is advanced along its `itinerary`, defined as a time-ordered list of station-to-station arcs. For each arc:

- The MILP-derived decision variables **z** and **zz** are examined to determine whether the arc corresponds to a TU transport or an empty repositioning.
- It is also verified whether the arc forms part of a platoon (i.e., when two or more CUs share the same arc at the same time).
- At the scheduled start time of the arc, either
  - E4+E5 (*load TU* $\rightarrow$ *depart loaded*) is logged if a TU has been assigned, or
  - E10 (*depart empty*) is logged otherwise,

  with the transition tagged as *platoon* or *solo*.
- After the specified `travel_time`, one of the following is logged:
  - E8+E9 if the arc corresponds to a TU drop-off,
  - E7 if the CU leaves a platoon mid-route and continues solo, or
  - E3 to record a pure repositioning arrival.

Collecting the Trace: All calls to `handle_event` result in the appending of a tuple

(time, TU_id or empty, CU_id, location, old state, event, new state, duration, arc)

to the global `trace` list. This list is subsequently exported into a pandas DataFrame and written to Excel for downstream analysis.

4.3. Disruption Implementation

Traditional planning in freight transport is typically undertaken within a static environment using Mixed Integer Linear Programming (MILP). Although MILP is capable of producing globally optimal solutions when all parameters are known in advance, it exhibits limitations in accommodating uncertainty, unforeseen events, and sudden variations in demand. Real-world operations are, however, rarely static: carrier breakdowns, arc closures, and ad hoc transport requests occur unpredictably, and static MILP solutions rapidly become infeasible without complete re-formulation.

To overcome this limitation, MILP is coupled with a Discrete-Event Simulation (DES) framework. Within this framework, the DES continuously monitors the system state, recording carrier availability, the progress of ongoing trips, and the consequences of disruptions. When an event arises, the DES generates a system snapshot that is used as input for re-optimisation. Ongoing trips are thereby "locked in" and permitted to complete, while subsequent tasks are re-planned under the updated conditions. This hybrid DES–MILP approach introduces the flexibility required to adapt dynamically to uncertainty and to perform on-demand re-optimisation in near real-time. The disruption scenarios employed for this purpose are summarised in Table 4.8.

Disruption Type	Description	Simulated Case
CU Breakdown	CU breakdown during loading	Toy Case, Randstad
TU Breakdown	TU breakdown during loading	Toy Case
CU Delay	CU late arrival	Toy Case
TU Delay	TU late arrival	Toy Case
TU Addition	Request addition at different times	Randstad
Arc Removal	Travel route closed for maintenance	Randstad

Table 4.8: Disruption Types explored in the study

The analysis commences with a simplified Toy Case, which is employed to demonstrate the fundamental mechanics of disruption–response prior to the introduction of strategic, probabilistic disruptions. The framework is subsequently applied to the large-scale Randstad Case Study, where it is subjected to a set of disruptions designed to emulate complex, cumulative operational challenges. The principal objective of this chapter is to provide a transparent account of how disruptions are modelled, how the system is adapted through DES–MILP re-optimisation, and to establish the foundation for the performance analysis presented in the subsequent chapter.

4.3.1. Toy Case

In this section, the resilience of the system and the effectiveness of dynamic re-planning are assessed using a simplified Toy Case Study. The methodology is structured in three stages. First, a baseline operational plan is generated by solving the MILP for a fixed set of requests, representing a nominal disruption-free scenario.

Second, this baseline plan is executed within the Discrete-Event Simulation (DES), during which specific pre-defined disruptions are introduced. These disruptions are formulated to invalidate portions of the initial plan, thereby creating a disrupted scenario and enabling the measurement of their immediate impact on system performance.

Third, when a disruption results in plan failure, the simulation is paused. The current state of the system—including the locations and availability of all Carrier Units (CUs) and the status of all uncompleted requests—is then passed back into the MILP. The model is re-optimised to generate a recovery plan for the remainder of the operational period. This procedure enables direct comparison of key per-

formance indicators (KPIs) across the nominal, disrupted, and re-optimised scenarios.

Toy Case Disruption Scenarios
In this section, a probabilistic approach to modelling disruptions is adopted. Rather than assuming that a disruption will occur with certainty at a specific time, disruptions are defined on the basis of statistical likelihoods. This method employs what may be described as "synthetic data" to replicate the uncertainty and variability characteristic of real-world operations.

Table 4.9: Disruption scenarios employed in the Toy Case for probabilistic simulation across 200 runs.

ID	Disruption Type	Affected Unit	Event	Probability (%)	Delay Range
D_TU_001	DELAY	TU	E1	10	[2, 8]
D_CU_002	DELAY	CU	E3	5	[2, 8]
D_TU_BRK_1	BREAKDOWN	TU	E4	20	[100, 150]
D_CU_BRK_1	BREAKDOWN	CU	E4	10	[0, 0]

Each row in Table 4.9 defines a unique type of disruption that can be injected into the simulation. The columns collectively determine the behaviour of these events:

- **Disruption ID and Type:** Used to identify and categorise each failure as either a 'DELAY' or a more severe 'BREAKDOWN'.
- **Affected Event and Unit:** Specify the conditions under which a disruption may occur (the event code) and the entity that is impacted (either 'CU' or 'TU').
- **Probability %:** Defines the likelihood that a disruption will occur whenever its trigger event takes place, thereby introducing stochasticity into the simulation.
- **Delay Range:** If a disruption occurs, the delay duration is drawn randomly from within this range. The disruption is applied only for the selected time value.
- **Specific CU or TU ID:** An optional parameter that may be used to target a disruption to a single vehicle or shipment. This parameter is primarily employed in the Randstad Case Study.

By combining these parameters, the framework enables a flexible and robust representation of operational failures, ranging from frequent minor delays to rare but critical breakdowns. This configuration forms the basis for a comprehensive stress test of the system's resilience. Furthermore, by employing a probabilistic model, the DES can be executed across multiple iterations, each yielding a distinct sequence of disruptions. In this way, system performance is evaluated not against a single pre-determined failure, but across a spectrum of possible operational realities.

4.3.2. Randstad Case Study

For the more complex Randstad case study, a sequential, cumulative disruption methodology was employed to test the system's adaptive re-planning capabilities. The experiment begins with a baseline plan, generated by the MILP for a pre-defined set of requests. The DES then simulates this plan, but unlike the Toy Case, it is subjected to a series of cascading disruptions.

Disruption Types and Lock-in Mechanisms: The system handles three distinct types of disruptions, each with specific lock-in and re-optimisation strategies:

1. **Transport Unit (TU) Addition Disruptions:** The system accommodates dynamic request arrivals at various time points during simulation, including immediate additions at simulation start

and mid-simulation arrivals. When new requests arrive, the system immediately triggers re-optimisation to integrate these demands into the existing plan. The lock-in mechanism preserves all ongoing carrier operations, ensuring that carriers currently executing tasks continue their planned routes while the MILP re-optimises the remaining unassigned requests and newly arrived demands. This approach maintains operational continuity while accommodating dynamic demand changes. The system also supports time bracket analysis, where requests are added at different simulation times to evaluate the impact of timing on system performance, and time window flexibility analysis, where pickup and delivery windows are relaxed to assess the trade-offs between fulfillment and punctuality.

2. **Carrier Unit (CU) Breakdown Disruptions:** When carrier units fail during simulation, the system implements a comprehensive lock-in strategy. All active carrier operations are preserved and allowed to complete as planned, while the failed carrier's assigned tasks are immediately dropped from the system. The re-optimisation process then redistributes these dropped tasks among the remaining operational carriers, considering their projected availability times and current workload. This ensures that the system maintains service continuity despite capacity reduction. The system also supports time window flexibility analysis for CU breakdown scenarios, where pickup and delivery windows are relaxed to assess the system's ability to recover from capacity reductions under different flexibility levels.
3. **Arc Removal Disruptions:** The system handles both permanent and temporary arc removals. For permanent removals, the arc is unavailable throughout the entire planning horizon, requiring all affected requests to be rerouted through alternative paths. For temporary removals, the arc becomes unavailable during specific time intervals (e.g., from t=7 to t=17), after which it is restored to the network. The lock-in mechanism preserves all ongoing operations that do not depend on the removed arc, while the re-optimisation process reroutes affected requests through alternative network paths, adjusting pickup and delivery windows as necessary to maintain feasibility. This approach preserves operational continuity while adapting to structural network changes.

Combined Disruption Scenarios: In addition to individual disruption types, the system is designed to handle complex scenarios involving multiple simultaneous or sequential disruptions. These combined scenarios test the system's resilience under maximum operational stress, where the MILP must simultaneously address demand surges, capacity reductions, and structural network constraints.

4.3.3. A General Framework for Sequential Disruption Analysis

While this report focuses on the impact of three specific disruptions to test system resilience, the underlying methodology is built upon a general and robust algorithm. This framework, presented in Algorithm 2, is designed to systematically incorporate any number of disruptions occurring sequentially over time. It integrates a Mixed-Integer Linear Programming (MILP) optimiser with a Discrete Event Simulation (DES) in a rolling horizon approach, allowing the system to dynamically replan its operations in response to unforeseen events.

The core logic of the algorithm is to advance through time from one disruption event to the next, treating each disruption as a trigger for re-optimisation.

Algorithm 2 Sequential Disruption Framework for DES-MILP Integration

Require: Disruption configuration $\mathcal{D} = \{(d_1, t_1), (d_2, t_2), \ldots, (d_n, t_n)\}$, Total horizon T

Extract baseline MILP inputs from case study
Set $t_{\text{opt}} = 0$
for all i in $\mathcal{D}$ **do**
    $t_i \leftarrow$ disruption time
    if $t_i = t_{\text{opt}}$ **then**
        Apply disruption logic to inputs at $t = t_{\text{opt}}$
        Prepare disrupted MILP inputs
        Run MILP optimisation to generate plan file (.pkl)
        $t_{i+1} \leftarrow$ next disruption time
        if t_{i+1} exists **then**
            Simulate DES from t_{opt} to t_{i+1} with lock-in method
            Receive Excel output from t_{opt} to t_{i+1}
            Extract MILP inputs at disruption time t_{i+1}
            $t_{\text{opt}} \leftarrow t_{i+1}$
            continue
        else
            Simulate DES from t_{opt} to T without lock-in
            Generate Excel output from t_{opt} to T
        end if
    else
        Run MILP optimisation to generate plan file (.pkl)
        Simulate DES from t_{opt} to t_i with lock-in method
        Receive Excel output from t_{opt} to t_i
        Extract MILP inputs at disruption time t_i
        Apply disruptions at $t = t_i$
        Run disrupted MILP from t_i to T to generate plan file (.pkl)
        $t_{i+1} \leftarrow$ next disruption time
        if t_{i+1} exists **then**
            Simulate DES from t_i to t_{i+1} with lock-in method
            Receive Excel output from t_i to t_{i+1}
            Extract MILP inputs at disruption time t_{i+1}
            $t_{\text{opt}} \leftarrow t_{i+1}$
            continue
        else
            Simulate DES from t_i to T without lock-in
            Generate Excel output from t_i to T
        end if
    end if
end for
Aggregate performance metrics across all simulation segments
Calculate system resilience indicators

Algorithm Explanation

The Sequential Disruption Framework is designed to integrate disruptions into a coupled DES–MILP environment in a stepwise manner. While the pseudocode explicitly distinguishes between disruptions occurring at the initial optimisation time ($t_{opt} = 0$) and later stages, the underlying logic is uniform: the

system simply moves forward in time until a disruption occurs, applies it, re-optimises, and continues simulating until the next disruption or the final horizon. The explanation below simplifies the process into a general workflow, independent of the code-specific branching.

The algorithm progresses sequentially across the planning horizon by alternating between simulation and re-optimisation steps. Its key purpose is to ensure that disruptions are incorporated at the correct times, while the system continuously adapts through re-optimised MILP plans. The main steps can be summarised as follows:

Inputs and Initialisation:- The inputs are initialised as follows:

- Disruptions: $\mathcal{D} = \{(d_1, t_1), (d_2, t_2), \ldots, (d_n, t_n)\}$, where d_i is the disruption realised at time t_i.
- Total horizon: T, the terminal time of interest.
- Baseline MILP inputs are extracted from the case study and the optimisation clock is initialised, $t_{\text{opt}} \leftarrow 0$.

Iterative Disruption Handling:- For each disruption time in ascending order, repeat:

- **Simulate to the next disruption:** Run the DES from the current optimisation time t_{opt} up to the next disruption time t_{next} using the lock-in method to honour previously fixed decisions.
- **Apply disruption at t_{next}:** Update inputs according to the disruption logic (e.g., capacity changes, resource unavailability, timing shifts).
- **Re-optimise:** Solve the MILP with the disrupted inputs to obtain a refreshed executable plan (e.g., exported as a `.pkl` plan file).
- **Record outputs:** Store the DES segment outputs (e.g., Excel reports) covering $[t_{\text{opt}},\ t_{\text{next}}]$.
- **Advance the clock:** Set $t_{\text{opt}} \leftarrow t_{\text{next}}$ and proceed to the next disruption time.

Final Stage (Post-Last Disruption):- After the final disruption has been applied and re-optimised, simulate from the last t_{opt} to the end of the horizon T *without* lock-in, and generate the final Excel output for this terminal segment.

Post-Processing:- Following post-processing steps were performed for further analysis:

- Aggregate the outputs across all simulation segments.
- Compute performance metrics and system resilience indicators to quantify response quality under the realised disruption sequence.

This chapter has detailed the methods used to introduce disruptions into the simulation framework. Established a clear process for testing the system against both predictable, deterministic failures in the toy case and more complex, cascading disruptions in the Randstad network. Furthermore, by incorporating a probabilistic model for loading delays, we have set the stage for a more realistic assessment of operational uncertainty.Validation of the DES logic (Algorithm 1) was performed using the master example in Appendix A.5.

The impact of these implemented disruptions on the system's operational efficiency and service reliability will be quantitatively analysed in the following chapter. The Key Performance Indicators (KPIs) scenarios are presented and evaluated in chapter 5.

5

Results & Discussion

This chapter presents a comprehensive analysis of the system's performance under various conditions. We will evaluate a set of Key Performance Indicators (KPIs) for our two primary case studies: the simplified Toy Case and the more complex Randstad Case Study. The analysis aims to quantify the impact of different disruptions on operational efficiency and service reliability, thereby providing insights into the system's overall resilience.

5.1. Key Performance Indicator (KPI) Selection

To evaluate the performance of the autonomous pod system, a set of Key Performance Indicators (KPIs) was selected. These KPIs are designed to provide a comprehensive view of the system's operation by measuring its overall effectiveness, the efficiency of its resources, and the quality of service offered. The performance is measured by analysing the output of the Discrete Event Simulation (DES) under both a nominal (baseline) scenario and various disruption scenarios.

5.1.1. Category 1: Service Effectiveness & Quality

These KPIs measure the system's ability to achieve its primary goal of fulfilling transport requests and the reliability of its service.

fulfilment Rate (%)

This is the most fundamental KPI, as it directly measures the success of the system in completing its tasks. A high fulfilment rate indicates an effective system, whereas a low rate suggests a failure in planning or an inability to cope with disruptions. The fulfilment rate is calculated as the percentage of successfully delivered Transport Units (TUs) relative to the total number of TUs considered for the plan. The number of fulfilled trips can be found in the "fulfilment Summary" sheet from the DES code output, shown in section 4.2.2 or just by counting the number of T4 (Delivered) states in the "Trace" sheet.

$$\text{fulfilment Rate} = \left(\frac{\text{Number of successfully delivered TUs}}{\text{Total number of TUs in the system}} \right) \times 100\%$$

Average Delivery Delay

This KPI measures the system's reliability and adherence to the optimal plan generated by the baseline MILP. It quantifies the quality of service by measuring the deviation from the planned schedule. A low average delay, particularly in a disruption scenario, indicates a robust and resilient system that can manage unexpected events effectively. It is the average of the difference between the actual delivery time recorded in the DES and the planned delivery time from the baseline MILP solution, for all completed

requests.

$$\text{Average Delay} = \frac{\sum(\text{Actual Delivery Time} - \text{Baseline Delivery Time})}{\text{Number of completed requests}}$$

5.1.2. Category 2: Resource Efficiency

These KPIs measure how well the system utilises its assets, the Carrier Units (CUs).

Carrier Utilisation (%)

This measures the operational efficiency of the carrier fleet. High utilisation indicates that these CUs are not sitting idle. However, a utilisation rate approaching 100% may suggest a lack of spare capacity, potentially reducing the system's ability to absorb disruptions. "Busy time" for a carrier includes all productive work: transporting a transport unit (TU), repositioning (travelling empty), loading, and unloading. Events where the CU is busy are E4, E5, E6, E7, E8, E9 and E10. With this KPI measure in hand, the idle time of the CU can be calculated, i.e the total % of time the CU is at the station waiting. The average utilisation across all carriers.

$$\text{Carrier Utilisation} = \left(\frac{\text{Total Busy Time}}{\text{Total Available Time}}\right) \times 100\%$$

$$\text{Idle Time (\%)} = 100\% - \text{Carrier Utilisation (\%)} \tag{5.1}$$

Empty Travel Ratio (%)

This is a direct measure of logistical inefficiency. Time spent by a carrier travelling empty to its next pickup location is necessary but unproductive (also known as deadheading). An effective plan should minimise this ratio. Comparing this KPI between nominal and disruption scenarios reveals how disruptions force less efficient movements. The ratio of time spent travelling empty to the total travel time.

$$\text{Empty Travel Ratio} = \left(\frac{\text{Total time spent repositioning (empty)}}{\text{Total travel time (transport + repositioning)}}\right) \times 100\%$$

Average TUs per CU

This Key Performance Indicator (KPI) measures the overall productivity of the carrier fleet. It quantifies, on average, how many requests each Carrier Unit (CU) completed during the operational horizon. A higher value indicates greater efficiency, suggesting that each vehicle is being used effectively to fulfill demand. Conversely, a lower value may point to underutilisation or an excess of non-productive travel, such as empty repositioning. The metric is calculated as the ratio of the total number of successfully delivered transport units to the total number of carriers in the fleet.

$$\text{Average TUs per CU} = \frac{\text{Total Fulfilled Requests}}{\text{Total Number of Carrier Units}} \tag{5.2}$$

5.1.3. Category 3: System-Specific Performance

This category focuses on measuring the performance of unique features of the autonomous system.

Platooning Rate (%)

As platooning is a key system feature designed for energy savings and efficiency, this KPI directly measures how often this strategy is successfully implemented. A higher rate indicates better coordination between carriers. The percentage of total travel time that carriers spend moving in a platoon.

$$\text{Platooning Rate} = \left(\frac{\text{Total time all CUs spent in a platoon}}{\text{Total travel time of all CUs}}\right) \times 100\%$$

5.1.4. Discussion

The Key Performance Indicators detailed in this chapter have been carefully selected to provide a multidimensional view of the system's performance. By organising them into three distinct categories, Service Effectiveness & Quality, Resource Efficiency, and System-Specific Performance can move beyond a single measure of success and create a holistic operational profile.

This approach is crucial for a comprehensive resilience analysis. For instance, a system might maintain a high fulfilment Rate (Effectiveness) during a disruption, but at the cost of a drastically increased Empty Travel Ratio (Inefficiency). Similarly, a high Carrier Utilisation rate might seem positive but could also indicate a lack of spare capacity to handle further unexpected events.

Ultimately, these KPIs are the analytical tools that will be used to interpret the simulation outputs. They provide the quantitative basis for comparing the baseline, disrupted, and re-optimised scenarios, allowing us to draw meaningful conclusions about the system's robustness and the effectiveness of the dynamic re-planning strategies. The application of these metrics to the simulation results will be presented in chapter 5.

5.2. Toy Case

The Toy Case serves as a foundational model to evaluate the core logic of the DES-MILP framework. In this section, its performance is examined under a more realistic setting involving probabilistic failures. This approach enables the establishment of a baseline understanding of how the system responds to and recovers from disruptions within a controlled environment.

Table 5.1: Disruption scenarios employed in the Toy Case for probabilistic simulation across 200 runs.

ID	Disruption Type	Affected Unit	Event	Probability (%)	Delay Range
D_TU_001	DELAY	TU	E1	10	[2, 8]
D_CU_001	DELAY	CU	E3	5	[2, 8]
D_TU_BRK_1	BREAKDOWN	TU	E4	20	[100, 150]
D_CU_BRK_1	BREAKDOWN	CU	E4	10	[0, 0]

Table 5.1 outlines the set of disruption scenarios considered in the experimental design. These scenarios encompass two types of events—delays and breakdowns—applied to both transport units (TUs) and carrier units (CUs). Each disruption is characterised by an associated event identifier, a probability of occurrence, and a corresponding delay range. For instance, minor delays are modelled for both TUs and CUs with probabilities of 10% and 5%, respectively, while breakdowns are represented as rarer but more severe events, particularly for TUs, where delays may extend from 100 to 150 time units.

These disruption scenarios were applied across 200 simulation cases. The disruptions are introduced probabilistically, based on predefined probabilities. Specifically, a random number generator determines whether a given disruption occurs: if the generated value is less than or equal to the assigned probability, the disruption is activated in that simulation run. This probabilistic application reflects the inherent uncertainty of real-world operations while maintaining a structured and repeatable testing environment.

Such a design ensures that the Toy Case not only validates the underlying mechanisms of the framework but also provides a robust foundation for subsequent experimentation with more complex and realistic scenarios.

5.2.1. KPI Summary

This subsection presents the performance of the Toy Case under probabilistic disruptions by focusing on one representative case from the 200 simulation runs. The objective is to illustrate how the system reacts to randomly applied disruptions and how optimisation measures can mitigate their impact.

Table 5.2 reports the disruptions that occurred in the selected case, together with their triggering events, affected units, and delay implications. The first disruption corresponds to a carrier unit (CU) breakdown at time step 2, triggered by event E4, which imposed a delay of 140-time units. Subsequently, a transport unit (TU) breakdown occurred at time step 11, also triggered by event E4, but with no additional delay recorded. This sequence of disruptions highlights how both CU and TU failures can emerge within the same operational horizon, demonstrating the stochastic and cumulative nature of disruption propagation.

Table 5.2: Disruptions applied in the selected simulation case.

Time	Disruption Name	Triggering Event	Affected CU	Affected TU	Delay Added
2	D_CU_BRK_1	E4	2	5	140
11	D_TU_BRK_1	E4		4	0

Building on this, Table 5.3 presents the corresponding key performance indicators (KPIs) for the baseline, disrupted, and optimised settings. The baseline reflects system performance without disruptions, while the disrupted case captures the deterioration caused by the breakdowns of the CU and TU. The optimised case incorporates the recovery strategy implemented by the DES-MILP framework.

Table 5.3: KPI summary for the Toy Case under probabilistic disruptions.

KPI	Baseline	Disrupted Network	Reoptimised Network
Total Requests	5	4	4
Fulfilled Requests	5	1	3
Unfulfilled Requests	0	3	1
fulfilment Rate (%)	100.00	25.00	75.00
Average Delivery Delay	N/A	N/A	3.33
Average Relative Delay (%)	N/A	N/A	16.60
Max Delivery Delay	N/A	N/A	7.00
Min Delivery Delay	N/A	N/A	-2.00
Number of Delayed Deliveries	N/A	N/A	2.00
Percentage of Delayed Deliveries	N/A	N/A	66.67
Carrier Utilisation (%)	42.20	10.00	33.30
Idle Time (%)	57.80	90.00	66.70
Empty Travel Ratio (%)	7.90	0.00	13.40
Platooning Rate (%)	0.00	0.00	3.70
Average TUs per CU	1.67	0.33	1.00

The results demonstrate that disruptions substantially degrade system performance. In the disrupted case, only 2 of the 5 total requests were fulfilled, resulting in a fulfilment rate of 20% compared with 100% in the baseline. Carrier utilisation also dropped sharply from 42.2% to 10%, while idle time increased to 90%. However, once the re-optimisation procedure was applied, performance improved notably. The fulfilment rate increased to 60%, demonstrating that the optimisation strategy successfully recovered part of the disrupted demand. Improvements were also observed in other operational measures. Carrier utilisation rose from 10% in the disrupted case to 33.3% in the optimised case, while idle time dropped from 90% to 66.7%. Moreover, the optimisation introduced platooning opportunities (3.7%) that were absent in both the baseline and disrupted scenarios, highlighting an additional efficiency gain. These results confirm that although the disruptions caused severe deterioration, the framework was able to restore system effectiveness to a significant extent.

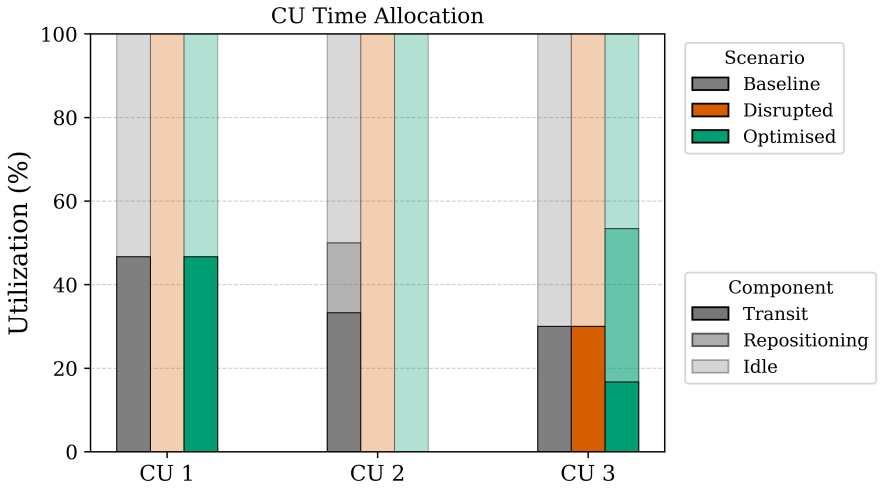

Figure 5.1: Carrier unit time allocation for Run 135 under baseline, disrupted, and optimised scenarios.

The CU time allocation in Figure 5.1 further illustrates the dynamics behind these performance differences. In the disrupted case, both CU 1 and CU 2 were completely idle due to breakdowns, while CU 3 maintained some transit activity but remained largely underutilised. The optimised scenario, however, redistributed workload more effectively: CU 1 returned to a balanced utilisation similar to the baseline, CU 2 remained idle due to persistent unavailability, and CU 3 exhibited a clear shift, with a significant share of its time reallocated to repositioning activities. This reallocation explains the partial recovery in fulfilment rate, as the framework leveraged the remaining functional resources to satisfy additional requests.

5.2.2. Aggregate Analysis (200 Runs)

While the case-specific analysis provides valuable insights into individual disruptions, the boxplots in Figure 5.2 summarise results over all 200 probabilistic runs, thereby offering a broader view of system behaviour. These 200 probabilistic disruptions were applied to the baseline with the probability of occurrence specified in Table 5.1.

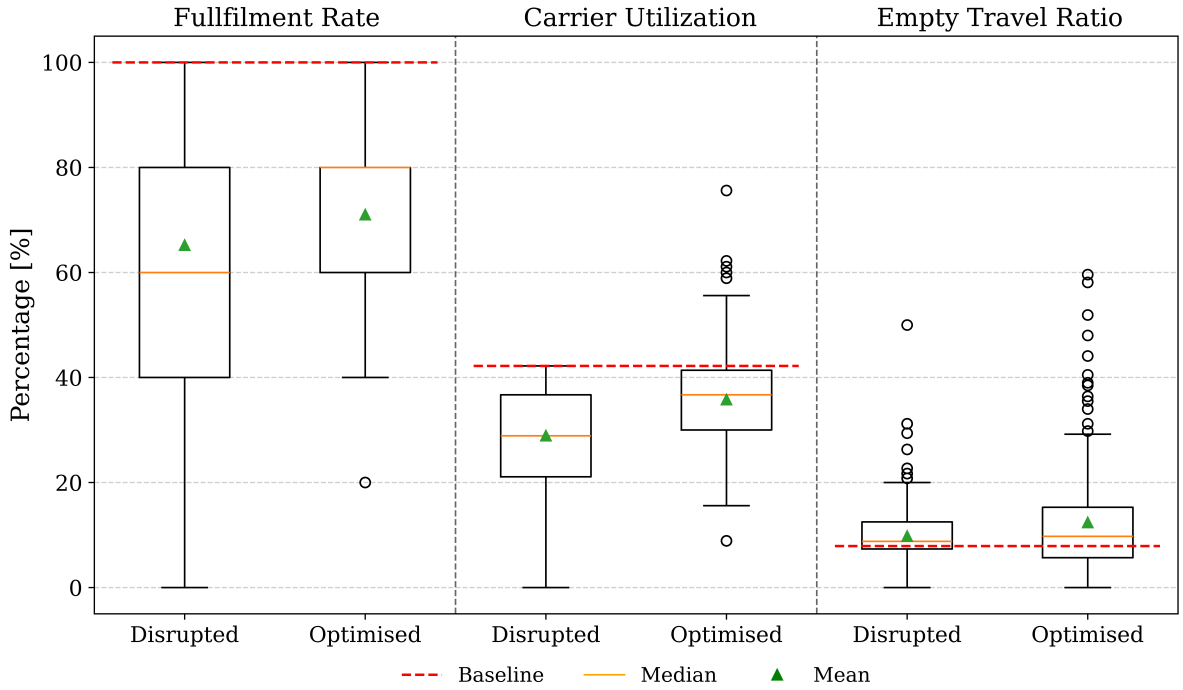

Figure 5.2: Distribution of fulfilment rate, carrier utilisation, and empty travel ratio across 200 runs for disrupted and optimised cases. Baseline values are shown as dashed lines.

The fulfilment rate distribution highlights the strong adverse impact of disruptions, with the median value in the disrupted case falling to around 60% and a wide variance across runs. In comparison, the optimised case demonstrates a consistent upward shift, with median fulfilment rising to approximately 80%. Although the baseline value of 100% is not fully restored, the improvement is substantial and consistent across runs, underscoring the robustness of the optimisation approach.

Carrier utilisation exhibits a similar pattern. The disrupted case is characterised by lower median utilisation, reflecting widespread CU inactivity during breakdowns. The optimisation process improves utilisation, pushing the distribution closer to the baseline, though again not fully reaching it.

Finally, the empty travel ratio shows a more nuanced effect. In the disrupted case, the ratio is generally low, as many CUs are idle rather than moving. Optimisation increases the ratio somewhat, since repositioning movements are introduced to reconfigure the system. While this may appear less efficient in isolation, it represents a necessary trade-off, enabling higher fulfilment rates and better overall resource usage.

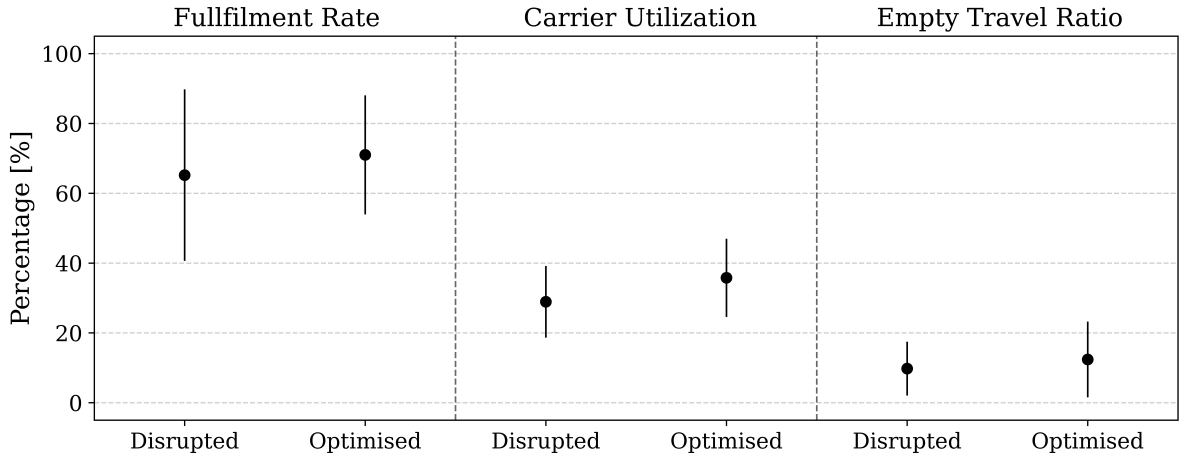

Figure 5.3: Errorbar plot of fulfilment rate, carrier utilisation, and empty travel ratio across 200 runs for disrupted and optimised cases.

The performance of the system in terms of fulfilment rate, carrier utilisation, and empty travel ratio was further examined using measures of variability and precision. Figure 5.3 shows the error bar plot visualising the $\mu \pm \sigma$, where μ is the mean and σ is the standard deviation. In the optimised case, fulfilment rate had a mean of 71.0% with a standard deviation of 16.8%, while in the disrupted case, the mean dropped to 65.2% with an even larger standard deviation of 24.3%. This indicates substantial variability in fulfilment across simulation runs. By contrast, carrier utilisation was more stable, with mean values of 35.8% ± 11.0% (optimized) and 28.9% ± 10.0% (disrupted). The empty travel ratio showed the lowest spread, at 12.4% ± 10.6% (optimized) and 9.8% ± 7.5% (disrupted).

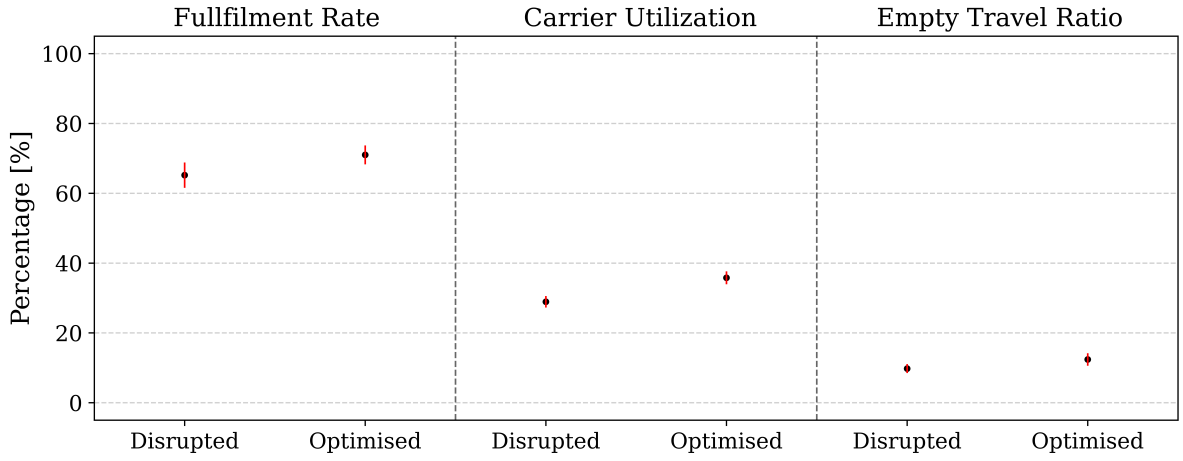

Figure 5.4: 95% confidence intervals of fulfilment rate, carrier utilisation, and empty travel ratio across 200 runs for disrupted and optimised cases.

When considering the 95% confidence intervals of the mean (Figure 5.4), the results demonstrate that despite the variability, the mean estimates are statistically reliable. For fulfilment rate, the CIs were [68.5–73.5]% (optimised) and [61.8–68.6]% (disrupted). Carrier utilisation had narrow CIs of [34.2–37.4]% (optimised) and [27.5–30.3]% (disrupted), while the empty travel ratio showed [10.8–14.0]% (optimised) and [8.7–10.8]% (disrupted). The confidence intervals are considerably narrower than the standard deviations due to the large number of runs, which strengthens confidence in the estimated means.

Beyond the primary KPIs analysed in the disruption scenarios, the system performance evaluation

encompasses a comprehensive set of operational metrics that provide deeper insights into the system's behaviour under various conditions. These additional metrics include detailed statistical analysis of fulfilment rates, delivery delays, carrier utilisation patterns, and operational efficiency indicators across multiple simulation runs.

The statistical analysis, based on 200 simulation runs, reveals significant performance variations between optimised and disrupted system states. The optimised system demonstrates higher fulfilment rates and carrier utilisation, while the disrupted system shows reduced operational efficiency but maintains better delivery punctuality. Detailed statistical summaries, including mean values, standard deviations, median values, and 95% confidence intervals for all performance metrics, are provided in Tables A.4 and A.5 in the Appendix.

In this context, robustness means the ability to keep service performance high, and variability controlled, when the system is perturbed—without retuning the model. The Toy Case results indicate that the DES–MILP loop is robust in that sense. In the representative run, re-optimisation lifts fulfilment from a severely disrupted state (25%) to 75%, with carrier utilisation recovering from 10.0% to 33.3% while platooning reappears (3.7%)—at the expected cost of more empty repositioning (13.4%). These patterns show that the framework can systematically reconfigure assets to restore service after shocks.

Aggregated over 200 stochastic runs, the robustness signal is stronger: the *median* fulfilment shifts from roughly 60% (disrupted) to roughly 80% (re-optimised), and the *dispersion* of outcomes narrows (fulfilment standard deviation reduces from 24.3% to 16.8%). Mean utilisation increases ($28.9\% \pm 10.0\%$ to $35.8\% \pm 11.0\%$), while the empty-travel ratio rises moderately ($9.8\% \pm 7.5\%$ to $12.4\% \pm 10.6\%$), evidencing a controlled efficiency trade-off in exchange for more reliable service. Overall, the combination of (i) consistent uplift in fulfilment across seeds, and (ii) reduced variability in outcomes, supports the claim that the proposed re-optimisation policy delivers robust performance under probabilistic disruptions.

Key Insights

The combined analysis of individual runs and aggregated outcomes leads to several key insights. First, disruptions lead to severe degradation in both service quality and resource efficiency, confirming the system's vulnerability to stochastic failures. Second, the optimisation framework consistently mitigates these effects, restoring a substantial share of lost performance. In particular, the fulfilment rate shows marked improvements, with recovery from severe shortfalls to moderate-to-high levels across most runs. Finally, the results demonstrate that optimisation does not merely stabilise the system but actively reconfigures resource usage, introducing repositioning and platooning strategies that were absent in the baseline, thereby enhancing resilience.

5.3. Randstad Case Study

Building upon the insights from the Toy Case, the analysis will be applied to the large-scale Randstad Case Study. This section evaluates the system's resilience against more complex and impactful disruptions, reflecting realistic operational challenges. Comparison is made with baseline performance against three scenarios: an unexpected increase in demand, a critical carrier breakdown, an arc breakdown, and a combination of these two disruptions.

5.3.1. TU Addition

This subsection analyses how the system responds to sudden increases in demand through the addition of new transport units during simulation. The analysis examines both simultaneous and non-simultaneous demand surge scenarios to understand the system's capacity to handle varying patterns of increased workload. Followed by time flexibility and time bracket analysis.

Two distinct demand surge patterns are investigated to evaluate system performance under different temporal distributions of increased demand:

Simultaneous Demand Surge:- Three transport units (TU4, TU15, and TU18) are added simultaneously at time t=0, representing an immediate and substantial increase in system workload. This scenario tests the system's ability to handle a sudden spike in demand from the beginning of the simulation period.

Non-Simultaneous Demand Surge:- Transport units are added at different time points to simulate a more gradual increase in demand. TU4 is introduced at time t=4, followed by TU15 at time t=25, creating a staggered demand pattern that allows the system to adapt incrementally to increased workload.

Analysis Framework

The comparison between baseline performance and these demand surge scenarios provides insights into:

- System capacity utilisation under increased demand
- Effectiveness of re-optimisation strategies for different demand patterns
- Trade-offs between simultaneous vs. staggered demand introduction
- Impact on key performance indicators such as fulfilment rates, delivery delays, and carrier utilisation

This analysis establishes the foundation for understanding how the system's adaptive capabilities respond to different types of demand disruptions, setting the stage for a more detailed examination of each scenario's specific characteristics and performance outcomes.

Simultaneous surge in request (TU) demand:

This analysis examines the system's performance when three additional transport units (TU4, TU15, and TU18) are added simultaneously at time t=0, creating an immediate surge in demand that tests the system's capacity to handle increased workload from the beginning of the simulation period.

The characteristics of the added transport units are detailed in Table 5.4.

Table 5.4: Transport Units Added for Simultaneous Demand Surge Analysis

TU ID	Origin	Destination	Pickup Window	Delivery Window
TU4	Station 7	Station 8	[4, 10]	[16, 24]
TU15	Station 10	Station 9	[25, 30]	[33, 39]
TU18	Station 2	Station 3	[14, 20]	[25, 33]

This simultaneous demand surge increases the total system workload from 15 to 18 transport requests, representing a 20% increase in demand. The analysis compares three scenarios: the baseline system with original demand, the disrupted system with increased demand but no re-optimisation, and the re-optimised system that adapts to the new demand pattern.

The KPIs explained in table 5.5 demonstrate how the system responds to a sudden increase in workload and the effectiveness of re-optimisation strategies in improving fulfilment rates and operational efficiency under simultaneous demand surge conditions.

Table 5.5: KPI Comparison for Randstad TU Addition (Simultaneous Surge)

KPI Name	Baseline	Disrupted Network	Reoptimised Network
Total Requests	15	18	18
Fulfilled Requests	15	15	16
Unfulfilled Requests	0	3	2
fulfilment Rate (%)	100.00	83.00	88.00
Average Delivery Delay	N/A	N/A	0.36
Average Relative Delay (%)	N/A	N/A	1.60
Max Delivery Delay	N/A	N/A	4.00
Min Delivery Delay	N/A	N/A	-1.00
Number of Delayed Deliveries	N/A	N/A	2.00
Percentage of Delayed Deliveries	N/A	N/A	14.29
Carrier Utilisation (%)	56.80	56.80	54.50
Idle Time (%)	43.20	43.20	45.50
Empty Travel Ratio (%)	11.70	11.70	8.70
Platooning Rate (%)	24.60	24.60	18.70
Average TUs per CU (All CUs)	1.50	1.50	1.60

Non-simultaneous surge in request (TU) demand:
This scenario simulates an unexpected surge in demand by introducing additional Transport Unit (TU) requests at the start of the simulation that is at t = 0 and at t = 25. The analysis focuses on the system's ability to adapt its plan to accommodate new, high-priority tasks. The performance impact is detailed in table 5.6.

Table 5.6: KPI Comparison for Randstad TU Addition (Non-simultaneous Surge)

KPI Name	Baseline	Disrupted Network	Reoptimised network
Total Requests	15	17	17
Fulfilled Requests	15	15	16
Unfulfilled Requests	0	2	1
fulfilment Rate (%)	100.00	88.00	94.00
Average Delivery Delay	N/A	N/A	0.73
Average Relative Delay (%)	N/A	N/A	4.10
Max Delivery Delay	N/A	N/A	6.00
Min Delivery Delay	N/A	N/A	-1.00
Number of Delayed Deliveries	N/A	N/A	4.00
Percentage of Delayed Deliveries	N/A	N/A	26.67
Carrier Utilisation (%)	56.80	56.80	53.80
Idle Time (%)	43.20	43.20	46.20
Empty Travel Ratio (%)	11.70	11.70	10.70
Platooning Rate (%)	24.60	24.60	14.90
Average TUs per CU (All CUs)	1.50	1.50	1.50

In this analysis, TU 4 and TU 15 are added to the system. At every TU addition, the system will re-optimise the system depending upon the current CU and TU status.

A critical observation across both scenarios is the system's adaptive resource reallocation behaviour, where existing baseline requests are dynamically managed to accommodate new demands.

Baseline Request Replacement Mechanism: A particularly noteworthy observation in both scenarios is the replacement of TU3 from the original baseline. When new TUs were introduced, the system's re-optimisation process effectively "replaced" TU3 by reallocating its assigned resources to fulfill the new demands. This replacement mechanism demonstrates the MILP model's ability to dynamically prioritise and reallocate resources based on current system conditions, rather than maintaining rigid adherence to the original baseline plan.

Scenario 1 - Simultaneous TU Addition (TU4, TU15, TU18 at t=0): This scenario investigates the system's performance when three new Transport Units (TU4, TU15, and TU18) are added simultaneously at the very beginning of the simulation (t=0), increasing the total demand from 15 to 18 requests. The replacement of TU3 by TU4 and TU15 results in a net increase of 2 requests (15 - 1 + 2 = 16 total requests) as seen in table 5.4.

Key Observations from Simultaneous TU Addition:

- **fulfilment Impact:** The addition of 3 TUs at t=0 immediately reduced the fulfilment rate from 100% (Baseline) to 83% in the disrupted state, as the system could only fulfill the original 15 requests. Post-re-optimisation, the fulfilment rate improved to 88%, indicating that the re-optimised plan successfully integrated and fulfilled two of the three new requests, reducing unfulfilled requests from 3 to 2.
- **Delivery Delays:** The re-optimised network shows an average delivery delay of 0.36 time units, with a maximum delay of 4.00 time units. Interestingly, a minimum delay of -1.00 indicates that some deliveries were completed earlier than their original planned times, suggesting efficient re-routing. Two deliveries (14.29% of relevant deliveries) experienced delays.
- **Carrier Utilisation and Idle Time:** Carrier Utilisation slightly decreased from 56.80% (Baseline/Disrupted) to 54.50% in the re-optimised network. Correspondingly, Idle Time increased from 43.20% to 45.50%. This suggests that while the system handled more requests, the re-optimisation prioritised fulfilment and delay reduction over maximising carrier utilisation.
- **Empty Travel Ratio:** A significant improvement was observed in the Empty Travel Ratio, which decreased from 11.70% to 8.70% in the re-optimised network. This indicates that the re-optimisation effectively found more efficient routes, reducing the carrier repositioning trip.
- **Platooning Rate:** The Platooning Rate decreased from 24.60% to 18.70% after re-optimisation. This suggests that to accommodate the new requests and optimise other KPIs, the re-optimisation had to break up some platoons, as platooning often requires strict timing and routing, which can be less flexible under dynamic conditions.
- **Average TUs per CU:** The Average TUs per CU slightly increased from 1.50 to 1.60 in the re-optimised network, reflecting the carriers handling a marginally higher workload.

Scenario 2 - Non-Simultaneous TU Addition (TU4 at t=0, TU15 at t=25): This scenario examines a staggered demand surge where TU4 is introduced at the simulation start (t=0), followed by TU15 later in the simulation at t=25. Similar to the simultaneous scenario, TU3 from the baseline was replaced by TU4, resulting in a net increase of 1 request ($15 - 1 + 2 = 16$ total requests).

Key Observations from Non-Simultaneous TU Addition:

- **fulfilment Impact:** The addition of 2 TUs (TU4 at t=0, TU15 at t=25) increased total requests to 17. The disrupted state maintained 15 fulfilled requests, resulting in an 88% fulfilment rate. Post-re-optimisation, the fulfilment rate improved to 94%, successfully fulfilling 16 out of 17 requests and reducing unfulfilled requests from 2 to 1.
- **Delivery Delays:** The re-optimised network shows an average delivery delay of 0.73-time units, with a maximum delay of 6.00 time units. Four deliveries (26.67% of fulfilled requests) experienced delays, which is higher than the simultaneous scenario, suggesting that staggered additions may create more scheduling challenges.
- **Carrier Utilisation and Idle Time:** Carrier Utilisation decreased from 56.80% to 53.80% in the re-optimised network, with Idle Time increasing to 46.20%. This indicates that the re-optimisation prioritised fulfilment over utilisation efficiency.
- **Empty Travel Ratio:** The Empty Travel Ratio improved from 11.70% to 10.70% in the re-optimised network.
- **Platooning Rate:** The Platooning Rate significantly decreased from 24.60% to 14.90% after re-optimisation, indicating that the staggered addition pattern made platooning more challenging to maintain.

- **Average TUs per CU:** The Average TUs per CU remained constant at 1.50 across all scenarios, suggesting that the staggered addition pattern did not significantly alter the average carrier workload.

Comparative Analysis: Comparing both scenarios reveals a clear trade-off between fulfilment quantity and delivery quality. The non-simultaneous addition strategy demonstrates superior fulfilment capabilities, achieving a 94% fulfilment rate compared to 88% for the simultaneous approach, successfully fulfilling 16 requests with only 1 unfulfilled, versus 2 unfulfilled in the simultaneous scenario. Conversely, the simultaneous addition strategy excels in delivery punctuality and operational efficiency for fulfilled requests, achieving significantly lower average delivery delays (0.36 vs. 0.73 time units), fewer delayed deliveries (2 vs. 4), lower empty travel ratio (8.7% vs. 10.7%), and higher platooning rate (18.7% vs. 14.9%). This indicates that while the non-simultaneous approach prioritises fulfilling as many requests as possible through incremental adaptation, the simultaneous approach prioritises efficient delivery and operational excellence for the requests it can fulfill, suggesting that the choice between strategies depends on whether the primary objective is maximising fulfilment quantity or ensuring high-quality delivery performance for fulfilled requests.

Adding to this analysis, TU addition disruption was tested with varying time window flexibility and time bracket analysis. This investigation explores how the system's re-optimisation capabilities are influenced by the allowance to relax delivery and pickup time windows, as well as the timing of dynamic request introductions. The analysis examines two key dimensions: (1) the impact of time window flexibility levels (0, 10, and 40-time units) when adding 2 TUs at a specific point in time, and (2) the effect of adding 3 TUs at different time brackets (t=0, t=10, t=20) to understand how timing affects system performance and resource allocation.

TU time window flexibility analysis

Two dynamic transport units (TU15 and TU18) are added at time t=14 to simulate mid-simulation demand changes. The system is then re-optimised with varying levels of time window flexibility to accommodate these new requests. The characteristics of the added transport units are detailed in Table 5.7.

Table 5.7: Dynamic Transport Units Added for Time Window Flexibility Analysis

TU ID	Origin	Destination	Pickup Window	Delivery Window
TU15	Station 10	Station 9	[25, 30]	[33, 39]
TU18	Station 2	Station 3	[14, 20]	[25, 33]

The analysis compares five scenarios to evaluate the trade-offs between system flexibility and performance:

1. **Baseline**: Original system with 15 requests, no disruptions
2. **Disrupted**: System with 17 total requests (15 original + 2 dynamic) but no re-optimisation
3. **Flexibility = 0**: Re-optimisation with no window relaxation (strict time constraints)
4. **Flexibility = 10**: Re-optimisation with 10 time units of window relaxation
5. **Flexibility = 40**: Re-optimisation with 40 time units of window relaxation (maximum flexibility)

This analysis reveals the critical trade-off between maintaining strict delivery schedules and achieving higher fulfilment rates through increased operational flexibility. The results demonstrate how window relaxation strategies can improve system resilience while potentially introducing delivery delays. Table 5.8 details the list of KPIs analysed in this scenario.

Table 5.8: KPI Comparison for Randstad TU Addition (Flexible Window Analysis)

KPI Name	Baseline	Disrupted	Time window flexibility = 0	Time window flexibility = 10	Time window flexibility = 40
Total Requests	15	17	17	17	17
Fulfilled Requests	15	15	13	15	15
Unfulfilled Requests	0	2	4	2	2
fulfilment Rate (%)	100.00	88.00	76.00	88.00	88.00
Average Delivery Delay	N/A	N/A	0.46	2.93	3.00
Average Relative Delay (%)	N/A	N/A	1.80	9.80	10.10
Max Delivery Delay	N/A	N/A	9.00	10.00	10.00
Min Delivery Delay	N/A	N/A	-4.00	-4.00	-4.00
Number of Delayed Deliveries	N/A	N/A	2.00	8.00	8.00
Percentage of Delayed Deliveries	N/A	N/A	15.38	53.33	53.33
Carrier Utilisation (%)	50.50	50.50	49.00	58.00	58.00
Idle Time (%)	49.50	49.50	51.00	42.00	42.00
Empty Travel Ratio (%)	9.70	9.70	12.10	12.00	10.70
Platooning Rate (%)	18.00	18.00	23.10	26.20	23.30
Average TUs per CU (All CUs)	1.50	1.50	1.30	1.50	1.50

The time window flexibility analysis reveals critical trade-offs between fulfilment quantity and delivery punctuality. When 2 TUs are added with no time window flexibility (flexibility = 0), the system achieves a 76% fulfilment rate, fulfilling only 13 out of 17 requests (as shown in table 5.8). However, as flexibility increases to 10 and 40-time units, the fulfilment rate improves to 88%, successfully fulfilling 15 requests and reducing unfulfilled requests from 4 to 2. This improvement comes at a significant cost to delivery punctuality: average delivery delays increase from 0.46-time units (flexibility = 0) to 2.93 time units (flexibility = 10) and 3.00 time units (flexibility = 40). The percentage of delayed deliveries

also increases dramatically from 15.38% to 53.33% with higher flexibility levels, indicating that while the system can accommodate more requests by relaxing time constraints, it does so at the expense of delivery timeliness.

Time Bracket analysis

This subsection presents a comprehensive analysis of system performance under different timing scenarios for dynamic request addition. The study investigates how the timing of introducing additional transport units (TUs) affects overall system efficiency, fulfilment rates, and operational metrics.

Table 5.9: KPI Comparison for Randstad TU Addition (Time Bracket Analysis)

KPI Name	Baseline	Disrupted	TU added at t = 0	TU added at t = 10	TU added at t = 20
Total Requests	15	18	18	18	18
Fulfilled Requests	15	15	16	13	15
Unfulfilled Requests	0	3	2	5	3
fulfilment Rate (%)	100.00	83.33	88.89	72.22	83.33
Average Delivery Delay	N/A	N/A	0.36	2.25	0.60
Average Relative Delay (%)	N/A	N/A	2.60	8.00	1.90
Max Delivery Delay	N/A	N/A	11.00	10.00	6.00
Min Delivery Delay	N/A	N/A	-2.00	-7.00	-3.00
Number of Delayed Deliveries	N/A	N/A	1.00	5.00	4.00
Percentage of Delayed Deliveries	N/A	N/A	7.14	41.67	26.67
Carrier utilisation (%)	50.50	50.50	50.80	53.00	52.50
Idle Time (%)	49.50	49.50	49.20	47.00	47.50
Empty Travel Ratio (%)	9.70	9.70	6.00	8.90	12.20
Platooning Rate (%)	18.00	18.00	13.90	14.70	25.00
Average TUs per CU (All CUs)	1.50	1.50	1.60	1.60	1.50

Experimental Setup
Three dynamic transport units (TU4, TU15, and TU18) were introduced at different time points during the simulation horizon to evaluate the system's adaptive capabilities. The characteristics of these additional requests are detailed in Table 5.10.

Table 5.10: Dynamic Transport Units Added for Time Bracket Analysis

TU ID	Origin	Destination	Pickup Window	Delivery Window
TU4	Station 7	Station 8	[4, 10]	[16, 24]
TU15	Station 10	Station 9	[25, 30]	[33, 39]
TU18	Station 2	Station 3	[14, 20]	[25, 33]

Time Bracket Scenarios
The analysis examines four distinct scenarios:

1. **Baseline**: Original system with 15 requests, no disruptions
2. **Early Addition (t=0)**: Dynamic TUs added at simulation start with immediate re-optimisation
3. **Medium Addition (t=10)**: Dynamic TUs added at t=10 with re-optimisation
4. **Late Addition (t=20)**: Dynamic TUs added at t=20 with re-optimisation

This time bracket analysis provides insights into the optimal timing for introducing additional transport requests and the system's ability to adapt to changing demand patterns throughout the planning horizon. The KPIs (table 5.9) demonstrate how early intervention can improve fulfilment rates while late additions may lead to increased delays and operational inefficiencies.

The time bracket analysis demonstrates how the timing of TU additions significantly impacts system performance. Adding 3 TUs at different simulation times reveals distinct performance patterns:

Early Addition (t=0): Adding TUs at the simulation start results in the best overall performance, achieving an 88.89% fulfilment rate with the lowest average delivery delay (0.36-time units) and minimal delayed deliveries (7.14%). This suggests that early integration allows the system to optimally plan and allocate resources from the beginning. Dynamic requests 4 and 15 are fulfilled in this bracket; that is, 2 of 3 added requests are fulfilled.

Mid-Simulation Addition (t=10): Adding TUs at t=10 presents the most challenging scenario, resulting in the lowest fulfilment rate (72.22%) and highest average delivery delay (2.25-time units). The system struggles to accommodate mid-simulation demand surges, with 41.67% of deliveries experiencing delays, indicating that mid-simulation disruptions create the most complex scheduling challenges. The dynamic request TU15 added in this bracket is fulfilled. That is 1 of 3 added requests that are fulfilled.

Late Addition (t=20): Adding TUs at t=20 shows moderate performance, achieving an 83.33% fulfilment rate with moderate delays (0.60-time units average). This suggests that late additions provide some planning flexibility but still create scheduling constraints. Here, none of the dynamically added requests are fulfilled.

Operational Efficiency Trends: The analysis reveals several operational efficiency trends across different scenarios. Carrier utilisation generally increases with TU additions, peaking at 53.0% when TUs are added at t=10, indicating that mid-simulation additions require higher carrier workload. The empty travel ratio shows interesting patterns, with the lowest ratio (6.0%) occurring when TUs are added at t=0, suggesting more efficient routing for early additions. Platooning rates vary significantly, with the highest rate (25.0%) occurring at t=20, indicating that late additions provide more opportunities for carrier coordination and platoon formation.

Strategic Implications: The combined analysis of time window flexibility and time bracket effects reveals that system performance is highly sensitive to both the timing of disruptions and the flexibility allowed in time constraints. Early TU additions with moderate time window flexibility appear to provide the best balance between fulfilment and delivery quality, while mid-simulation additions create the most challenging scenarios regardless of flexibility levels. This suggests that proactive demand management and early integration strategies may be more effective than reactive approaches to dynamic demand surges.

5.3.2. CU Breakdown

Here, a critical carrier failure is modelled by simulating the unexpected breakdown of a Carrier Unit Initially a single CU breakdown disruption is analyzed to test the system's ability to re-route tasks and maintain service levels with reduced fleet capacity. The resulting KPIs are compared against the baseline in table 5.11. At t = 5, CU 5 breaks, and the system will lock in the trips that are in transit and will re-optimise from time horizon 5 to 40. This CU breakdown analysis is extended to capture how the system reacts to time window flexibility.

Table 5.11: KPI Comparison for Randstad CU Breakdown

KPI Name	Baseline	Disrupted Network	Reoptimised Network
Total Requests	15	15	15
Fulfilled Requests	15	13	14
Unfulfilled Requests	0	2	1
fulfilment Rate (%)	100.00	87.00	93.00
Average Delivery Delay	N/A	N/A	3.07
Average Relative Delay (%)	N/A	N/A	11.80
Max Delivery Delay	N/A	N/A	10.00
Min Delivery Delay	N/A	N/A	-3.00
Number of Delayed Deliveries	N/A	N/A	9.00
Percentage of Delayed Deliveries	N/A	N/A	64.29
Carrier Utilisation (%)	56.80	56.40	65.30
Idle Time (%)	43.20	43.60	34.70
Empty Travel Ratio (%)	11.70	12.50	11.70
Platooning Rate (%)	24.60	23.50	17.80
Average TUs per CU (All CUs)	1.50	1.44	1.56

A set of key performance indicators (KPIs) is compared across the baseline, disrupted, and reoptimised networks to quantify the immediate and post-reoptimisation impacts of the breakdown. Second, the allocation of CU time is assessed to illustrate how operational activities such as transit, repositioning, and idle periods are redistributed when one CU becomes unavailable. Finally, the role of temporal flexibility is investigated by introducing alternative delivery time windows, thereby assessing the extent to which scheduling flexibility can mitigate disruption and restore system performance.

Single CU Breakdown: The breakdown of a single CU introduces a range of performance impacts that can be observed both at the system level and in the distribution of operational activities across the

remaining CUs. These effects are summarised in Table 5.11 and visualised in Figure 5.5.

Impacts on Key Performance Indicators (Table 5.11):

- **Service performance:**
  - Fulfilment rate decreases from 100% in the baseline to 87% in the disrupted scenario, corresponding to two unfulfilled requests.
  - Reoptimisation partially restores performance, increasing the fulfilment rate to 93% and reducing the number of unfulfilled requests to one.
  - The improvement comes at the expense of delivery punctuality, with an average relative delay of 11.8% and nine delayed deliveries (64% of total).
- **Utilisation and idle time:**
  - Carrier utilisation remains largely unchanged between the baseline (56.8%) and the disruption (56.4%), suggesting limited immediate redistribution of tasks.
  - Reoptimisation, however, increases utilisation to 65.3% and reduces idle time to 34.7%, reflecting intensified use of the remaining units.
- **Operational efficiency:**
  - The empty travel ratio remains broadly constant (around 12%), indicating that repositioning inefficiencies are not significantly worsened by the breakdown.
  - The platooning rate falls from 24.6% in the baseline to 17.8% in the reoptimised case, signalling a decline in cooperative opportunities as the system prioritises request fulfilment.

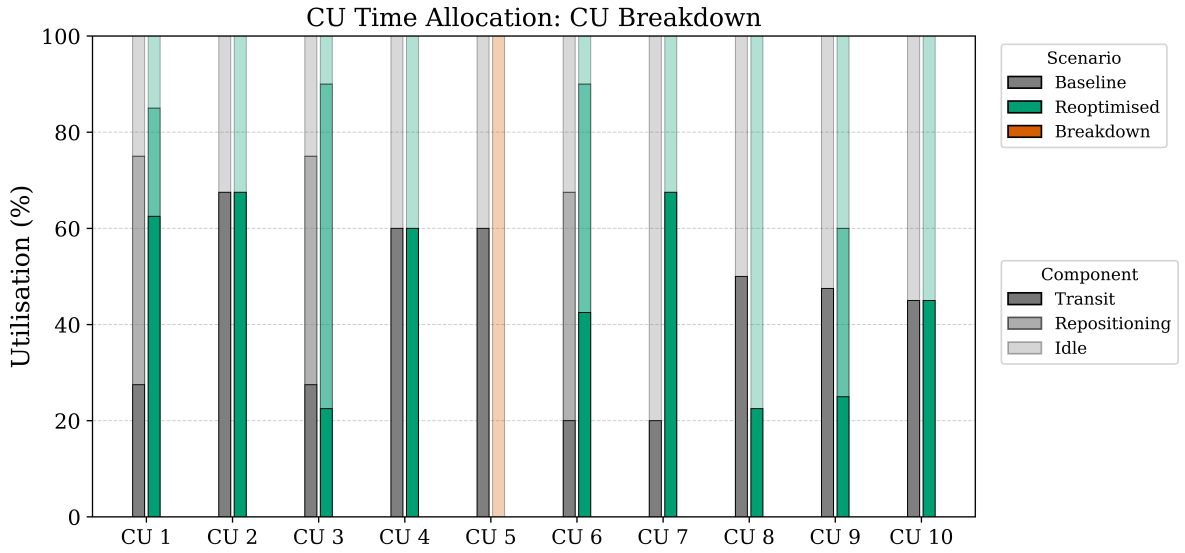

Figure 5.5: CU utilisation summary for the CU breakdown disruption

Impacts on CU Time Allocation (Figure 5.5):

- CU 5, the disrupted unit, shows a complete absence of activity, eliminating its contribution to transit, repositioning, or idle time.
- The lost workload is redistributed primarily to CU 2, CU 3, and CU 6, which experience notable increases in transit and repositioning shares.
- Idle time is consistently reduced across most CUs, consistent with the higher system-wide utilisation observed in the KPI table.

- The redistribution results in greater operational imbalances across the fleet, with some CUs carrying disproportionately higher loads.
- These imbalances constrain flexibility and reduce the prevalence of platooning, thereby limiting efficiency-enhancing behaviours.

In summary, while reoptimisation allows the system to absorb the effects of a CU breakdown and partially restore service levels, it does so by intensifying the utilisation of remaining units and reducing operational slack. This creates a trade-off between system resilience and efficiency.

Further tests are run to examine the system's resilience and adaptability when a carrier unit experiences a breakdown during operation. The analysis examines how varying levels of time window flexibility can mitigate the impact of carrier failures on overall system performance and request fulfilment.

Experimental Configuration: Carrier Unit 5 (CU5) experiences a breakdown at time t = 5, representing a realistic operational disruption scenario. This carrier failure occurs early in the simulation timeline, requiring the system to immediately re-optimise the remaining carrier assignments to handle the workload previously allocated to the failed carrier. The system must then adapt to this reduced carrier capacity while maintaining service quality for existing and future transport requests.

The analysis compares five scenarios to assess the effectiveness of time window flexibility in managing carrier breakdowns:

1. **Baseline**: Original system with all 10 carriers operational, no disruptions
2. **Disrupted**: System with CU5 failure at t=5 but no re-optimisation applied
3. **Flexibility = 0**: Re-optimisation with no window relaxation (strict adherence to original schedules)
4. **Flexibility = 10**: Re-optimisation with 10 time units of window relaxation
5. **Flexibility = 40**: Re-optimisation with 40 time units of window relaxation (maximum operational flexibility)

This analysis demonstrates how time window flexibility can serve as a critical recovery mechanism when the system experiences carrier capacity reduction. The results reveal the trade-offs between maintaining strict delivery commitments and achieving higher fulfilment rates through increased scheduling flexibility in the face of operational disruptions.The KPIs are detailed in table A.3.

Flexibility Window Analysis: The role of temporal flexibility in mitigating the impacts of a CU breakdown is shown in Figure 5.6. The figure reports changes in fulfilment rate, carrier utilisation, and empty travel ratio, all expressed relative to the baseline scenario. The results demonstrate that the introduction of delivery time windows substantially alters system performance across these dimensions.

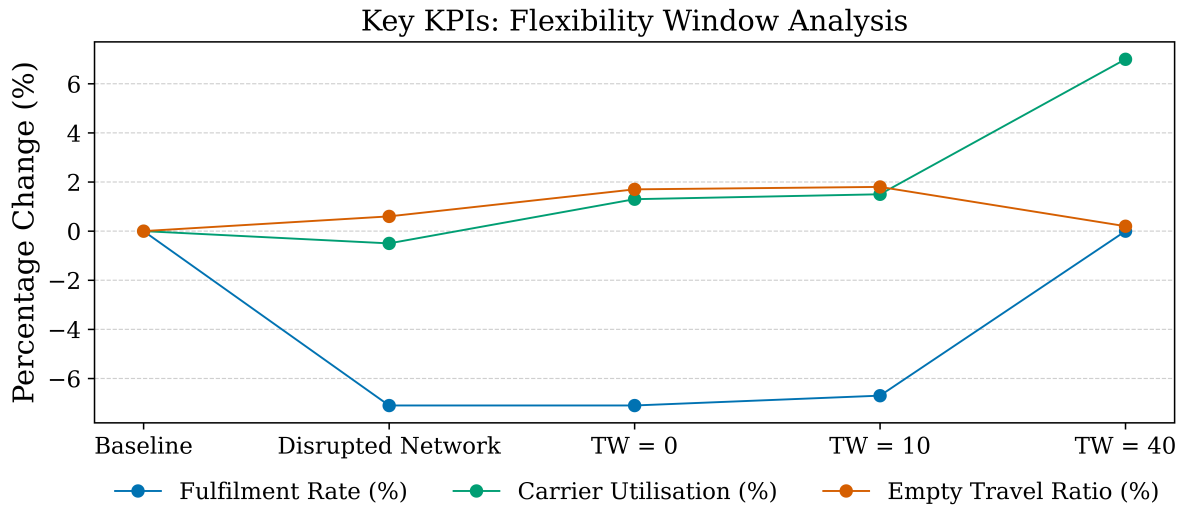

Figure 5.6: KPI percentage change variation from baseline with flexible time windows

Service performance (Fulfilment Rate):

- The CU breakdown leads to a reduction of approximately −7.1% relative to baseline.
- Reoptimisation without added flexibility (TW = 0) does not alter this shortfall, as the rate remains at −7.1%.
- A 10-unit window offers only a marginal improvement, raising the rate to −6.7%.
- With 40 units of flexibility, the system fully recovers, returning to baseline performance (0.0% change).

Resource efficiency (Carrier Utilisation):

- The breakdown slightly reduces utilisation by −0.5%.
- Even with no flexibility (TW = 0), reoptimisation improves utilisation by +1.3%.
- At TW = 10, utilisation increases further to +1.5%.
- The greatest improvement occurs at TW = 40, where utilisation rises sharply by +7.0%, indicating highly effective deployment of available CUs when broader scheduling freedom is permitted.

Operational trade-offs (Empty Travel Ratio):

- Empty travel initially increases by +0.6% following the breakdown.
- Under TW = 0, the ratio rises further to +1.7%, and at TW = 10 it peaks at +1.8%.
- At TW = 40, the ratio returns close to baseline, with only a marginal increase of +0.2%, suggesting that higher flexibility enables more efficient routing that offsets the earlier repositioning burden.

In summary, the introduction of delivery flexibility produces two important outcomes. First, it directly supports service quality by restoring the fulfilment rate to baseline levels once sufficient flexibility (40 minutes) is granted. Second, it enhances system efficiency by increasing carrier utilisation and ultimately stabilising empty travel requirements. Small levels of flexibility provide only incremental benefits, whereas higher flexibility allows the network to both recover service levels and achieve a more efficient deployment of resources.

5.3.3. Arc Removal

This disruption tests the system's ability to re-route tasks in response to an unexpected arc removal/block. The resulting KPIs (table 5.12) are compared against the baseline. The arc (1,3) is broken from time 7 to 17; during this time period, the arc is inactive and cannot be used by any of the carriers. At t = 7, the carriers which are currently using this arc are locked in to complete their trips, and the carriers that would eventually be using this arc are rerouted. Similarly, at t = 17, the arc reopens, and the simulation continues from 17 to 40.

Table 5.12: KPI Comparison for Randstad Arc Removal

KPI Name	Baseline	Disrupted Network	Reoptimised network
Total Requests	15	15	15
Fulfilled Requests	15	13	15
Unfulfilled Requests	0	2	0
fulfilment Rate (%)	100.00	87.00	100.00
Average Delivery Delay	N/A	N/A	-0.93
Average Relative Delay (%)	N/A	N/A	-2.3
Max Delivery Delay	N/A	N/A	4.00
Min Delivery Delay	N/A	N/A	-11.00
Number of Delayed Deliveries	N/A	N/A	6.00
Percentage of Delayed Deliveries	N/A	N/A	40
Carrier Utilisation (%)	56.80	50.50	50.5
Idle Time (%)	43.20	49.50	49.50
Empty Travel Ratio (%)	11.70	11.60	9.7
Platooning Rate (%)	24.60	20.8	18.0
Average TUs per CU (All CUs)	1.50	1.30	1.50

The analysis evaluates how the system adapts to reduced network topology and the effectiveness of re-optimisation in maintaining service quality under structural constraints.

Performance Impact Analysis: Arc removal depresses service performance without changing demand volume: total requests remain at 15, but the *disrupted* network shows a Fulfilment Rate drop from **100%** to **87%**, with **2** unfulfilled requests. This confirms that topological losses immediately curtail feasible routings even when all other resources are intact (Table 5.12).

Re-optimisation Recovery: Re-optimisation restores **100%** fulfilment (**0** unfulfilled), demonstrating that the MILP can find alternative paths to recover service despite the missing arc. However, core efficiency does not fully return to baseline: Carrier Utilisation remains at **50.5%** (vs. **56.8%** baseline) and Idle Time stays elevated at **49.5%** (vs. **43.2%**).

Delivery Performance Analysis: The re-optimised plan achieves an *average delivery delay* of **-0.93** time units (slightly earlier than the baseline plan on average), with **6** delayed deliveries (**40%** of fulfilled jobs). Delay dispersion spans from a **max** of **+4.00** to a **min** of **-11.00** time units, indicating that some requests benefit from shorter detours while others incur moderate lateness.

Carrier Utilisation Patterns: The carrier utilisation analysis, as shown in Figure 5.7, reveals distinct patterns of workload redistribution following arc removal and re-optimisation. The visualisation demonstrates how individual Carrier Units adapt to the structural disruption, with some CUs experiencing significant changes in utilisation while others remain unaffected.

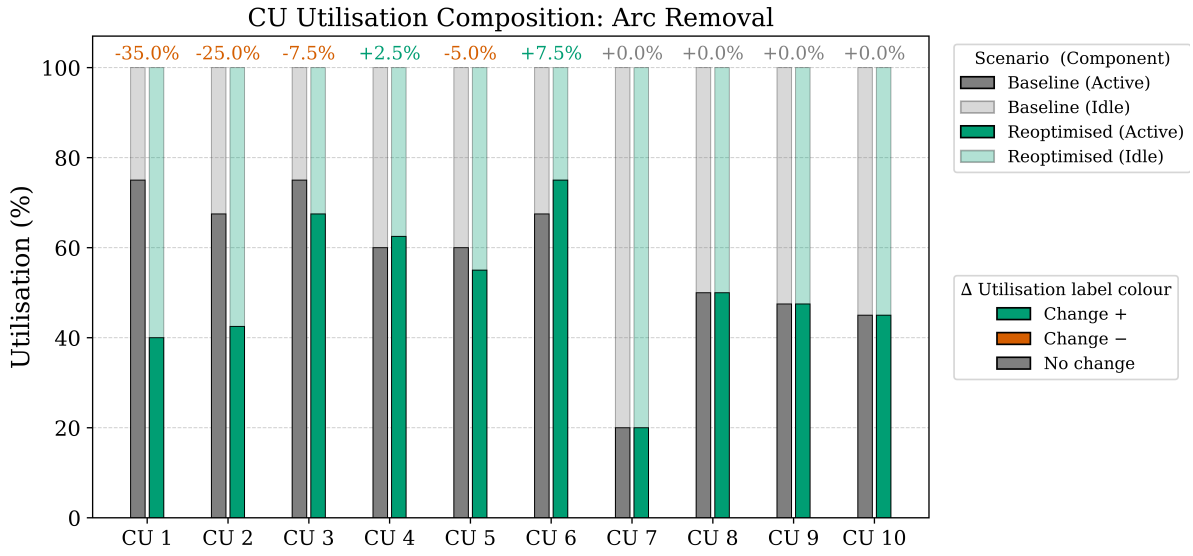

Figure 5.7: Carrier utilisation under arc disruption

Key Utilisation Trends:

- **Significant Decreases:** CU1 and CU2 experience substantial decreases in active utilisation (-35.0% and -25.0% respectively), indicating that the arc removal disrupts their primary routing patterns and reduces their operational effectiveness.
- **Moderate Decreases:** CU3 and CU5 show moderate decreases in active utilisation (-7.5% and -5.0% respectively), suggesting partial impact from the structural disruption.
- **Increases:** CU4 and CU6 demonstrate increases in active utilisation (+2.5% and +7.5% respectively), indicating that these CUs are leveraged to compensate for the reduced capacity of other carriers.
- **Unaffected CUs:** CU7, CU8, CU9, and CU10 show no change in utilisation, suggesting that their operational patterns are independent of the removed arc or that they were already operating at optimal levels.

Impact on a Specific Task: The Case of TU 6: The most direct consequence of removing arc (1,3) was on TU 6, which was originally scheduled to use this path. The following analysis details how the system adapted its plan for this single task.

Baseline Scenario Performance: In the original network configuration:

- **TU 6 Route**: Direct path $1 \rightarrow 3$
- **Travel Time**: 9 time units
- **Pickup Time**: $t = 7$
- **Delivery Time**: $t = 20$ (estimated)

Arc Removal Scenario Performance: After removing arc (1,3):

- **TU 6 Route**: Alternative path $1 \to 6 \to 5 \to 3$
- **Travel Time**: 15 time units $(5 + 4 + 6)$
- **Pickup Time**: $t = 7$ (unchanged)
- **Delivery Time**: $t = 24$

Detailed Route Analysis: The re-optimisation algorithm successfully found a viable, albeit longer, alternative route for TU 6.

Original Route (Baseline)

TU 6: Station 1 $\to$ Station 3 (Direct)
Arc $(1, 3)$: 9 time units
Total travel time: 9 time units

Alternative Route (Arc Removed)

TU 6: Station 1 $\to$ Station 6 $\to$ Station 5 $\to$ Station 3
Arc $(1, 6)$: 5 time units
Arc $(6, 5)$: 4 time units
Arc $(5, 3)$: 6 time units
Total travel time: 15 time units

Performance Metrics Comparison: The disruption forced a trade-off between completing individual tasks and maintaining overall network efficiency, ultimately leading to service failures.

Metric	Baseline	Arc Removed	Impact
Route Length (TU 6)	9 time units	15 time units	+67% increase
Delivery Time (TU 6)	$t \approx 20$	$t = 24$	+4 time units delay

Table 5.13: Performance Metrics Comparison

Operational Efficiency Metrics: Post re-optimisation, Carrier Utilisation holds at **50.5%** (below the **56.8%** baseline), and **Idle Time** at **49.5%** (above **43.2%**), signalling persistent productivity loss from the removed link. Notably, the Empty Travel Ratio *improves* to **9.7%** (vs. **11.6%** disrupted and **11.7%** baseline), implying that the solver prefers longer but more loaded itineraries over deadheading. Average TUs per CU rebounds to **1.50** (from **1.30** disrupted), matching baseline throughput per carrier.

Platooning Rate Impact: Platooning Rate declines from **24.6%** (baseline) to **18.0%** after re-optimisation—an absolute drop of **6.6** points (≈**27%** relative). Reduced connectivity limits simultaneous, co-timed paths, making coordinated platoons harder to form.

Strategic Implications: Arc removal creates *structural* limits that optimisation cannot erase: while service can be brought back to **100%** fulfilment, utilisation and platooning remain below baseline. In practice, this argues for (i) *immediate re-optimisation* to recover service, coupled with (ii) *targeted network redundancy* (loops/micro-detours) on fragile OD corridors to restore efficiency and platooning potential.

5.3.4. System Resilience Under Multiple Disruptions

This section examines the system's response to multiple concurrent and sequential disruptions that occur throughout the simulation horizon. Two distinct disruption scenarios are analysed to understand how different disruption types interact and compound their effects on system performance.

Scenario 1: Sequential Mid-Simulation Disruptions The first analysis investigates a two-phase disruption sequence where a Carrier Unit (CU) breakdown occurs mid-simulation, followed by dynamic Transport Unit (TU) additions. This scenario tests the system's ability to recover from initial capacity loss while simultaneously accommodating new demand.

Scenario 2: Triple Disruption Combination The second analysis presents a more complex scenario involving three simultaneous disruption types: (1) permanent arc removal from the network topology, (2) immediate addition of three TUs at simulation start (t=0), and (3) a CU breakdown occurring mid-simulation. This scenario evaluates system resilience under maximum disruption load, combining structural, demand, and capacity constraints.

Both scenarios utilise the re-optimisation framework to assess how the MILP model adapts to changing system conditions and quantifies the cumulative impact of multiple disruption events on key performance indicators, including fulfilment rates, delivery delays, and carrier utilisation.

Scenario 1: Sequential Mid-Simulation Disruptions:

This scenario examines a two-phase disruption sequence where a Carrier Unit breakdown occurs mid-simulation, followed by dynamic Transport Unit additions. The system's ability to recover from initial capacity loss while simultaneously accommodating new demand is evaluated as shown in 5.8.

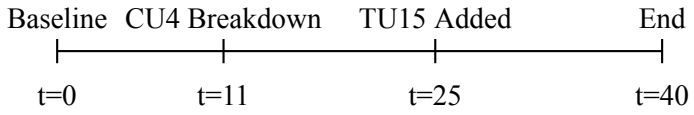

Figure 5.8: Sequential Disruption Timeline - Scenario 1

Table 5.14: KPI Summary for Randstad Case Study under 2 Disruptions

KPI Name	Baseline	Disrupted Network	Reoptimised Network
Total Requests	15	16	16
Fulfilled Requests	15	13	15
Unfulfilled Requests	0	3	1
fulfilment Rate (%)	100.00	81.00	94.00
Average Delivery Delay	N/A	N/A	-0.29
Average Relative Delay (%)	N/A	N/A	-0.50
Max Delivery Delay	N/A	N/A	8.00
Min Delivery Delay	N/A	N/A	-11.00
Number of Delayed Deliveries	N/A	N/A	6.00
Percentage of Delayed Deliveries	N/A	N/A	42.86
Carrier Utilisation (%)	56.80	56.40	57.8
Idle Time (%)	43.20	43.60	42.2
Empty Travel Ratio (%)	11.70	12.60	9.10
Platooning Rate (%)	24.60	23.70	6.30
Average TUs per CU (All CUs)	1.50	1.44	1.67

The disruption sequence creates a supply-demand mismatch where the system must serve additional demand (TU15) with reduced carrier capacity (CU4 unavailable), testing the effectiveness of the re-optimisation framework under constrained resources. The KPIs for this scenario are detailed in table 5.14.

Scenario 2: Triple Disruption Combination:
This scenario presents a more complex disruption pattern involving three simultaneous disruption types that test the system's resilience under maximum disruption load.

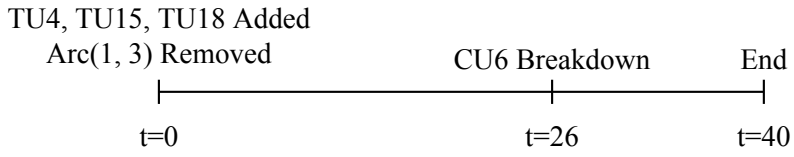

Figure 5.9: Triple Disruption Timeline - Scenario 2

This scenario creates a triple constraint scenario in which the system must operate with reduced connectivity, increased demand, and reduced capacity (CU breakdown) simultaneously.

Table 5.15: KPI Summary for Randstad Case Study under 3 Disruptions

KPI Name	Baseline	Disrupted Network	Re-optimised Network
Total Requests	15	18	18
Fulfilled Requests	15	12	15
Unfulfilled Requests	0	6	3
fulfilment Rate (%)	100.00	67.00	83.00
Average Delivery Delay	N/A	N/A	-0.64
Average Relative Delay (%)	N/A	N/A	-1.90
Max Delivery Delay	N/A	N/A	5.00
Min Delivery Delay	N/A	N/A	-9.00
Number of Delayed Deliveries	N/A	N/A	2.00
Percentage of Delayed Deliveries	N/A	N/A	14.29
Carrier Utilisation (%)	56.80	48.60	54.40
Idle Time (%)	43.20	51.40	45.60
Empty Travel Ratio (%)	11.70	9.50	6.40
Platooning Rate (%)	24.60	19.60	15.00
Average TUs per CU (All CUs)	1.50	1.33	1.67

The analysis covers two distinct scenarios: (1) a two-disruption scenario involving a Carrier Unit breakdown and dynamic TU addition, and (2) a three-disruption scenario combining arc removal, TU additions, and CU breakdown. These scenarios test the system's ability to maintain operational effectiveness under increasingly complex disruption conditions.

Two-Disruption Scenario Analysis: The two-disruption scenario demonstrates the system's response to a combination of capacity reduction and demand increase. The KPI summary reveals significant performance impacts, with the fulfilment rate dropping from 100% (Baseline) to 81% in the disrupted state, representing a 19% reduction in service quality. However, the re-optimisation process successfully recovers performance, achieving a 94% fulfilment rate and reducing unfulfilled requests from 3 to 1. Notably, the re-optimised network shows a negative average delivery delay (-0.29 time units), indicating that deliveries are completed earlier than planned, suggesting efficient resource reallocation and scheduling optimisation.As shown in table 5.14

Three-Disruption Scenario Analysis: The three-disruption scenario presents a more challenging operational environment, combining structural, demand, and capacity constraints. The disrupted state shows severe performance degradation, with fulfilment rate dropping to 67% and 6 unfulfilled requests (shown in table 5.15). The re-optimisation process demonstrates remarkable recovery capabilities, improving fulfilment rate to 83% and reducing unfulfilled requests to 3. The re-optimised network achieves a negative average delivery delay (-0.64 time units), indicating superior scheduling efficiency despite the increased complexity.

Carrier Unit Utilisation Patterns: The carrier utilisation analysis, as shown in Figure 5.10, reveals distinct patterns of resource reallocation under multiple disruptions. The visualisation demonstrates how

individual Carrier Units adapt to increasing disruption levels, with some CUs experiencing complete breakdowns (CU4 and CU6) while others are leveraged more heavily to maintain system functionality.

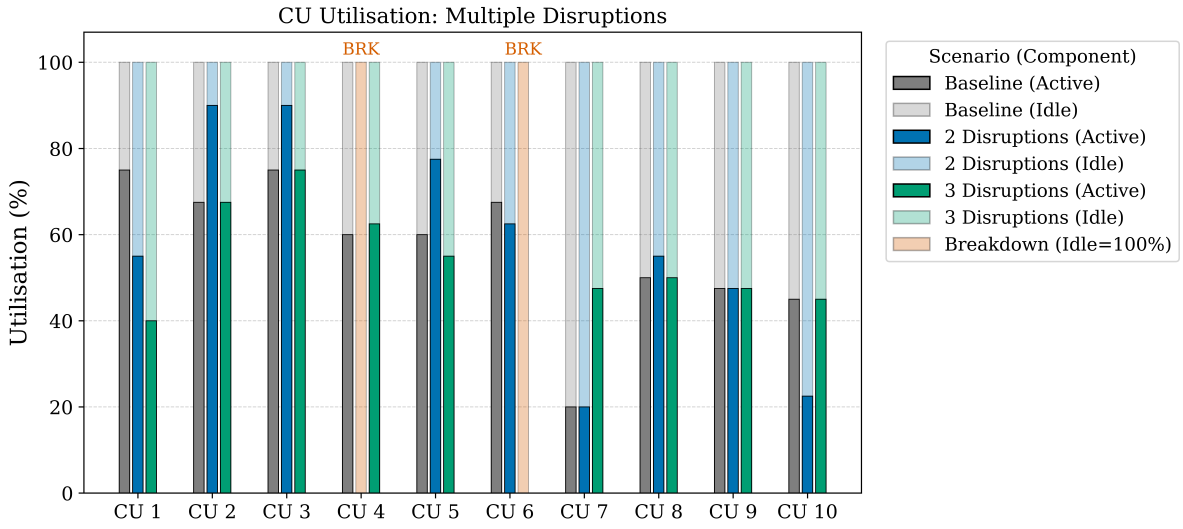

Figure 5.10: Carrier utilisation under multiple disruptions

Key Utilisation Trends:

- **Breakdown Impact:** CU4 and CU6 experience complete breakdowns in both scenarios, represented by 100% idle utilisation, significantly reducing available system capacity.
- **Compensatory Utilisation:** CU3 and CU7 demonstrate increased active utilisation under disruptions, with CU3 showing a significant increase by 15% (2 Disruptions) and CU7 transitioning from 20% active (Baseline) to 47.5% active (3 Disruptions) .
- **Reduced Utilisation:** CU1 and CU10 show decreased active utilisation under disruptions, suggesting re-routing effects and reduced overall system capacity.
- **Stable Performance:** CU9 maintains consistent utilisation across all scenarios, indicating its operational independence from the specific disruption patterns.

Operational Efficiency Analysis: The operational efficiency metrics reveal interesting patterns across both scenarios. In the two-disruption scenario, carrier utilisation increases from 56.8% (Baseline) to 57.8% (Re-optimised), while the three-disruption scenario shows a decrease from 56.8% (Baseline) to 54.4% (Re-optimised). The empty travel ratio demonstrates significant improvement in both scenarios, decreasing from 11.7% (Baseline) to 9.1% (2 Disruptions) and 6.4% (3 Disruptions), indicating more efficient routing and resource allocation.

Platooning Rate Impact: The platooning rate shows a consistent decline across both scenarios, decreasing from 24.6% (Baseline) to 6.3% (2 Disruptions) and 15.0% (3 Disruptions). This suggests that multiple disruptions create scheduling constraints that make platoon formation more challenging, as the system prioritises individual request fulfilment over coordinated carrier operations.

Comparative Performance Analysis: Comparing the two scenarios reveals that the three-disruption scenario presents more significant challenges, with lower fulfilment rates (83% vs. 94%) and higher unfulfilled requests (3 vs. 1). However, the re-optimisation process demonstrates remarkable resilience, successfully recovering from severe disruption impacts and maintaining operational effectiveness. The

negative delivery delays in both scenarios indicate that the re-optimisation framework not only recovers from disruptions but also improves upon the original baseline performance in terms of delivery timeliness.

Strategic Implications: The analysis demonstrates that while multiple disruptions create significant operational challenges, the re-optimisation framework provides effective recovery mechanisms. The system's ability to maintain negative delivery delays under complex disruption scenarios suggests that the MILP model successfully identifies more efficient scheduling solutions than the original baseline plan. This indicates that the disruption-recovery process can lead to system improvements beyond simple restoration to pre-disruption performance levels.

6

Conclusion

This thesis set out to evaluate and enhance the operational performance of autonomous pod-based railway systems by developing a framework that integrates existing Mixed-Integer Linear Programming (MILP) for optimisation with Discrete Event Simulation (DES) for dynamic evaluation. The primary objective is to assess the system's resilience and the effectiveness of a dynamic re-planning strategy when faced with operational disruptions. By modelling both simple and complex networks under various failure scenarios, this research has provided significant insights into the behaviour of such autonomous logistics systems.

This research successfully answered its guiding sub-questions. The key factors influencing carrier-TU matching, such as temporal and spatial constraints, were identified in the literature review (Chapter 2), addressing SQ1. A detailed state-event model was then developed (Chapter 3) and validated with a master example (Section A.5) to accurately capture system dynamics, answering SQ2. The core questions on strategy effectiveness and system robustness (SQ3 and SQ4) were addressed through extensive disruption analysis (Chapters 4 and 5), which consistently demonstrated that dynamic re-optimisation is a highly effective tool for mitigating disruptions.

The results from the case studies confirm the value of the integrated DES-MILP approach. The analysis of the selected Key Performance Indicators (KPIs) across the baseline, disrupted, and re-optimised scenarios has yielded several key findings:

- **Dynamic Re-planning is a Highly Effective Recovery Tool:** The results consistently confirm that re-optimising the plan at the moment of a disruption significantly improves performance. In the Toy Case, re-optimisation was critical, restoring the **Fulfillment Rate** from a disrupted 25% back to 75%. This demonstrates the fundamental value of the adaptive framework in overcoming service failures.
- **Statistical Reliability and Variability of Toy Case KPIs:** Across all KPIs, the optimized system is more efficient and consistent, while disruption leads to lower fulfillment and utilization, higher idle time, and the loss of platooning. The standard deviations reveal greater variability in fulfillment rate, especially under disruption, while carrier utilization and empty travel ratio remain comparatively stable. At the same time, the 95% confidence intervals are narrow for all KPIs, confirming that despite run-to-run fluctuations, the reported mean values are statistically reliable.
- **Resilience Comes with Quantifiable Trade-offs:** To maintain service levels, the system often sacrifices operational efficiency. For instance, while recovering from the CU breakdown in the Randstad Case, the **Carrier Utilisation** increased from 56.8% to 65.3%, but this came at the cost of a lower **Platooning Rate**, which fell from 24.6% to 17.8%. This highlights a clear trade-off between service completion and logistical efficiency.

- **System performance is limited by network topology (arc removal):** When arc $(1, 3)$ is blocked (from $t = 7$ to $t = 17$), the disrupted plan achieves 87% fulfilment (13/15), but re-optimisation restores 100% (15/15) by rerouting (see Table 5.12). Efficiency does not fully recover: **Carrier utilisation** falls from 56.8% (baseline) to 50.5% (re-optimised) and **idle time** rises from 43.2% to 49.5%; the **empty-travel ratio** improves 11.7% $\rightarrow$ 9.7%; the **platooning rate** declines 24.6% $\rightarrow$ 18.0%. This shows topology losses can cap efficiency and coordinated movements even when service is fully recovered.
- **Time Window Flexibility is a Key Enabler of Resilience:** The analysis of time window flexibility in the Randstad Case demonstrates that even a small amount of flexibility (e.g., 10 time units) can significantly improve the fulfillment rate after a disruption. This highlights the importance of incorporating flexibility into service level agreements to enhance system robustness.
- **Early Intervention is More Effective than Reactive Measures:** The time bracket analysis in the Randstad Case shows that adding new transport units early in the planning horizon (t=0) leads to better performance than adding them later (t=10 or t=20). This suggests that proactive demand management and early integration of new requests are crucial for maintaining system efficiency.

6.1. Industrial and Policy Insights

This study underscores the importance of aligning policy frameworks and industrial practices to enhance resilience in long-haul autonomous rail freight operations. On the policy side, the results demonstrate that allowing moderate delivery window flexibility enables recovery of high fulfilment rates after disruptions, though with trade-offs in punctuality. Regulators should therefore design mechanisms that incentivise flexibility through pricing and contractual caps. The larger advantages of implementing early transport units over mid-horizon additions point out that booking cut-offs and incentives for early commitment will improve overall service robustness. Since service re-optimisation always recovers the service level under single and multiple disruptions, digital decision-support systems, like the DES-MILP integrations, should be considered vital resilience assets. However, the lack of progress in the cases of arc removal justifies public spending on redundant infrastructure, such as alternative routes and loops. In the end, practice-based monitoring of the implementation of resilience principles, delays, empty travel, utilisation, and platooning of the reporting system should be incorporated within the regulatory reporting framework to provide visibility and standards between operators. These observations align with previous studies on circulation and redundancy in rail systems [16].

From an industrial perspective, the findings provide operational insight for operators and equipment builders. The most effective operational doctrine, which ensures recovery even during complex disruptions, was the lock-in and rolling re-planning mechanism. This can be institutionalised as an autonomous freight standard control room procedure. The observed punctuality versus fulfilment trade-offs give rise to differentiated service level agreements, premium contracts with sharp fulfilment time windows and low resilience, and flexible contracts with higher disruption fulfilment guarantees. This evidence in support of early TU insertions also underlines the importance of demand-shaped TU, for example, pricing policies that pay for early bookings. At the fleet management level, the recovery of the disruption as a carrier that is unevenly used indicates the need for standby CU resources, plus the CU is placed in an active way to conserve platoon opportunities and reduce unnecessary travel. Also, the enduring consequences of removing arcs indicate the need for closer cooperation with infrastructure managers for the design of detour loops and micro-routes that protect critical OD pairs. These industrial implications align with relocation and circulation strategies in shared-mobility and transport literature [19], reinforcing the importance of flexible service portfolios and strategic fleet positioning for resilient and efficient operations.

6.2. Contributions

This research makes several key contributions to the field of autonomous logistics and the Pods4Rail project. The primary contribution is the development and successful implementation of an integrated, event-driven re-optimisation framework that bridges the gap between static planning and dynamic execution. By using the DES to simulate a plan and feed real-time state information back to the MILP upon a disruption, this work provides a practical methodology for enabling adaptive, resilient operations.

6.3. Limitations and Future Research

While this study provides a robust framework, it is subject to several limitations that open avenues for future research:

- The analysis was conducted using synthetic data. Future work should aim to validate the models using real-world operational data to enhance the accuracy of the findings.
- The set of disruptions, while representative, was limited. Further research could explore a wider range of stochastic events, such as variable travel times, station capacity blockages, or more complex carrier failure modes.
- A limitation of the current study is its relatively short time horizon of 40 time stamps. Expanding this planning period in future work is essential for scaling the analysis to larger railway networks and incorporating more complex, dynamic scenarios

Ultimately, this thesis has demonstrated that the integration of optimisation and simulation provides a powerful tool for designing and managing resilient autonomous transportation systems. The proposed framework not only allows for a detailed evaluation of system performance but also provides the means to actively enhance it in the face of the uncertainty inherent in all real-world logistics operations.

References

[1] Maersk, *Intermodal transportation: What is it and how does it work?* Maersk.com, Sep. 2024. [Online]. Available: https://www.maersk.com/logistics-explained/transportation-and-freight/2024/09/06/intermodal-transportation.

[2] A. Agamez-Arias and J. Moyano-Fuentes, "Intermodal transport in freight distribution: A literature review," *Transport Reviews*, vol. 37, no. 6, pp. 782–807, Mar. 2, 2017. doi: 10.1080/01441647.2017.1297868.

[3] Pods4Rail, *Homepage 01 - Pods4Rail*, Dec. 2024. [Online]. Available: https://pods4rail.eu/.

[4] X. Liao, J. Han, S. Mahnam, and A. Paz Martinez, "Unlocking the potentials of modularity in railways, a heuristic framework for pods scheduling," in *27th IEEE International Conference on Intelligent Transportation Systems, ITSC 2024*, IEEE - Institute of Electrical and Electronics Engineers, 2024.

[5] A. Di Febbraro, N. Sacco, and M. Saeednia, "One-way car-sharing profit maximization by means of user-based vehicle relocation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 2, pp. 628–641, May 28, 2018. doi: 10.1109/tits.2018.2824119.

[6] A. Di Febbraro, N. Sacco, and M. Saeednia, "One-way carsharing," *Transportation Research Record Journal of the Transportation Research Board*, vol. 2319, no. 1, pp. 113–120, Jan. 1, 2012. doi: 10.3141/2319-13.

[7] X. Liao, J. Han, and M. Saeednia, "Modular vehicle routing on railways: Opportunities for intermodality," 2024.

[8] E. M. Cepolina, A. Farina, and A. Pratelli, *Car-sharing relocation strategies: a state of the art*. Jun. 20, 2014, pp. 109–120. doi: 10.2495/978-1-84564-908-1/010.

[9] J. Chakraborty, D. Pandit, F. Chan, and J. (Xia, "A review of ride-matching strategies for ridesourcing and other similar services," *Transport Reviews*, vol. 41, no. 5, pp. 578–599, Sep. 3, 2021. doi: 10.1080/01441647.2020.1866096.

[10] A. Tafreshian, N. Masoud, and Y. Yin, "Frontiers in service science: Ride matching for peer-to-peer ride sharing: A review and future directions," *Service Science*, vol. 12, no. 2, pp. 44–60, Jun. 2020. doi: 10.1287/serv.2020.0258.

[11] Y. H. Wu, L. J. Guan, and S. Winter, *Peer-to-Peer shared ride systems*. Jan. 1, 2008, pp. 252–270. doi: 10.1007/978-3-540-79996-2_14.

[12] X. Bei and S. Zhang, "Algorithms for trip-vehicle assignment in ride-sharing," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 25, 2018. doi: 10.1609/aaai.v32i1.11298.

[13] R. Ma, L. Yao, L. Song, and M. Jin, "A novel algorithm for peer-to-peer ridesharing match problem," *Neural Computing and Applications*, vol. 31, pp. 247–258, S1 Jan. 2019. doi: 10.1007/s00521-018-3733-5.

[14] J. Chang, M. Yu, S. Shen, and M. Xu, "Location design and relocation of a mixed car-sharing fleet with a CO_2 emission constraint," *Service Science*, vol. 9, no. 3, pp. 205–218, Sep. 2017. doi: 10.1287/serv.2017.0178.

[15] N. Milosavljević, D. Teodorović, V. Papić, and G. Pavković, "A fuzzy approach to the vehicle assignment problem," *Transportation Planning and Technology*, vol. 20, no. 1, pp. 33–47, Sep. 1996. doi: 10.1080/03081069608717578.

[16] A. Alfieri, R. Groot, L. Kroon, and A. Schrijver, "Efficient circulation of railway rolling stock," *Transportation Science*, vol. 40, no. 3, pp. 378–391, Jul. 20, 2006. doi: 10.1287/trsc.1060.0155.

[17] M. Peeters and L. Kroon, "Circulation of railway rolling stock: a branch-and-price approach," *Computers & Operations Research*, vol. 35, no. 2, pp. 538–556, Feb. 2007. doi: 10.1016/j.cor.2006.03.019.

[18] D. Canca, M. Sabido, and E. Barrena, "A rolling stock circulation model for railway rapid transit systems," *Transportation research procedia*, vol. 3, pp. 680–689, Jan. 1, 2014. doi: 10.1016/j.trpro.2014.10.047.

[19] S. Illgen and M. Höck, "Literature review of the vehicle relocation problem in one-way car sharing networks," *Transportation Research Part B: Methodological*, vol. 120, pp. 193–204, Feb. 2019. doi: 10.1016/j.trb.2018.12.006.

[20] G. Alfian, J. Rhee, M. Ijaz, M. Syafrudin, and N. Fitriyani, "Performance analysis of a forecasting relocation model for one-way carsharing," *Applied Sciences*, vol. 7, no. 6, p. 598, Jun. 9, 2017. doi: 10.3390/app7060598.

[21] M. Clemente, M. P. Fanti, A. M. Mangini, and W. Ukovich, "The vehicle relocation problem in car sharing systems: Modeling and simulation in a petri net framework," in *Application and Theory of Petri Nets and Concurrency*, J.-M. Colom and J. Desel, Eds., red. by D. Hutchison et al., vol. 7927, Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 250–269. doi: 10.1007/978-3-642-38697-8_14.

[22] S. Weikl and K. Bogenberger, "Relocation strategies and algorithms for free-floating car sharing systems," *IEEE Intelligent Transportation Systems Magazine*, vol. 5, no. 4, pp. 100–111, 2013. doi: 10.1109/MITS.2013.2267810.

[23] A. Ait-Ouahmed, D. Josselin, and F. Zhou, "Relocation optimization of electric cars in one-way car-sharing systems: Modeling, exact solving and heuristics algorithms," *International Journal of Geographical Information Science*, vol. 32, no. 2, pp. 367–398, Feb. 1, 2018. doi: 10.1080/13658816.2017.1372762.

[24] M. Lai, Q. Hu, Y. Liu, and Z. Lang, "A rolling-horizon decision framework for integrating relocation and user flexibility in one-way electric carsharing systems," *Transportation Research Part C: Emerging Technologies*, vol. 144, p. 103 867, Nov. 2022. doi: 10.1016/j.trc.2022.103867.

[25] A. G. Kek, R. L. Cheu, Q. Meng, and C. H. Fung, "A decision support system for vehicle relocation operations in carsharing systems," *Transportation Research Part E: Logistics and Transportation Review*, vol. 45, no. 1, pp. 149–158, Jan. 2009. doi: 10.1016/j.tre.2008.02.008.

[26] G. Alfian, J. Rhee, and B. Yoon, "A relocation simulation model for one-way carsharing service," in *2014 IEEE International Conference on Industrial Technology (ICIT)*, Busan, South Korea: IEEE, Feb. 2014, pp. 718–723. doi: 10.1109/ICIT.2014.6895020.

[27] A. Varga, "Discrete event simulation system," in *Proc. of the European Simulation Multiconference (ESM'2001)*, vol. 17, 2001.

[28] J.-F. Cordeau and G. Laporte, "The dial-a-ride problem: Models and algorithms," *Annals of Operations Research*, vol. 153, no. 1, pp. 29–46, Jun. 6, 2007. doi: 10.1007/s10479-007-0170-8.

[29] F.-S. Hsieh, “Comparison of a hybrid firefly–particle swarm optimization algorithm with six hybrid firefly–differential evolution algorithms and an effective cost-saving allocation method for ridesharing recommendation systems,” *Electronics*, vol. 13, no. 2, p. 324, Jan. 11, 2024. doi: 10.3390/electronics13020324.

A

Appendix

A.1. Summary of Literature

Table A.1: Summary of Literature Relevant to the Project

AP - Assignment problem, RP - Repositioning problem, RA - Railway assignment problem

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RI	Findings	Relevance to My Work
Tafreshian et al. (2020) [10]	Ride-matching in P2P systems.	Optimise matching, routing, and scheduling to maximise efficiency, reduce costs, and improve service quality.	MILP, ILP, bipartite graphs, general graphs	Exact algorithms, approximation algorithms (e.g., scaling algorithms), and heuristic methods (e.g., clustering, genetic algorithms).	On-demand and one-time trip requests.	Tight time windows for drivers and riders, limited vehicle capacity, and spatial-temporal constraints.	✓	-	-	P2P ride-sharing systems can reduce single-occupancy vehicles, alleviate traffic congestion, and lower transportation costs.	Provides insights into matching strategies applicable to carrier-TU assignments.
Wu et al. (2008) .[11]	Matching clients (riders) with hosts (drivers) in a peer-to-peer shared ride system.	Optimise ride-sharing by matching clients with hosts based on preferences (e.g., travel time, fare, convenience) using local communication strategies.	Multi-agent simulation model with heterogeneous agents (clients and hosts).	Heuristic algorithms and multi-criteria optimisation.	On-demand and ad-hoc trip requests.	Limited communication range, local knowledge, vehicle capacity, and spatial-temporal constraints.	✓	-	-	Demonstrated effectiveness of decentralized matching algorithms.	Focuses on matching based on spatial and temporal constraints, similar to carrier-TU matching.

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Ma et al. (2019) [13]	Single-driver multiple-rider (SDMR) ridesharing matching problem	Optimise ride-matching to achieve stable or nearly stable matches, balancing system-wide benefits and individual cost savings	Two-sided stable matching model, MILP.	Recursive algorithm for feasible rider sets, delete operator for preference lists, and heuristic algorithms.	On-demand and ad-hoc trip requests	Time windows, vehicle capacity, spatial-temporal constraints, and cost feasibility.	✓	-	-	The recursive algorithm and delete operator improve computational efficiency, making it suitable for large-scale ridesharing systems.	Focuses on matching based on spatial and temporal constraints, similar to carrier-TU matching
Alfieri et al. [16]	Efficient circulation of railway rolling stock	Minimise fixed and variable costs while ensuring passenger demand is met and shunting constraints are respected	Integer multicommodity flow model with transition graphs for train compositions.	Heuristic algorithm with node elimination, disconnection elimination, and CPLEX solver	Passenger demand	Shunting constraints, maximum train length, seat capacity requirements, and train unit order in compositions	✓	✓	✓	The proposed approach efficiently reduces the number of train units and optimises their circulation, ensuring passenger demand is met while minimising costs.	Integer multicommodity flow model and transition graphs to optimise the assignment and relocation of carriers for TUs, ensuring carriers are available at the right time and location for seamless intermodal transitions and platoon formations.

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Cordeau & Laporte (2007) [28]	Dial-a-Ride Problem (DARP)	Minimise cost while accommodating as many users as possible under constraints.	MILP, Three-index and two-index formulations.	Exact algorithms (branch-and-cut), heuristics (tabu search, genetic algorithms).	Static, Dynamic	Vehicle capacity, time windows, maximum ride time, route duration, precedence constraints.	✓	-	-	Heuristics exist for static DARP, capable of solving instances with hundreds of users.	Focuses on matching vehicles to passengers/goods with temporal and spatial constraints.
Bei & Zhang (2018) [12]	Ride-Sharing Assignment	Minimise total driving distance while assigning two requests to one car.	Combinatorial optimisation problem (NP-hard).	Approximation algorithm with 2.5x optimal cost guarantee.	Static	Each car serves exactly two requests, distance constraints (Manhattan/Euclidean).	✓	-	-	The algorithm has an approximation ratio of 2.5 in the worst case, but empirical results show a much better ratio (around 1.1-1.2). The ratio improves as the number of requests and drivers increases.	Focuses on matching vehicles to riders with temporal and spatial constraints.
Peeters & Kroon [17]	Efficient circulation of railway rolling stock	Minimize seat shortages, shunting operations, and carriage kilometers while ensuring passenger demand is met	Integer programming model with Dantzig-Wolfe decomposition	Branch-and-price algorithm with column generation	Passenger demand	Maximum train length, seat capacity requirements, and train unit order in compositions	✓	✓	✓	The branch-and-price approach outperforms CPLEX in solving real-life instances, providing efficient rolling stock circulations with reduced seat shortages and shunting operations	The branch-and-price algorithm can be adapted to dynamically assign carriers to TUs at intermodal terminals.

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Milosavljević et al. (1996)[15]	Vehicle Assignment Problem	Assign vehicles to transportation requests in the ”best possible way”.	a Fuzzy logic-based model to formalise the dispatcher’s knowledge and rules.	Heuristic algorithm with fuzzy logic for preference calculation.	Static	Vehicle capacity, distance, number of available vehicles, and dispatcher’s subjective preferences.	✓	-	-	The fuzzy logic-based model outperforms manual dispatcher assignments, especially in terms of time efficiency. The model achieves equal or better results compared to an experienced dispatcher.	Offers a method to manage uncertainties in vehicle assignment, highly relevant to matching carriers to TUs.
Illgen & Höck (2019) [19]	Vehicle Relocation Problem (VReP)	Optimise vehicle relocations to balance supply and demand in one-way car-sharing networks.	Mixed-Integer Programming (MIP), Simulation, Multi-stage Approaches.	Optimisation (MIP), Simulation (discrete event, agent-based), Heuristics.	Static, Dynamic,	Vehicle capacity, station capacity, demand variability, relocation costs, fleet size.	✓	✓	-	One-way car-sharing systems face vehicle imbalance, requiring relocations. MIP models are optimal but computationally expensive. Simulation models handle stochastic demand better. Multistage approaches combine optimisation and simulation for better results.	The challenges and solutions discussed are analogous to the relocation needs in autonomous pod-based rail systems.

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Clemente et al. (2013) [21]	Vehicle imbalance in car sharing systems	To improve system performance by balancing vehicle distribution using user-based incentives	Timed Petri Net (TPN) framework, UML activity diagrams, and simulation models	Simulation in MATLAB	Dynamic (real-time user demand)	Maximum waiting time (10 mins)- Maintenance constraints (1-hour or 8-hour service)- Limited parking capacity	-	✓	-	Real-time monitoring and incentives improve system performance.	Offers strategies and modelling techniques relevant to managing static demand and carrier relocations in autonomous pod-based rail systems.
Canca et al. (2014) [18]	Rolling stock circulation optimisation for Rapid Transit Systems.	Minimize the fleet size required to cover all train schedules, ensure a balanced weekly operation for maintenance efficiency	MILP	Optimization solvers	Static	Vehicle and carriage rest location	✓	✓	✓	The proposed model can minimize empty train movements and balance the weekly kilometers covered by each train unit, effectively managing maintenance and operational costs	Minimizing empty movements can be adapted to efficiently manage the assignment and relocation of carriers.

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Weikl & Bogenberger (2012) [22]	Vehicle imbalance in free-floating car sharing systems	To optimize vehicle distribution and improve system efficiency using relocation strategies	Two-step algorithm (offline demand clustering and online optimization)	Rolling horizon method (user-based), Three-phase decision support system (operator-based), Two-step algorithm (integrated)	Dynamic (real-time user demand)	Cost of relocation vs. benefits, Parking capacity	-	✓	-	Demonstrated the effectiveness of strategic relocation planning and real-time adjustments in managing vehicle fleets efficiently.	These strategies provide a framework for managing static demand carrier relocation in autonomous pod-based rail systems, using predictive and real-time data-driven approaches to ensure optimal placement
Cepolina et al. (2012)[8]	Vehicle imbalance in car sharing systems	To address vehicle imbalance in car sharing systems and improve system efficiency	MILP, DES	Optimisation algorithms, simulation models	Dynamic (real-time user demand)	High and low critical thresholds. Parking capacity	-	✓	-	Third-generation systems reduce the need for relocation by increasing vehicle availability.	The automated relocation strategies detailed can be adapted to manage the non-dynamic, scheduled relocations of carriers in our rail system, particularly in pre-planning deployments and real-time adjustments.
Ait-Ouahmed, Josselin, and Zhou [23]	Vehicle imbalance in one-way electric car-sharing systems	To optimise vehicle relocation to maximise user satisfaction and minimise operational costs	MILP	Greedy Algorithm, Tabu Search	Dynamic (real-time user demand), Static	Vehicle capacity at stations , Recharging time constraints	-	✓	-	Tabu search provides near-optimal solutions with a 10% gap compared to MILP. The greedy algorithm is efficient for large-scale problems.	Heuristic methods like Tabu search can be applied to efficiently manage carrier relocations in our rail system.

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Chang et al.[14]	Car-sharing fleet location and relocation	Maximise profit from renting cars while minimising relocation and maintenance costs, subject to budget and CO emission constraints	ILP	Gurobi solver for ILP optimisation.	One-way and round-trip demand	Arc capacity, FCFS (First-Come, First-Served).	-	✓	-	High car utilisation, low demand losses, and denied trips. CO emission limits reduce profit, but the high demand for energy-efficient cars compensates.	Similar strategies can be adapted for scheduling and relocating carriers in our pod-based rail system to minimise environmental impact and operational costs.
Chakraborty et al.[9]	Ride-matching for rides-ourcing and similar services	Optimise the allocation of trips to drivers and users, considering spatio-temporal demand and supply variations.	Classification-based and simulation-based approaches for search strategies; optimal and simulation-based approaches for assignment strategies.	Various algorithms, including k-means clustering, Dijkstra's least-cost path, and multi-objective optimisation.	One-way and round-trip demand	Passenger waiting time, driver idle time	✓	✓	-	Search strategies guide riders and drivers to suitable locations, while assignment strategies ensure trip allotment.	The dynamic and real-time matching algorithms can be adapted to optimise the static, scheduled relocations of carriers in our autonomous rail system.

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Hsieh[29]	Ride-sharing optimization and cost allocation	Optimize ridesharing cost savings and allocate savings among stakeholders effectively	MILP	Hybrid Firefly-PSO (FPSO) and Firefly-DE (FDEi) algorithms	Static	Demand and supply constraints, cost-saving constraints, drivers' single winning bid constraint, binary decision variables	✓	-	-	The FPSO's(Firefly Particle Swarm Optimization) ability to quickly converge and its robustness in solution quality make it a viable option compared to the traditional PSO (Particle Swarm Optimization)	These optimization techniques can be adapted to enhance the scheduling and routing efficiency of carriers in autonomous rail systems.
Kek et al.[25]	Vehicle relocation in carsharing systems	Develop a decision support system to optimise vehicle relocation in carsharing systems.	MILP	Three-phase OTS (Optimisation-Trend-Simulation) decision support system	Static	Vehicle availability, station capacity	✓	✓	-	Improved zero-vehicle-time (ZVT) by 4.6% to 13.0%, maintained low full-port-time (FPT), and reduced the number of relocations (NR) by 37.1% to 41.1%.	The Shortest Time and Inventory Balancing techniques can be adapted to optimise the dynamic/static allocation of carriers in autonomous rail systems.

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Liao et al. [7]	PDMVRP - with platooning	To enhance rail-based intermodal freight systems' flexibility and integration into mobility services by efficiently routing modular vehicles, ultimately reducing costs and improving capacity utilisation in railway environments.	MILP	-	Static	Time windows for pick-ups and deliveries.	✓	-	✓	The proposed MILP model effectively reduces transportation costs and improves railway capacity utilisation through optimised scheduling and platooning of modular vehicles.	Forms the foundation for this project topic.
Minghui Lai et al.[14]	One-way electric car-sharing system	Maximise profit by optimising car relocation, user flexibility, and EV charging in real-time.	MILP	Iterated Local Search (ILS) heuristic	Dynamic, uncertain	Parking capacity, battery charging	✓	✓	-	Rolling-horizon decision framework, particularly the iterated local search algorithm, is highly efficient and outperforms both the Particle Swarm Optimisation (PSO) algorithm and a First-Come, First-Served (FCFS) greedy policy.	The iterated local search algorithm and dynamic reoptimization can enhance carrier scheduling in autonomous rail systems

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
Alfian et al.[20]	One-way car-sharing system imbalance	Improve system utilisation and reservation acceptance ratio by forecasting relocation.	Discrete Event Simulation (DES) model	Multilayer Perceptron (MLP) for forecasting	Dynamic	Car availability, and customer reservation constraints.	✓	✓	-	Forecasting relocation outperforms traditional relocation methods in terms of system utilisation and reservation acceptance ratio. However, it incurs higher relocation costs.	The predictive relocation strategies can be adapted to manage carrier distribution in autonomous rail-based systems.
Ganjar Alfian et al. [26]	One-way car-sharing system imbalance	Minimise relocation cost while maintaining customer satisfaction and vehicle utilisation.	Discrete Event Simulation (DES) model	Periodically triggered relocation (every 6 hours)	Dynamic, uncertain	Parking capacity, car availability,	✓	✓	-	The proposed 6-hourly relocation model reduces relocation costs by 30% compared to static shortest-time relocation but slightly decreases utilisation and acceptance ratios.	The Periodic Relocation technique can be adapted for scheduled carrier redistribution in rail-based systems to maintain optimal carrier availability.
Liao et al. [4]	Modular vehicle scheduling	To minimise the makespan and optimise the use of railway infrastructure by scheduling modular vehicles efficiently.	Heuristic framework	Heuristic for carrier scheduling	Static	Availability of carriers, Time constraints for pickup and delivery	✓	-	-	The framework effectively reduces makespan and optimises capacity utilisation, particularly effective with larger problem sizes.	Forms a foundation for this project

Continued on next page

Table A.1 continued from previous page

Author	Problem	Objective	Model	Algorithm	Demand Type	Key Constraints	AP	RP	RA	Findings	Relevance to the project
This paper	Carrier-TU assignment and relocation in autonomous pod-based rail systems.	Optimise carrier-TU matching and relocation to improve efficiency.	MILP integrated with DES	Possibly Rolling Horizon, Greedy, Tabu Search, Iterated Local Search	Static (scheduled demand)	Temporal constraints (pickup/drop-off times), spatial constraints (station locations), carrier availability, platooning (secondary factor)	✓	✓	✓	Potentially the proposed models would reduce operational costs, improve turnaround times, and ensure high service reliability. Integration of MILP and DES with Rolling Horizon provides a robust framework for dynamic decision-making.	Forms a foundation for this project

A.2. Extract of DES - Master Example

Table A.2: Extract of DES trace for the master example

Time	Request	CU	Old State	Event	New State	Arc	Trip
Repositioning platoon from 3→4 (t=2→3):							
2	–	CU1	C5 (At Station, Waiting)	E10 (CU departs station empty)	C7 (Repositioning – Platoon)	3→4	Relocation
2	–	CU2	C5 (At Station, Waiting)	E10 (CU departs station empty)	C7 (Repositioning – Platoon)	3→4	Relocation
3	–	CU1	C7 (Repositioning – Platoon)	E3 (CU arrives at station)	C5 (At Station, Waiting)	–	Relocation
3	–	CU2	C7 (Repositioning – Platoon)	E3 (CU arrives at station)	C5 (At Station, Waiting)	–	Relocation
Pickup and transport platoon from 4→5 (t=3→4):							
3	1	CU1	C5 (At Station, Waiting)	E4 (CU loads TU1)	C1 (TU Loaded to CU)	–	Transport
3	2	CU2	C5 (At Station, Waiting)	E4 (CU loads TU2)	C1 (TU Loaded to CU)	–	Transport
3	1	CU1	C1 (TU Loaded to CU)	E5 (TU–CU depart pickup station)	C3 (In Transit – Platoon)	4→5	Transport
3	2	CU2	C1 (TU Loaded to CU)	E5 (TU–CU depart pickup station)	C3 (In Transit – Platoon)	4→5	Transport
4	1	CU1	C3 (In Transit – Platoon)	E8 (TU–CU arrive delivery station)	C4 (At Delivery Station)	4→5	Transport
4	1	CU1	C4 (At Delivery Station)	E9 (CU unloads TU1 at delivery station)	C5 (At station, Waiting)	–	Transport
Mid-route split and solo continuation from 5→6 (t=5→6):							
5	2	CU2	C3 (In Transit – Platoon)	E7 (CU leaves platoon)	C2 (In Transit – Solo)	5→6	Transport
6	2	CU2	C2 (In Transit – Solo)	E8 (TU–CU arrive delivery station)	C4 (At Delivery Station)	5→6	Transport
6	2	CU2	C4 (At Delivery Station)	E9 (CU unloads TU2 at delivery station)	C5 (At Station, Waiting)	5→6	Transport

A.3. Optimization Model Overview

In order to assign transport requests to carrier units and to plan their exact pickup, delivery, and repositioning moves (including platooning decisions), a mixed-integer linear program (MILP) is solved as a first step. Although the detailed formulation is outside the scope of this thesis, the MILP performs three main tasks:

1. **Assignment:** Selects for each transport request which carrier unit (CU) will serve it.
2. **Timing:** Determines the exact pickup and delivery times within user-specified windows.
3. **Routing & Platooning:** Specifies, for each CU, the sequence of network arcs to traverse (possibly with loaded or empty moves) and identifies which arcs will be travelled in a platoon.

A.3.1. Key Inputs

- **Network topology:** Set of stations, travel times on each bidirectional link, and station capacity limits.
- **Carrier fleet:** IDs and their initial station locations.
- **Transport requests:** For each request, an origin and destination station, a fixed (precomputed) route of intermediate stations, and allowable pickup/delivery time windows.
- **Operation parameters:** Fixed loading and unloading durations, and any minimum platoon-size requirements or other business rules.

A.3.2. Key Outputs

- **Assignment decisions:** Which CU serves each request.
- **Schedule:** Exact pickup and delivery times for every transport request.
- **CU itineraries:** Ordered lists of station-to-station moves (with departure times) for each CU, marked as either *loaded* (transport) or *empty* (repositioning), and the subset of moves executed in platoon.
- **TU movements:** For each transport request, the specific arc-by-arc time steps on which it is carried by its assigned CU.

These outputs are then *parsed* into two Python data structures (`tu_data` and `cu_data`) which serve as the exact inputs for our discrete-event simulation in Section 4.2.1.

Table A.3: KPI Comparison for Randstad CU Breakdown (Flexible Window Analysis)

KPI Name	**Baseline**	**Disrupted Network**	**Time window flexibility = 0**	**Time window flexibility = 10**	**Time window flexibility = 40**
Total Requests	15	15	15	15	15
Fulfilled Requests	15	13	13	14	15
Unfulfilled Requests	0	2	2	1	0
Fulfillment Rate (%)	100.00	92.90	92.90	93.30	100.00
Average Delivery Delay	N/A	0.00	0.08	3.79	4.20
Average Relative Delay (%)	N/A	0.00	0.20	15.80	21.30
Max Delivery Delay	N/A	0.00	9.00	10.00	23.00
Min Delivery Delay	N/A	0.00	-4.00	-4.00	-7.00
Number of Delayed Deliveries	N/A	0.00	2.00	10.00	9.00
Percentage of Delayed Deliveries	N/A	0.00	15.38	71.43	60.00
Carrier Utilization (%)	50.50	50.00	51.80	52.00	57.50
Idle Time (%)	49.50	50.00	48.20	48.00	42.50
Empty Travel Ratio (%)	9.70	10.30	11.40	11.50	9.90
Platooning Rate (%)	18.00	19.20	11.80	15.70	13.20
Average TUs per CU (All CUs)	1.50	1.44	1.30	1.40	1.50

A.4. KPIs for Toy Case

The Toy Case analysis provides a quantitative overview of system performance under both optimised and disrupted conditions. Key performance indicators (KPIs) were evaluated across multiple runs to capture trends in efficiency, reliability, and delay behaviour. Detailed results for both cases are presented

in tables A.4 and A.5, including mean values, variability (standard deviation), and confidence intervals for all KPIs.

Table A.4: KPI Statistics for Optimized System (n = 200 runs)

KPI	Mean	Std. Dev.	Median	95% CI Low	95% CI High
Fulfillment Rate (%)	71.0	16.8	80.0	68.5	73.5
Average Relative Delay (%)	24.5	18.7	18.2	21.8	27.3
Carrier Utilization (%)	35.8	11.0	36.7	34.2	37.4
Idle Time (%)	64.2	11.0	63.3	62.6	65.8
Empty Travel Ratio (%)	12.4	10.6	9.8	10.8	14.0
Platooning Rate (%)	8.1	16.3	0.0	5.7	10.5
Average Delivery Delay	3.37	2.02	3.2	3.07	3.67
Max Delivery Delay	7.03	2.56	8.0	6.65	7.40
Min Delivery Delay	-0.18	1.63	0.0	-0.42	0.06
Number of Delayed Deliveries	2.36	1.06	2.0	2.21	2.52
Percentage of Delayed Deliveries	63.6	25.9	66.7	59.7	67.4

Table A.5: KPI Statistics for Disrupted System (n = 200 runs)

KPI	Mean	Std. Dev.	Median	95% CI Low	95% CI High
Fulfillment Rate (%)	65.2	24.3	60.0	61.8	68.6
Average Relative Delay (%)	3.6	7.1	0.0	2.6	4.5
Carrier Utilization (%)	28.9	10.0	28.9	27.5	30.3
Idle Time (%)	71.1	10.0	71.1	69.7	72.5
Empty Travel Ratio (%)	9.8	7.5	8.8	8.7	10.8
Platooning Rate (%)	0.0	0.0	0.0	0.0	0.0
Average Delivery Delay	0.44	0.81	0.0	0.33	0.56
Max Delivery Delay	1.25	2.08	0.0	0.96	1.54
Min Delivery Delay	0.03	0.24	0.0	-0.00	0.06
Number of Delayed Deliveries	0.44	0.74	0.0	0.33	0.54
Percentage of Delayed Deliveries	13.3	22.3	0.0	10.2	16.4

A.5. Master Example: Two-CU, Two-TU Scenario

To demonstrate that the DES operates correctly across all possible events (loading, departure, platooning, splitting, unloading, etc.), a small “master” example with two Carrier Units (CUs) and two Transport Units (TUs) is examined. In this scenario:

- Both CUs initially reposition empty from their respective start stations (stations 1 and 2) and form a *platoon*.
- The two CUs then collect two TUs together at station 4 and travel in platoon to station 5.

- At station 5, CU1 unloads TU1 and its journey is completed, while CU2 separates, transitions to *solo* travel, and continues with TU2 to station 6.

This example exercises the C7 (Repositioning – Platoon), C2→C3 (platoon formation), C3→C2 (platoon dissolution mid-route), and C2→C4 (solo arrival) transitions in full.

A.5.1. Input Data

Table A.6: Carrier-Unit itineraries for the master example (split view)

CU	Start	Arc (From→To)	Dep. Time	Travel Time
CU1	Station 1	1 → 3	0	2
		3 → 4	2	1
		4 → 5	3	1
CU2	Station 2	2 → 3	0	2
		3 → 4	2	1
		4 → 5	3	1
		5 → 6	5	1

Table A.7: Transport-Unit requests for the master example

TU	Pickup time	Pickup station station	Delivery time	Delivery station	Route
1	3	station 4	4	station 5	[4→5]
2	3	station 4	6	station 6	[4→5→6]

A.5.2. Event Timeline and State Transitions

Table A.2 shows a *selected extract* from the full simulation trace that illustrates all key transitions. This fully exercises all of our modelled transitions:

Walkthrough of Key Transitions

- **Repositioning platoon (3→4):** Both CUs are departed empty from station 3 at $t = 2$ in state `C7` (reposition–platoon), and arrival at station 4 occurs at $t = 3$.
- **Pickup and transport platoon (4→5):** At $t = 3$ each TU is arrived (E1), loaded (E4), and both CUs are departed together in `C3` (in-platoon) on arc 4→5. Arrival at station 5 occurs at $t = 4$ (E8/E9).
- **Mid-route split at station 5:** The TU assigned to CU1 is completed at this station, while CU2 is required to separate from the platoon. At $t = 5$, CU2 logs E7 (leave platoon), transitioning from `C3` to `C2` (in-solo), after which departure occurs carrying TU2 to station 6.
- **Solo transport to station 6:** CU2 arrives at station 6 at $t = 6$ (event E8), unloads TU2 (E9), and remains in state `C5` (waiting).

This illustrative example confirms that the DES implementation is capable of capturing all CU–TU interaction events, including platooning logic and mid-route splitting behaviour.