Co-funded by
the European Union

Ref. Ares(2024)5644297 - 05/08/2024

Project Acronym:	GREEN-LOG
Project Number:	101069892 (HORIZON-CL5-2021-D6-01 - Innovation action)
Project Full Title:	Cooperative and Interconnected Green delivery solutions towards an era of optimized zero emission last-mile Logistics

DELIVERABLE

D2.1 – Services for orchestrated deployment of new delivery methods V1

Dissemination level	PU – Public
Type of Document	DEM — Demonstrator, pilot, prototype
Contractual date of delivery	30-06-2024
Deliverable Leader	UoW
Status - version, date	Final – v1.0 05-08-2024
WP / Task responsible	WP2 / T2.1, T2.2, T2.3, T2.4
Keywords	Last-mile logistics data space, Logistics demand forecasting, Micro-consolidation centres' location optimisation, Simulation framework for last-mile logistics.

Executive Summary

GREEN-LOG aims to accelerate systemic changes in last mile delivery ecosystems for economically, environmentally and socially sustainable city logistics. The GREEN-LOG project is a multi-year project between 30 partners, deployed across Europe and the UK. The project employs beyond state-of-the-art technology, thorough investigation of the use cases of the logistics partners and real-world testing of the designed systems in Living Labs.

This deliverable presents the outputs of four (4) tasks (T2.1, T2.2, T2.3 and T2.4) of WP2 “New delivery methods and orchestrated deployment” up to M18. The objective is to integrate existing commercial logistic hubs, located in an urban or peri-urban context with all the advances of algorithmic simulation and prediction using real-time and future predictive artificial intelligence models.

More specifically, the deliverable presents the work undertaken towards the development of a suite of demand prediction models and optimisation services for scheduling last mile deliveries with the use of Micro Consolidation Centres (MCCs). The above-described services, together with tools and solutions being developed in WP3, utilise a city logistics data space presented in the deliverable. The GREEN-LOG data space facilitates data management to meet the project’s kai Living Labs’ needs. Furthermore, a simulation platform for the design and evaluation of what-if scenarios and long-term planning from either the LSP or an authority’s perspective is described in the report.

The developed tools aim to reduce congestion, exhaust emissions, demand response time and improve the overall sustainability of last-mile delivery solutions. Situational information including traffic prediction, route accessibility, vehicle availability and historical performance will be amalgamated to provide guidance to logistics partners.

The design, modelling, implementation and testing of the first (alpha) release of each software component are shown in this report. The integrated platform providing access to tools and algorithms developed in T2.1-T2.4 will be delivered in M21 through the deliverable D2.2 “Augmented intelligence modelling platform V1”.

Deliverable Leader (organisation)	UNIVERSITY OF WOLVERHAMPTON (UoW)
Contributors (organisations)	FRONTIER INNOVATIONS EE (FI), FRONTIER KENTRO KAINOTOMIAS MAKE (FRIC), ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS - RESEARCH CENTER (AUEB), FUNDACIO EURECAT (EUT), PANEPISTIMIO AIGAIOU (UAEGEAN), AIMSUN SLU (AIMSUN), MOSAIC FACTOR SL (MOSAIC), MOBY X SOFTWARE LIMITED (MOBY), UNIVERSITY OF WOLVERHAMPTON (UoW), E-NOVIA SPA (E-NOVIA), ACS SMSA, Ferrocarrils de la Generalitat de Catalunya (FGC), SOLUCIONS ULTIMA MILLA SL (VANaPED), FedEx Express Greece (FedEx), PEDAL & POST (PEDAL)
Reviewers (organisations)	ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS - RESEARCH CENTER (AUEB), ACS TACHIDROMIKES IPIRESIES MONOPROSOPI ANONYMI EMPORIKI ETAIREIA (ACS SMSA)
Approved by (organisations)	Netcompany-Intrasoft S.A. (INTRA)

Document History			
Version	**Date**	**Contributor(s)**	**Description**
0.1	2024-08-05	UoW	ToC
0.2	2024-05-30	UoW	Preliminary reports
0.3	2023-06-10	UoW	UoW content
0.4	2024-06-18	UoW, EUT, AUEB, AIMSUN	Preliminary report content
0.5	2024-06-18	UoW, EUT, AUEB, AIMSUN	Partner content finalised
0.6	2024-06-28	UoW, AUEB, ASC SMSA	Incorporated internal reviewers' amendments
0.7	2024-08-02	UoW	Comments from PC addressed
1.0	2024-08-05	INTRA	Final QA and submission

Table of Contents

Table of Figures

List of Tables

Definitions, Acronyms and Abbreviations

Acronym	Title
CNN	Convolution Neural Network
DL	Deep Learning
IDS	International Data Space
IDSA	International Data Space Association
IDS-RAM	International Data Space – Reference Architecture Model
JRC	Joint Research Centre
LL	Living Lab
LSP	Logistics Solution Partner
LSTM	Long Short-Term Memory
MDH	Mobile Delivery Hub
MILP	Mixed-Integer Linear Program
PEDAL	Pedal & Post
VRPTWMD	Vehicle Routing Problem with Time Windows and Mobile Depots
WP	Work Package

1 Introduction

1.1 Document scope

This report presents the culmination of work on the GREEN-LOG project Tasks 2.1, 2.2, 2.3, and 2.4. WP2 implements various tools that can support city logistics stakeholders in planning for last-mile services. The integrated platform (Task 2.5) translates the requirements' specifications of WP1 into a system intended for Living Lab demonstrations at various sites across Europe.

More specifically, the design and development of the GREEN-LOG data space (alpha version) is explained in the deliverable. The proposed solution incorporates a Data Management System (DMS), CKAN, a series of smart data models, and a context broker from the FIWARE ecosystem of tools, Orion-LD. The context broker stores context information updated from applications, ensuring that queries are resolved based on that information. This work is related to T2.1.

Furthermore, the deliverable presents the work undertaken within T2.2 and T2.3 towards the development of a suite of demand prediction models and optimisation services for scheduling last mile deliveries with the use of Micro Consolidation Centres (MCCs). The former utilises historical and day-to-day demand data to provide forecasts for medium-term (one week in the future) time horizons. The latter implements day-to-day optimisation algorithms for generating tours for vehicles in logistics fleets with the incorporation of dynamic positioning of MCCs that act as loading points for individual vehicles.

Finally, the deliverable explains the work towards the development of a simulation platform for assessing last-mile logistics scenarios (T2.4). The platform implements (i) a one-off two-stage approach suitable for what-if scenarios and long-term planning from either the LSP or an authority's perspective, and (ii) an iterative two-stage approach computing equilibrated fleet service conditions for day-day service fleet management (i.e., fleet scheduling).

1.2 Document structure

The deliverable is structured according to the outputs related to each of the task described above. The description of the work undertaken in each task is divided into sections including, where applicable, state-of-the-art review, methodological approach followed, specifications for the relevant components, implementation approach and results following early applications at the different pilot sites.

Task 2.1 has developed a Data Space to IDS-RAM3 standard to support the transfer and management of data shared between the commercial living lab users and the software implementations of research partners. This work is presented in Chapter 2.

Chapter 3 contains the results of Task 2.2 which focused on the development of artificial intelligence algorithms to model logistics demand and its prediction for different spatio-temporal configurations.

Task 2.3 developed optimisation models for scheduling day-to-day operations to consolidate delivery deployment through the use of micro-consolidation centres (MCC). These integrated models have been applied to all the living lab areas. Chapter 4 explains the activities undertaken in the context of this task.

Chapter 5 presents the outcomes of Task 2.4 which involved the development of a wide variety of micro- and meso-scopic models for the GREEN-LOG demonstration sites. The deliverable concludes with Chapter 6, which summarises the key outcomes of the preceding chapters, as well as future developments foreseen for the technologies and solutions presented.

Where software is referenced in the WP's, it will be made available through the project site or from the work package producers. If possible, at the time of submission, a link to the software is provided in the relevant section.

2 City Logistics Data Space

2.1 Background/State of the Art

A data space is a structured environment where data is collected, shared, and utilized to drive innovation and improve various aspects of life and industry. The concept of a data space is centred around the idea that data, when effectively harnessed, can lead to significant advancements across different sectors.

To maximise the benefits of data for the European economy and society, the European strategy for data was introduced in February 2020. This strategy aims to create Common European Data Spaces[1] in various strategic fields.

2.1.1 International Data Spaces Association

The International Data Spaces Association (IDSA), launched by the Fraunhofer Institute in 2015, comprises over 130 organisations from more than 20 countries. It has developed a reference architecture and formal standard for creating and operating virtual data spaces. The goal is to enable secure and trusted data spaces where companies of all sizes and industries can manage their data assets sovereignly.

The IDS architecture, known as IDS-RAM, is based on a widely accepted data governance model that ensures secure, reliable data exchange and easy data linking within enterprise ecosystems. It is a high-level, peer-to-peer architecture allowing data sharing among third parties while preserving data providers' digital sovereignty. This architecture supports the development of intelligent services and innovative business processes.

[1] Common European Data Spaces | Shaping Europe's digital future (europa.eu)

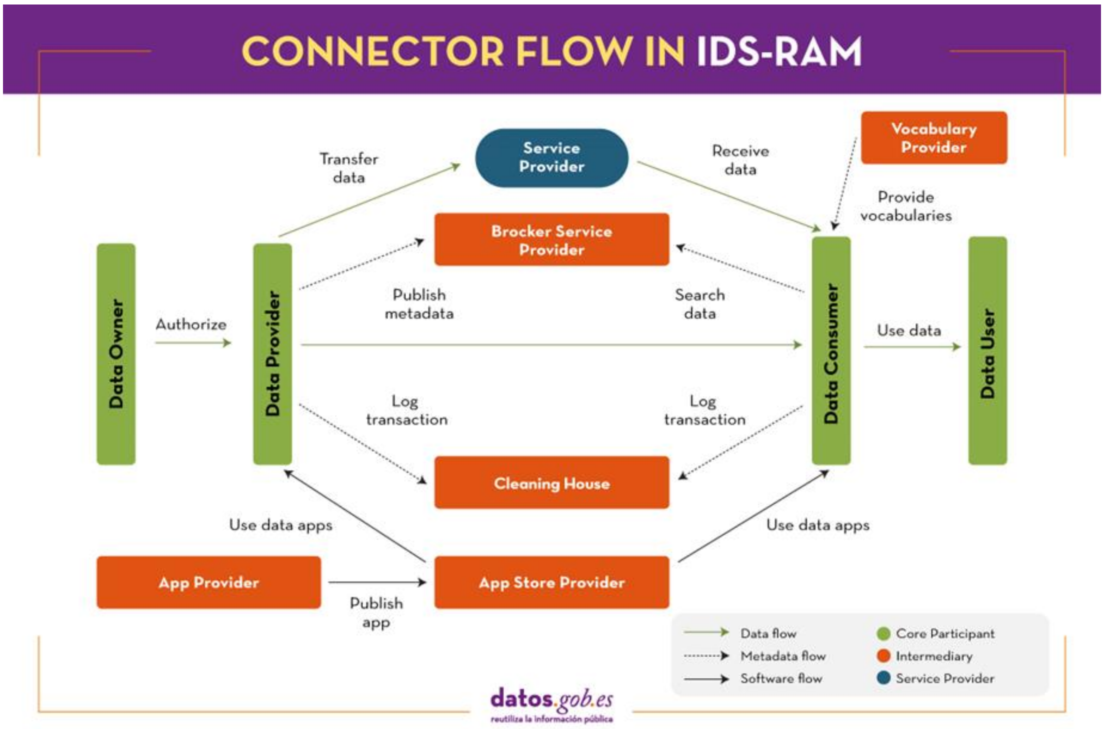


Figure 1 Connector flow in IDS-RAM

2.1.2 GAIA-X

The GAIA-X[2] initiative, originating from the Franco-German public-private ecosystem in 2019, became a pan-European non-profit association based in Brussels in February 2021. It aligns with the European Data Strategy, focusing on the data economy and cloud services, particularly leveraging the cloud as-a-service model for flexibility and ease.

GAIA-X aims to develop an open software layer (Figure 2) for control and governance, implementing a common set of policies and rules to ensure transparency, sovereignty, and interoperability between data and services across various cloud/edge technology stacks. This open layer and its associated policies can be deployed by any cloud service provider.

The initiative seeks to make access to datasets, services, and resources straightforward and transparent for ecosystem participants, despite these elements residing on different systems with unique characteristics. GAIA-X's primary task is to define a process architecture that organises this federation of computer systems, ensuring proper identification of data and service providers and users, and establishing the necessary technical and governance rules for a federated model.

It aspires to create coordination between the data layer (where data sets are shared, transformed, and exploited; in the image below, it is the blue layer) and the of infrastructure and resources (in red). This coordination between the layers is the key, as GAIA-X neither seeks to create projects based on data (there are other European initiatives with this objective), nor operates cloud services.

[2] https://gaia-x.eu

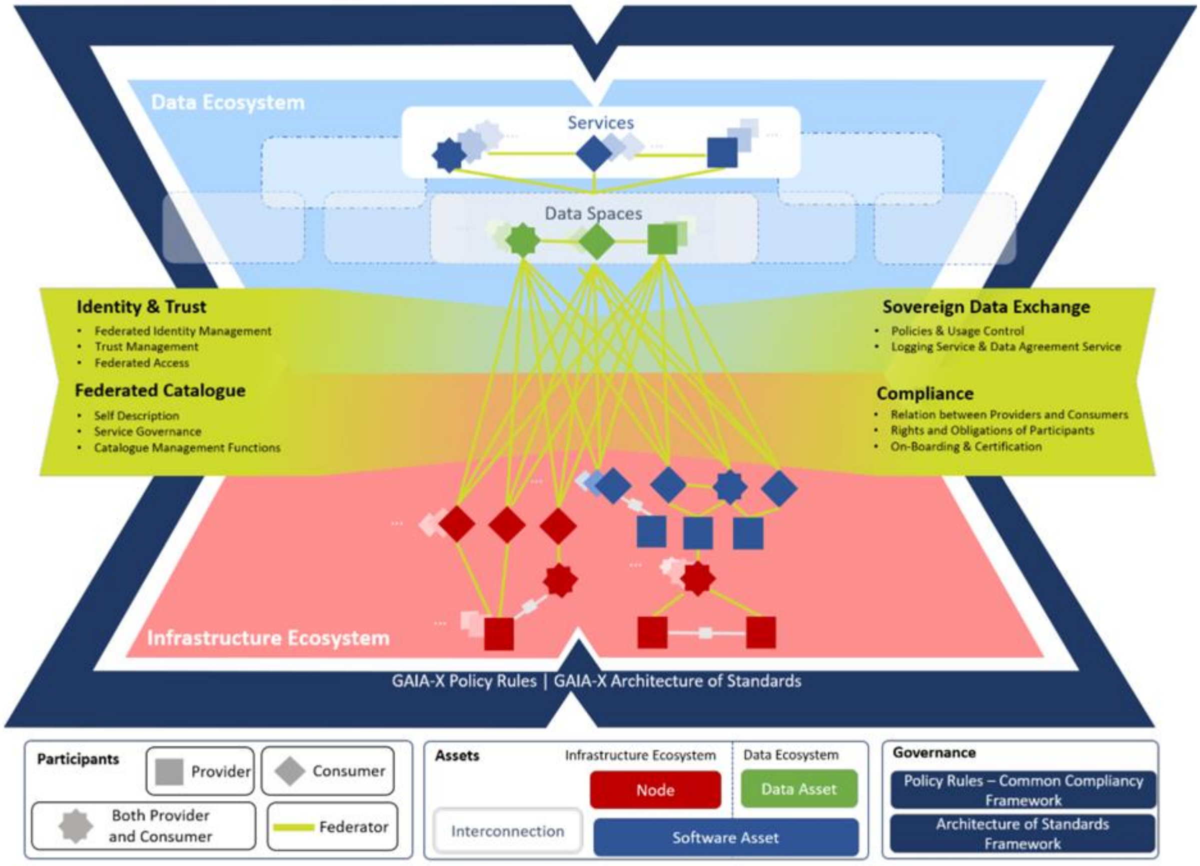


Figure 2 The GAIA-X software layer

2.1.3 iSHARE

The iSHARE[3] Foundation is an initiative aimed at facilitating secure and efficient data sharing within logistics and supply chain networks. It focuses on creating a standardised framework that enables organisations to share data in a controlled and secure manner, ensuring trust and interoperability among participants.

The foundation provides a standardised set of rules and protocols for data sharing, which all participants adhere to, thereby enhancing security and trust. Robust security measures and clear authorization protocols ensure that data is shared only with authorized parties, maintaining the confidentiality and integrity of the information. Additionally, iSHARE's framework is designed to be interoperable across different systems and platforms, allowing seamless data exchange between various stakeholders in the logistics and supply chain sectors. By reducing the need for individual bilateral agreements, iSHARE streamlines the data sharing process, making it more efficient.

[3] Https://ishare.eu

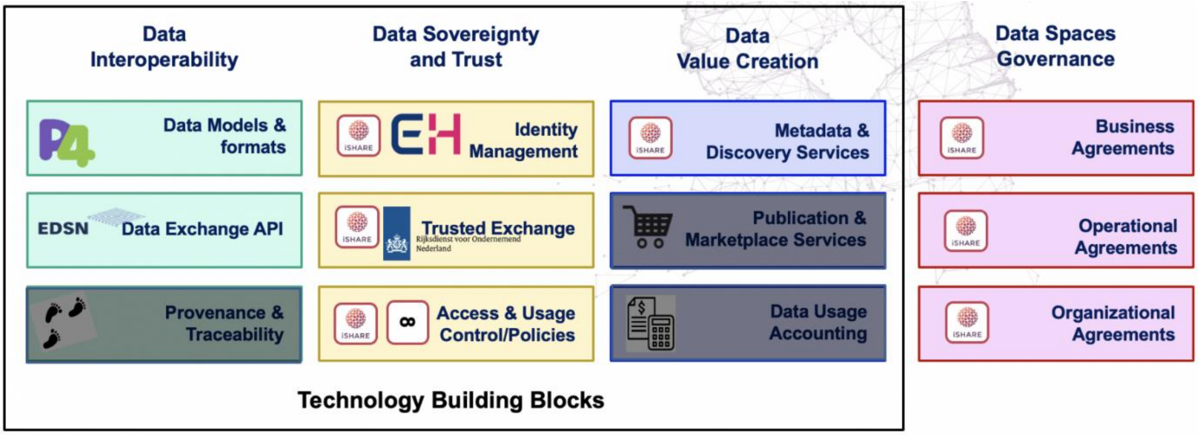


Figure 3 iSHARE technology building blocks

2.1.4 FIWARE

FIWARE[4] provides a curated framework of essential open-source software components and standards that support the development of interoperable smart solutions based on data, along with their exchange and reuse in advanced technologies such as IoT. The FIWARE reference architecture enables plug-and-play modules, making it adaptable to various domains and industries. It primarily relies on the ETSI-CIM (NGSI-LD) standard, which offers a common representation of information compatible with JSON-LD and Linked-Data principles.

This architecture allows for the creation of solutions that integrate data from different systems, repositories, and architectures (referred to as southbound APIs based on Agents). It also facilitates linking data with third-party components and applications to develop comprehensive systems.

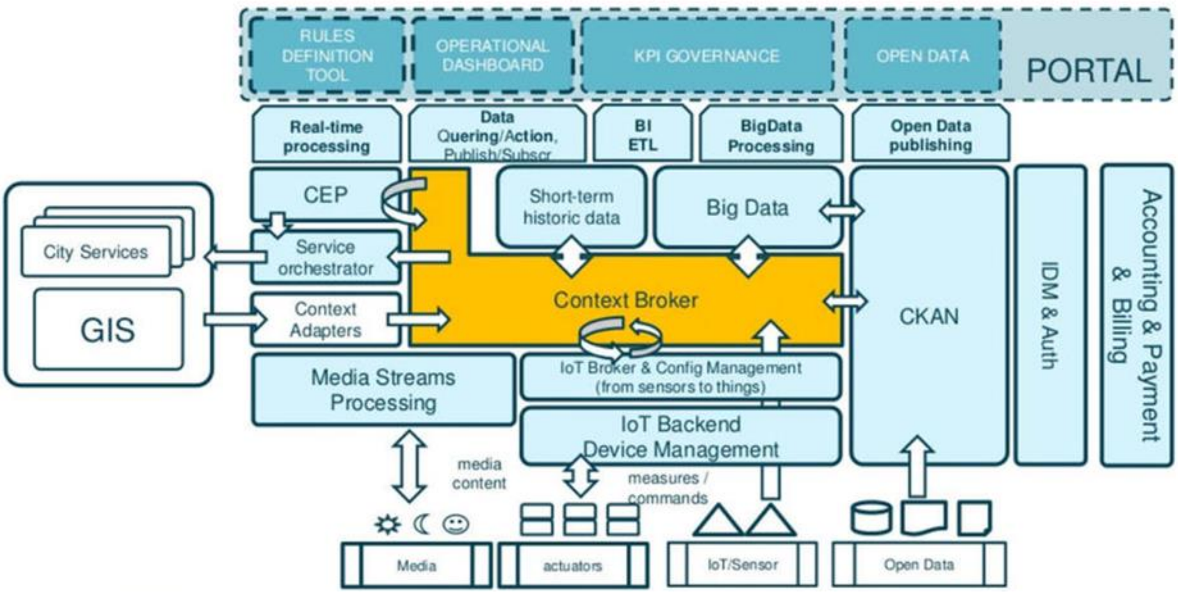


Figure 4 The FIWARE data model

[4] FIWARE

In addition, FIWARE has fostered a community focused on developing reusable and smart data models to represent entities and processes across different domains. This active community tests and validates the data models in real scenarios, ensuring their practicality and effectiveness.

Therefore, the main features of the FIWARE architecture relies on: (i) Modular application that could be transferable and replicable at different scales; (ii) Management of context information in a highly decentralised and large-scale manner; (iii) Open-source implementation and compatibility with informational standards such as NGSI-LD.

2.2 Methodological Approach

The chosen approach aims to ensure interoperability, scalability, and alignment with industry standards. A key factor in this decision is the utilization of smart data models, to enhance data consistency and integration. Developing an interoperable Data Space using FIWARE components, such as the Orion-LD context broker, requires a structured methodology to ensure successful integration, scalability, and functionality.

2.2.1 Mounting the Data Management System (CKAN)

One of the initial steps was the deployment of a Data Management System (DMS), in concrete, CKAN. This system facilitates the acquisition and management of initial and test datasets, enabling exploration of data structures, determination of metadata requirements, and selection of appropriate data models. This setup provides a controlled environment to experiment with data, gaining insights into its characteristics and requirements.

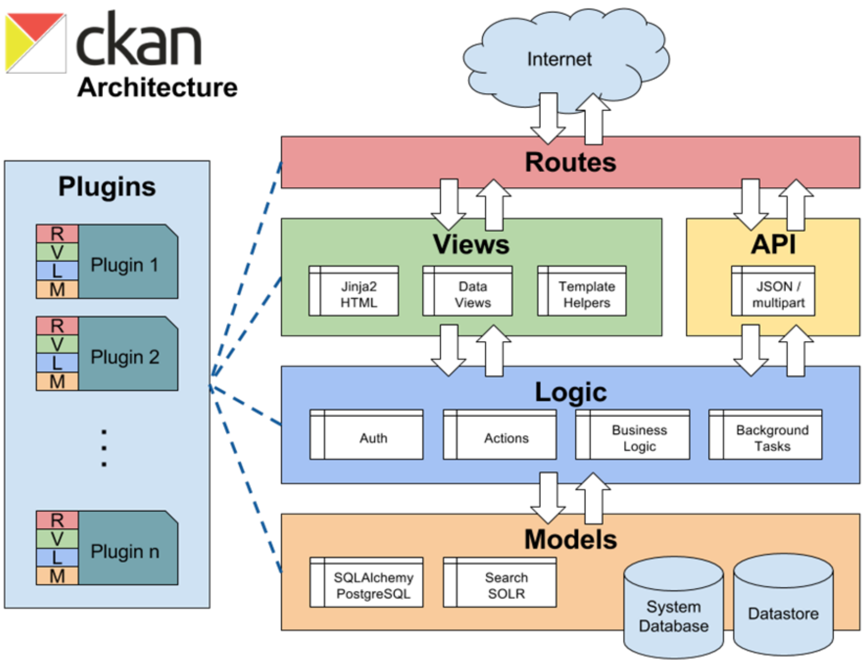


Figure 5 CKAN architecture

2.2.2 Developing the Smart Data Model

Adherence to the NGSI standard v2 will serve as the foundational framework ensuring consistency and interoperability across the data ecosystem. The process involves translating raw data using smart data models, thereby converting it into the NGSI-LD format.

NGSI-LD (Linked Data) enriches traditional NGSI v2 by embedding semantic meaning into the data, enabling better understanding and integration. This transformation process entails integrating various data models to construct a comprehensive context. These models, collectively, define the attributes of entities, properties, and relationships across diverse datasets.

By leveraging NGSI-LD and integrating multiple data models, the approach aims to establish a unified and precise context. This contextual clarity facilitates accurate specification and interaction with entities, properties, and relationships across different datasets. Ultimately, this methodology ensures that the data is not only standardised but also enriched with semantic meaning, thereby enhancing its utility and interoperability.

2.2.3 Contextualizing the data (Orion-LD)

The focus shifts to implementing the context broker, specifically Orion-LD, to apply the selected smart data model to the raw data. This step is pivotal in transforming the data into linked data that adheres to the NGSI standard.

Orion-LD is a robust platform designed for managing context information in accordance with NGSI-LD, the linked data version of the NGSI standard. By leveraging Orion-LD, the methodology ensures the effective application of the previously selected smart data model to raw data. This application process involves mapping the raw data's structure and attributes to the entities, properties, and relationships defined in the smart data model.

The transformation to linked data within the NGSI standard encompasses several key activities:

- **Mapping and Integration**: Mapping the raw data fields to the entities, properties, and relationships defined in the smart data model.
- **Normalisation**: Ensuring consistency and normalisation of data formats and structures to comply with NGSI-LD requirements.
- **Semantic Enrichment**: Enhancing the data with semantic annotations and metadata to provide richer context and meaning.
- **Validation and Compliance**: Validating the transformed data to ensure it meets NGSI-LD standards and specifications.

This transformation not only enhances interoperability but also enables advanced functionalities such as real-time querying, data fusion, and context-aware decision-making.

2.3 Component / Service / Model specification

2.3.1 Context Broker (Orion-LD)

Orion-LD serves as a Context Broker and a CEF building block for context data management which supports both the NGSI-LD and the NGSI-v2 APIs. Originally a fork of the Orion Context Broker, Orion-LD extends its capabilities to include NGSI-LD and linked data concepts. Orion-LD follows the ETSI specification for NGSI-LD and has been tested to ensure stability and speed, closely in compliance with version 1.3.1 of the NGSI-LD API specification.

Orion Context Broker stores context information updated by applications, so queries are resolved based on that information. Users can perform several operations through the Orion interface, including querying context information, updating context information, receiving notifications about changes in context information, registering context provider applications.

2.4 Implementation Approach / Description / Results

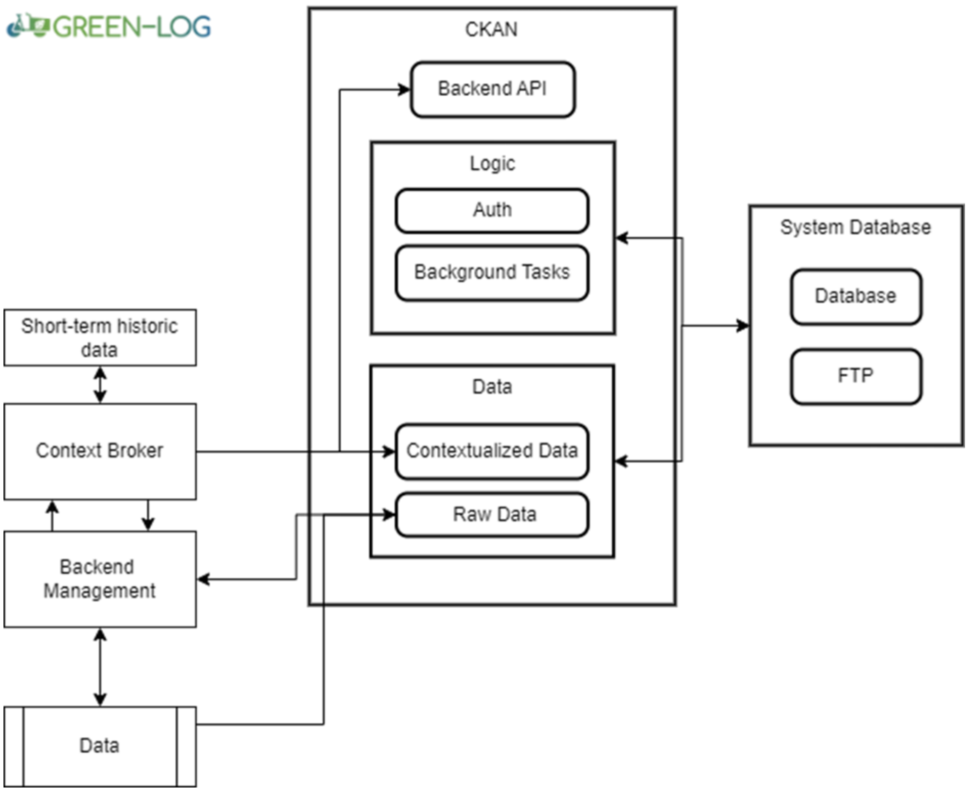


Figure 6 GREEN-LOG implementation of Data Space

The dataspace consists of various services that have been dockerized for ease of deployment and maintenance. Each of these services is a component in a Docker Compose setup. The project comprises three main parts, the backend API to support data exchange with the different components using a REST approach to interface the data repository, the Logic component that includes the authentication service and other additional background tasks for data harmonisation, and the Data component that stores both raw data and contextualized data.

2.4.1 Contextualization

To contextualize the data, several components from the FIWARE ecosystem have been implemented, including Quantum Leap, Orion context broker (Orion-LD), and Wilma. These components require additional resources to function properly, such as Keyrock for authentication and MongoDB for short-term historic data and subscription management.

- QuantumLeap is a REST service for storing, querying and retrieving NGSI v2 and NGSI-LD (experimental support) spatial-temporal data. QuantumLeap converts NGSI semi-structured data into tabular format and stores it in a time-series database, associating each database record with a time index, and a location if it's needed.
- Orion Context Broker allows to manage the entire lifecycle of context information including updates, queries, registrations and subscriptions. It is an NGSIv2 server implementation to manage context information and its availability. Using the Orion Context Broker, the user is able to create context elements and manage them through updates and queries.
- Wilma is a PEP-Proxy, that in this case, is implemented with Keyrock to enforce access control to backend applications, so only permitter users are able to access to the REST services.

So the 5 components interact by them in the contextualization flow. At the heart of this system is the Orion Context Broker, which serves as the central hub for context information. It collects, updates, queries, and manages context data from various sources, ensuring real-time data availability and consistency.

When a context change occurs within the Orion Context Broker, it notifies Quantum Leap, which subscribes to these changes. Quantum Leap is responsible for storing this updated context data in a time-series database. This storage capability enables us to perform spatial-temporal queries, allowing for the analysis of context data over time and across different locations.

Security and access control within our system are managed by Wilma, which acts as a gatekeeper for both the Orion Context Broker and Quantum Leap. Wilma relies on Keyrock for authentication, ensuring that only authorized users can access these services. Keyrock manages user identities and enforces access control policies, integrating seamlessly with Wilma to maintain secure interactions between users and the protected services.

Additionally, MongoDB plays a vital role in the functioning of the Orion Context Broker. It stores short-term historic data and manages subscriptions, facilitating efficient data retrieval and handling context updates effectively. This integration ensures that the context data is not only current but also historically accurate and accessible for various applications.

An additional service, known as the data connector, acquires raw data and transforms it into linked data using the selected smart data model. This transformed data is then sent to Orion-LD for further processing and management.

2.4.2 Data acquisition and storage

To facilitate data acquisition, various connectors will be developed to extract project-relevant data.

The Data Management System has been deployed, customized, and configured with proper authorization. The data connectors acquire raw data using the API from CKAN and translate it into linked data (LD). Once the data is contextualized, it is sent to the contextualization services. After the data has been fully processed and complies with the NGSI standard, it is stored again in the Data Management System. This contextualized data is then accessible via the CKAN UI and API.

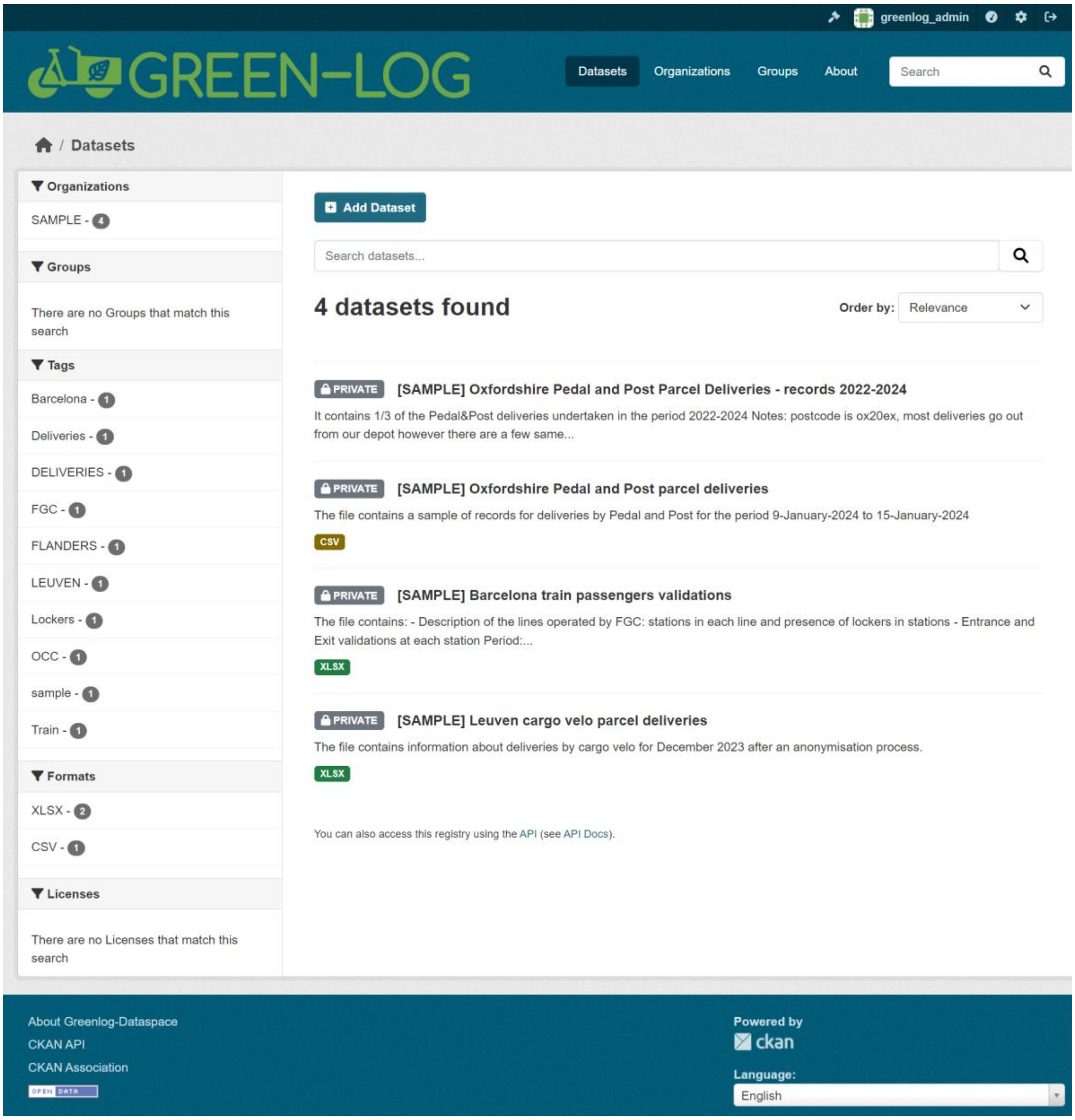


Figure 7 View of CKAN implementation for GREEN-LOG Data Space

3 Spatio-temporal demand prediction services

3.1 Background

WP2 designs, tests and configures last mile interventions to achieve sustainable businesses, road transport efficiency and environmental goals through the orchestrated deployment of last mile delivery solutions, with data driven simulation and optimisation models and decision support systems.

This section presents the work undertaken as part of Task 2.2 regarding the development of a demand prediction service that can be utilised by stakeholders to plan medium-term operations for last-mile logistics. The current version of the demand prediction service has been implemented using data from the Oxfordshire Living Lab (LL) and will subsequently be expanded to operate in the other project's LLs in the future. The Oxfordshire LL partners providing data and processing for demand prediction service are depicted in Figure 8.

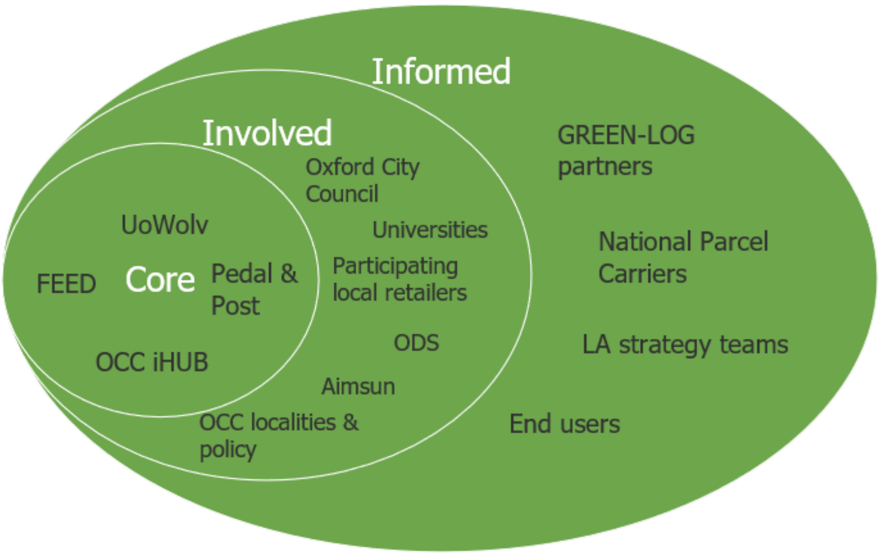


Figure 8 Partners providing data and processing for demand prediction service[5]

The developed service utilises historical and daily demand data to generate medium-term forecasts of demand.

For the development of this demand prediction service, different machine learning models have been designed. Among these, convolutional neural network (CNN) and long short-term memory neural networks (LSTM) are recognised as highly efficient deep learning methods for demand forecasting, ensuring high accuracy in the real-world environment. Therefore, we combine CNN with the LSTM to form a deep learning-based prediction model, CNN-LSTM, to conduct the logistics demand prediction efficiently and accurately.

[5] ODS: Odin Data Services, OCC iHUB: Oxfordshire County Council iHUB.

3.2 State of the Art

In recent years, alongside the rapid development of the world economy, the logistics industry has also been developing significantly. This expansion not only supports economic growth but also reflects the increasing demand for logistics services driven by economic development. However, the existing traffic conditions and infrastructure are inadequate to meet the growing logistics needs, thereby limiting its ability to continuously support the national economy.

In the modern world, due to many forces such as the increased customer expectations, pandemics, technological dependence and war, the business environment is dynamic and unstable. As a result, businesses must swiftly adapt to market changes and unexpected events to maintain competitiveness. These have emerged as issues that require urgent consideration. The theory of supply and demand shows that in any given period, when the logistics capacity fails to meet demand, it constrains overall demand. Conversely, an oversupply of logistics capacity results in a wastage. Therefore, logistics demand forms the foundation of logistics capacity supply.

Forecasting and analysing logistics demand provide a theoretical framework for social logistics activities, ensuring a relative balance between the supply and demand of logistics services. This equilibrium supports high efficiency and benefits in social logistics operations. The social and economic significance of logistics demand forecasting analysis also lies in this.

There are many ways to predict logistics demand, but it is difficult to accurately predict the flow of goods in the whole society. From the current theoretical basis and research methods, we can only grasp the trend direction and quantity of logistics demand from a macro perspective.

Applications of information and communication technologies assist organizations in making appropriate decisions and developing effective strategies to manage and control unanticipated effects. They assist organizations in dealing with supply chain issues caused by the increased movement of goods and data in the supply chain.

Many predictions and management tools based on artificial intelligence (AI) have been released, allowing enterprises to record large amounts of data and robust. AI has been widely used in various fields, such as engineering, sciences, banking, finance, economics, tourism, and healthcare. AI is the simulation of human intelligence using software-coded heuristics. AI is the primary innovation for creating intelligent machines that mimic human behaviour and perform various. AI is better suited for complex input-output relationships. Thanks to AI, manufacturers can effectively obtain upstream and downstream product information to make precise product predictions and decisions.

AI modelling is being created for intelligent tourism platforms to accurately predict tourism choice behaviour patterns. Thus, AI-based techniques have significantly contributed to the evolution of various sectors, industries, and supply chains. One of the most significant contributions of AI applications is demand forecasting, an essential supply chain management operation. It forecasts future demand to plan supply chain activities and operations to decrease shipping times, inventories, and operational expenses.

AI can analyse data and forecast demand, optimize logistics and transportation routes, and identify bottlenecks in the supply chain. Furthermore, AI helps to improve supply chain transparency, which is one of the critical requirements for a successful supply chain. Besides that, machine learning (ML) and deep learning (DL) techniques are also widely used in many fields, and they are deployed mostly for time series problems, for instance: healthcare, and trading and stock price predictions.

There are various AI methods used in demand prediction techniques in the literature. Deep Learning shows better performance and results in terms of forecasting compared to other methods. While forecasting based solely on historical data of manufacturing demand is achievable, the accuracy of the prediction is significantly enhanced when multiple factors are considered. Researchers set up strategies of the exponential forecasting framework for sales forecasting to optimize manufacturing planning and inventory. Therefore, our aim is to forecast future demands based on historical data coming from past manufacturer orders and retailers' demands based on customers' requirements, just enough time in advance to compensate for manufacturer Lead-times.

Several ML and DL methods are used to develop the accuracy of demand forecasting. Abbasimehr [REF-08] used multi-layers LSTM method to compare with other time series forecasting methods, such as K-nearest neighbours (KNN), exponential smoothing (ETS), AutoRegressive Integrated Moving Average (ARIMA), Support Vector Machines (SVM), Recurrent Neural Network (RNN), Artificial Neural Network (ANN) and Long-Short Term Memory (LSTM).

The results of this study show that LSTM method is more efficient compared to the tested methods with regards to performance measures [REF-08]. Wang et al. [REF-11] used LSTM, and Grey Model (GM) models to predict the future values based on historical data and the total industrial value. The Statistical Yearbook of GD was selected to represent the demand of the manufacturing industry. Other indicators are from this statistical yearbook from 2005 to 2020. The experiment shows that LSTM has excellent results in the previous comparative experiments. The GM model is the classic model in the field of Auto-regression. Nevertheless, it does not give accurate results, which is significantly lower than the forecasting results given by LSTM considering multiple factors [REF-09].

A study conducted by Raizada et al. [REF-10] is based on a comparative analysis of several Supervised Machine Learning algorithms, such as, Support Vector Machine (SVM), Random Forest Regression, K-NN algorithm and Extra Tree Regression to build a forecasting model for future sales of 45 retail outlets of Walmart store in India. The study shows that Extra Tree Regression Technique is the most efficient model to predict the sales for the selected dataset; however, the predictions obtained from the algorithm may vary based on the variance in training data [REF-10][REF-20]. Wang et al. [REF-05] compared the performance of classical forecasting models and the latest developing forecasting technologies for perishable products and non-perishable items of a large grocery retailer. The authors made a comparison in terms of performance and accuracy between many algorithms, such as: ARIMA, SVM, RNN and LSTM [REF-11][REF-20]. The study shows that SVM, RNN and LSTM have a high predictive performance to for perishable products, whereas ARIMA is outstanding in the runtime and LSTM is the most efficient method to deal with non-perishable items due to its advanced prediction performance [REF-11][REF-18].

There are several ways to apply demand forecasting. In general, the forecasts fluctuations depend on the model used. Using multiple forecasting models could also highlight differences in forecasts. These differences may indicate the need for more research or better data input.

3.3 Methodological Approach

3.3.1 Methods

The development of deep neural network has introduced the CNN, which has been proposed and is currently one of the most prominent algorithms. CNN is a type of deep neural network, known for its complex structure, typically comprising layers such as input, convolutional, pooling, fully connected, and output layers. It has successfully been applied in various fields such as detection and segmentation. It provided outstanding performance than the previous traditional machine learning algorithms in the above fields.

A CNN includes several convolutional layers, pooling layers, and fully connected layers, and it is a regularised type of feed-forward neural network that learns feature engineering by itself via filters (or kernel) optimisation. Vanishing gradients and exploding gradients, seen during back-propagation in earlier neural networks, are prevented by using regularised weights over fewer connections. The details of each component are introduced as follows. The function of the convolution layer is to collect the features of input data, which contains several convolution kernels.

Each cell of the convolution kernel corresponds to a bias vector and a weight coefficient, like a neuron of the feedforward neural network. Each neuron in the convolutional layer is related to multiple neurons in the area close to the previous layer. The size of the area depends on the size of the convolution kernel that is called “receptive field” in the literature. Its meaning can be compared to the receptive field of visual cortical cells.

During the convolution kernel operation, the input features will be scanned regularly, and the input features will be summed and multiplied by the matrix elements in the receptive field, and the deviations are superimposed. After feature extraction in the convolutional layer, the output feature map is going to be passed to the pooling layer for information filtering and feature selection.

The pooling layer contains preset pooling functions whose function is to replace the result of a single point with the statistics of the feature map of its neighbouring area in the feature map. The selection of the pooling area in the pooling layer is not different from the steps of the convolution kernel scanning feature map, controlled by the pooling size, step size, and filling. In the traditional feedforward neural network, the fully connected layer is equal to the hidden layer. The fully connected layer is in the last part of the hidden layer and only transmits signals to other fully connected layers. The featured graph loses the spatial topology in the fully connected layer and is expanded into a vector and passes the excitation function.

The convolutional layer and pooling layer can perform feature extraction on the input data. The role of the fully connected layer is to nonlinearly combine the extracted features to obtain an output. Fully connected layers play the role of “classifier” in the entire convolutional neural network. The convolutional layer, pooling layer, and activation function layer map the original data to the hidden layer feature space. The fully connected layer plays the role of mapping the learned “distributed feature representation” to the sample label space.

The fully connected layer isn’t expected to have features extracting ability but trying to use the existing high-order features to complete the learning goal. LSTM is specially devised to solve the long-term dependence problem and it has been successfully used in various fields such as machine translation, speech recognition, image description generation, video tagging and financial time series. It mainly includes forgetting gate, input gate, and output gate. The network can effectively mine the temporal dependence of information in the time series by adding the forget gate, input gate, and output gate to the hidden layer. At each moment, the unit receives the current data through three gates input and the last hidden state as well as the memory unit state.

Figure 9 shows a sample of the raw data from.

	shipment_service_type	shipment_required_service_date	shipment_postcode	shipment_latitude	shipment_longitude
0	D	04/04/2022	OX4 4PE	51.723081	-1.224018
1	D	04/04/2022	OX4 4PU	51.720039	-1.225235
2	D	04/04/2022	OX2 8HW	51.789436	-1.277997
3	D	04/04/2022	OX4 4YS	51.712315	-1.218971
4	D	04/04/2022	OX4 2QG	51.739862	-1.204468
...	...	...	...	...	...
71916	D	24/04/2024	OX4 4HF	51.726441	-1.230735
71917	P	24/04/2024	OX4 4HF	51.726441	-1.230735
71918	D	24/04/2024	OX3 9DU	51.764391	-1.218877
71919	P	25/04/2024	OX4 2PG	51.729679	-1.184887
71920	D	25/04/2024	OX4 1RW	51.744949	-1.232480

71921 rows × 5 columns

Figure 9 Raw Delivery data from PEDAL in unprocessed form

Raw data was explored and pre-processed aggregating demand per day from column “shipment_required_service_date”, “filtering shipment_service_type” labelled as “D” (for delivery) and ignoring spatial information from the rest of column (cf. p&p_explore.ipybn).

Figure 10 shows the resulting dataset aggregating deliveries per day under column “n_parcels”. Note that the resulting dataset initially included demand booked for future dates (e.g. 13-12-2024) at the time of the experiment, which have been filtered out (cf. dataset.csv). Moreover, temporal features had been added to the data, namely weekday, month, year and week to fully explore the temporal nature of the forecasting task.

	date	n_parcels
0	2022-04-04	170
1	2022-04-05	101
2	2022-04-06	119
3	2022-04-07	208
4	2022-04-08	167
...	...	...
640	2024-04-29	2
641	2024-04-30	2
642	2024-05-10	52
643	2024-05-24	52
644	2024-12-13	1

645 rows × 2 columns

Figure 10 Aggregated demand as number of parcels per day

A peak in demand was spotted in July 2022 which was discussed with PEDAL who confirmed they had manually added other source of delivery data (see Figure 11). It is important to note at this stage that PEDAL has different sources of delivery data, and this dataset is just a partial view on their actual delivery activities. That may be the reason why seasonality[6] is not reflected in the plot shown in Figure 11. For instance, looking at data from October 2022 and October 2023, the range of demand varied greatly from 0-100 to 0-600.

Seasonality helps to understand and predict recurring demand patterns, allowing stakeholders to make better-informed decisions. Whether it's the holiday shopping season or weather-related fluctuations, recognizing seasonality is crucial for effective forecasting and planning. The lack of monthly seasonality in data may be due to partial rendering of data or any other casualties that PEDAL may be able to report and/or complement in further stages of the project.

[6] Seasonality reflects regular and anticipated changes in data that recur every calendar year.

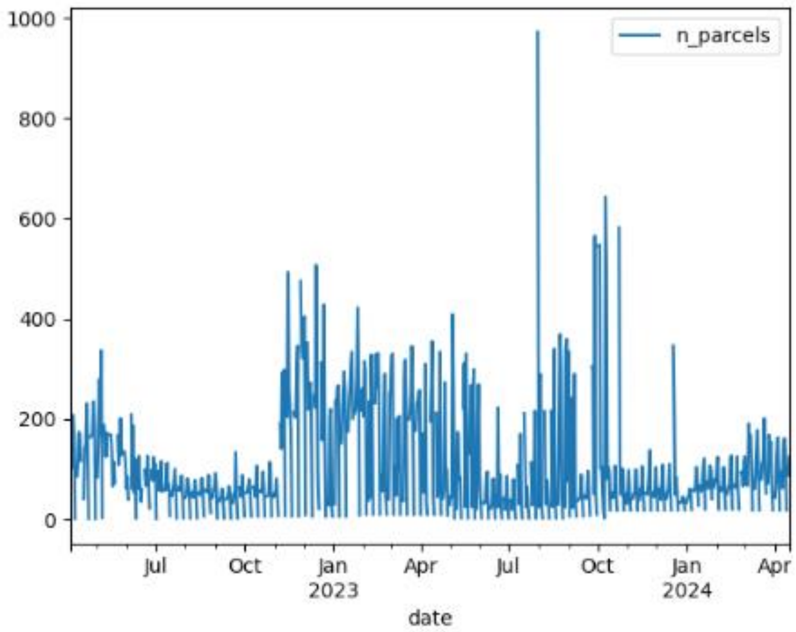


Figure 11 Plot number of parcels between April 2022 and April 2024.

3.3.2 Model Component

Considering the complex spatiotemporal features in the dataset, a deep learning-based demand prediction model called CNN-LSTM is used to efficiently process the dataset. This prediction model employs two different types of artificial neural network layers to fully capture spatial and temporal features in the dataset.

Firstly, it uses the convolutional neural network to capture spatial correlations in the dataset and slides over the input data via convolution kernel to extract features from local regions which helps detect local patterns and features in the dataset and a pooling layer is usually added after the layer to reduce the spatial dimension of the data to reduce the computational complexity.

After that, it uses the long short-term memory neural network to learn long-term dependencies in the sequence data and capture the time evolution pattern in the dataset and the long short-term memory neural network is a variant of recurrent neural network and is specifically used to process modelling of sequence data since it uses gating mechanisms including input gate, forget gate and output gate to control forgetting and remembering of past information to effectively deal with long sequence dependencies in the dataset.

The prediction model is used to process spatial and temporal information in sequence data. The general procedure of the model to predict spatial-temporal series data is as follows. The first stage is to prepare the input data. In general, sequence data can be represented as tensors in three dimensions, where the first dimension represents the number of samples, the second dimension represents time steps, and the third dimension represents the number of features. Then, sequence data is taken as input, and feature extraction is carried out through one or more convolutional layers.

The convolutional layer can extract the spatial features of the input data. And the pooling layer is added after the convolutional layer to reduce the dimension of the feature map and retain important feature information. The pooling layer can be down sampled in spatial dimensions, such as maximum pooling or average pooling, and the pooled feature map is converted into a one-dimensional vector and input into the layer. In this way, it can use the spatial features extracted by the convolutional layer and combine the timing information of the sequence data for modelling.

In the layer, multiple units can be set up to model timing information and capture long-term dependencies in sequence data and learn timing patterns in the data. After that, one or more fully connected layers can be added to output the final predicted results. The appropriate activation function and loss function can be selected according to the specific task requirements. It uses labelled data to train the model and perform hyper-parameter tuning, such as learning rate, batch size, network structure, etc.

The model can be optimized using conventional optimization algorithms. The trained model is evaluated using validation sets or test sets, and various evaluation metrics can be used to evaluate the performance of the model.

3.4 Component / Service / Model specification

In order better to distinguish the operation of the component from others in the system, the activities of the component are shown communicating between swim lanes. When data is available from the data store (lane WP1) a message is sent which is being watched for by the component in lane WP3. It is important to note that other ‘lanes’ or work package components may also be subscribed to listen for occurrences of this message. However, their consumption of that message data and their subsequent behaviour does not impact directly on Lanes WP1 or WP2 in this representation.

Figure 11 shows these relationships within the GREEN-LOG prediction and simulation system of WP2.

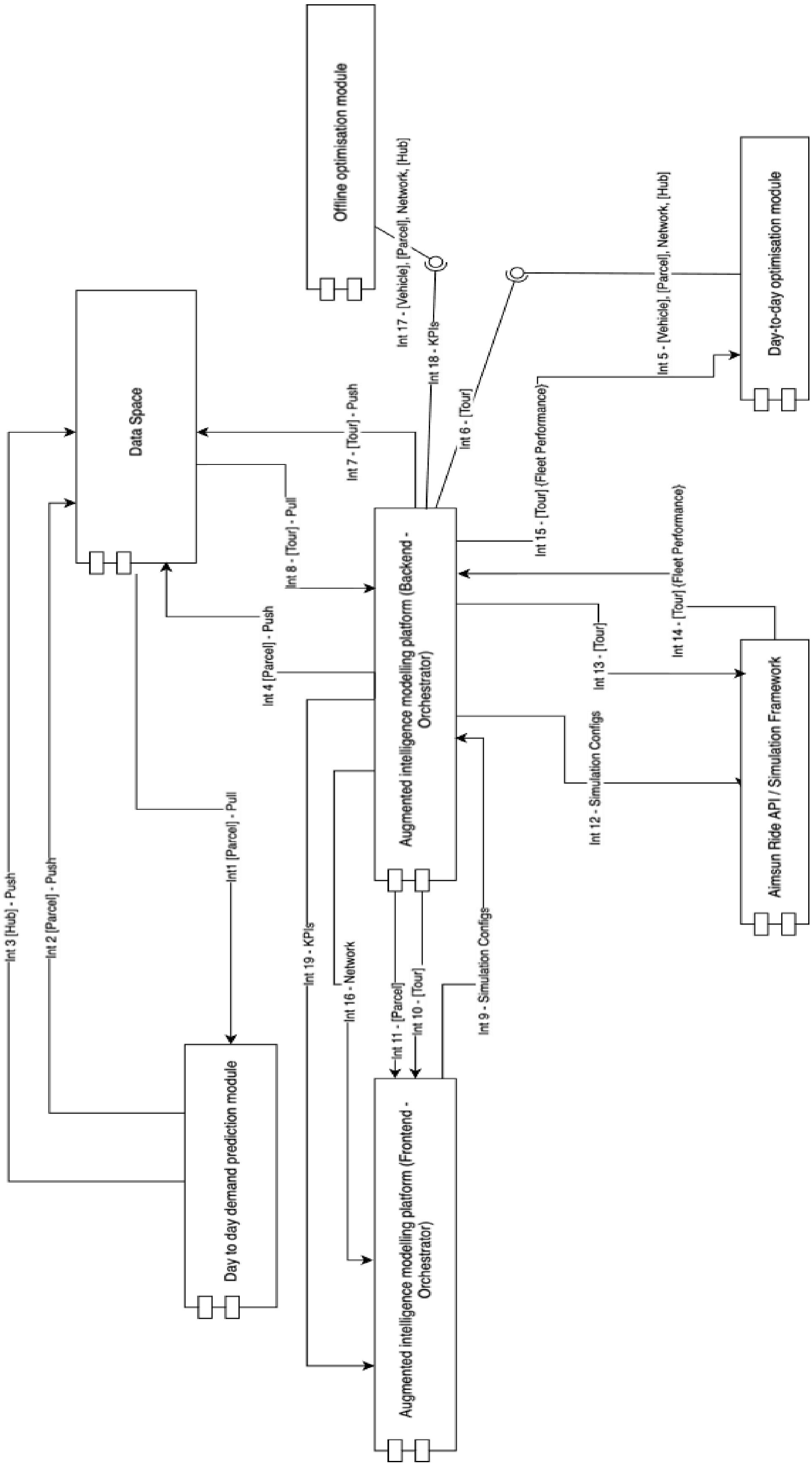


Figure 12 Component interactions in the GREEN-LOG WP2 prediction and simulation system.

As described in the original documentation for the spatio-temporal demand prediction services, the modelling processes for nowcasting and forecasting are implemented as services which both consume and deposit information from the central data store (Section 2).

The component which processes past parcel transport information and creates predictions of likely near-future parcel transport information is implemented as a service. However, the service cannot directly be accessed by other components of the GREEN-LOG system. As data becomes available on the Data Store, the spatio-temporal prediction service will consume that data, processes it, and return an updated data se to the data sore. At this stage the data store will notify any subscribers to the message, within the GREEN-LOG system that a process has run and updated data is available.

For brevity in the following discussion, we now describe the 'spatio-temporal demand prediction service' as the component. This component provides a service, accessible via the data store. This component implements a prediction model using algorithmic and data processing methods described in the previous sections (3.1, 3.2).

3.5 Implementation Approach / Description / Results

The CNN-LSTM model is tested on the dataset named 'icargo_archive_shipment' for the evaluation of its demand forecasting performance and the dataset is shown as below.

Table 1 The delivery dataset

Index	shipment_service_type	shipment_required_service_date	shipment_postcode	shipment_latitude	shipment_longitude
1	D	04/04/2022	OX4 4PE	51.723081	-1.2240178
2	D	04/04/2022	OX4 4PU	51.720039	-1.2252345
3	D	04/04/2022	OX2 8HW	51.789435	-1.2779975
4	D	04/04/2022	OX4 4YS	51.712314	-1.2189713
5	D	04/04/2022	OX4 2QG	51.739862	-1.2044678
6	D	04/04/2022	OX1 2HB	51.754159	-1.2633420
7	D	04/04/2022	OX3 0DN	51.764991	-1.2300986
8	D	04/04/2022	OX2 9NN	51.761385	-1.3513468
9	D	04/04/2022	OX2 0DG	51.753228	-1.2772390
10	D	04/04/2022	OX3 8HW	51.761693	-1.1918471

Then, the statistical analysis of the number of parcels on different dates in the dataset is performed and a histogram of the number of parcels by dates is plotted as below in Figure 13 and a geospatial data example of PEDAL deliveries is visualized in Figure 14.

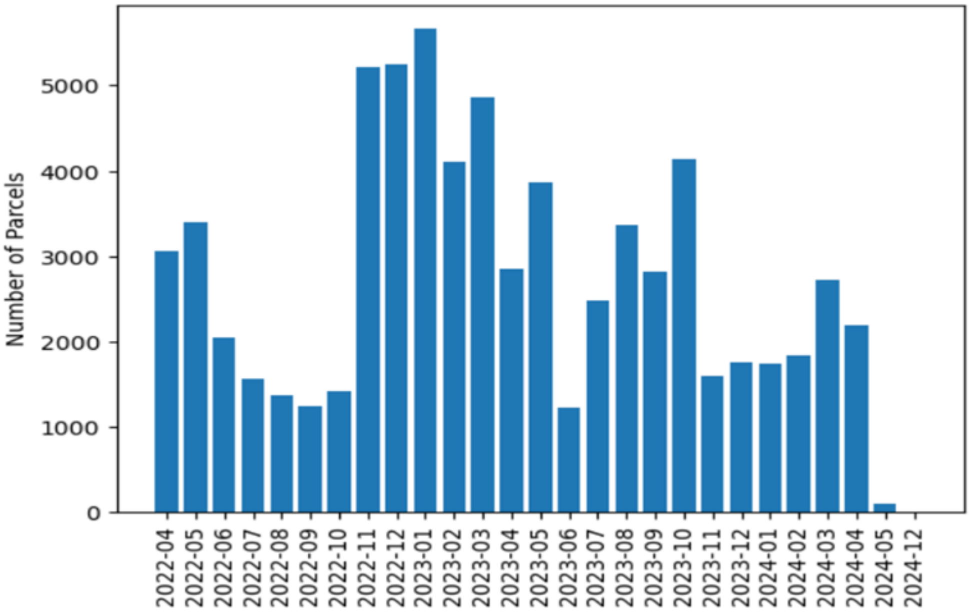


Figure 13 Number of parcels from the year of 2022 to 2024.

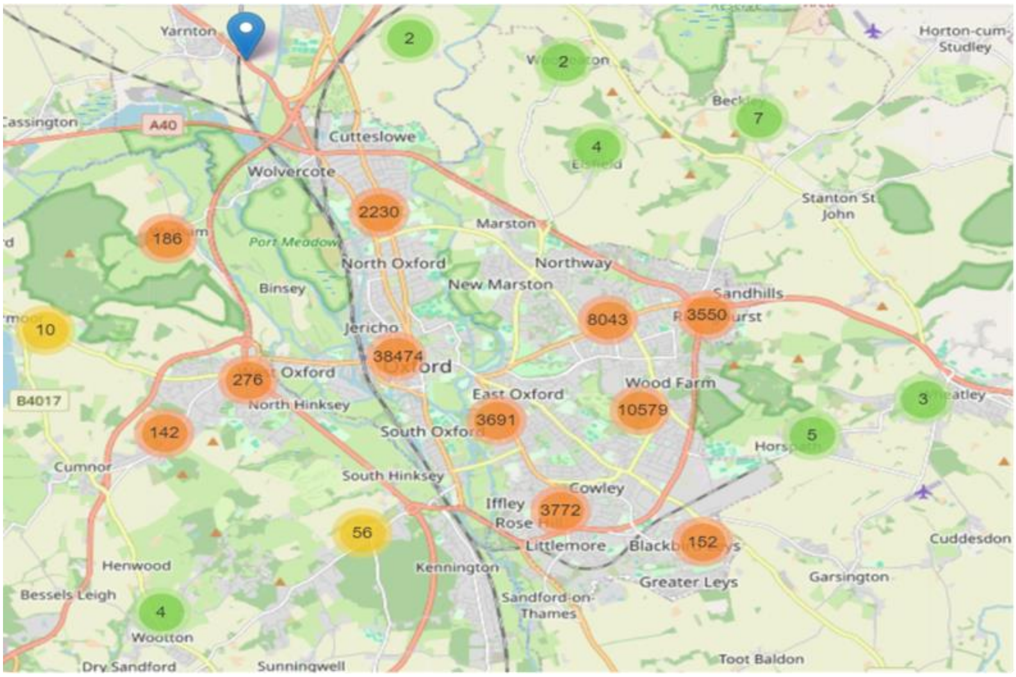


Figure 14 Geospatial data example of PEDAL deliveries.

The first experiment carried out to explore daily patterns in the dataset was to cluster days in 10 patterns using clustering (cf. clustering.ipynb). Figure 15 shows the results of the experiment per day and Figure 16 per weekday.

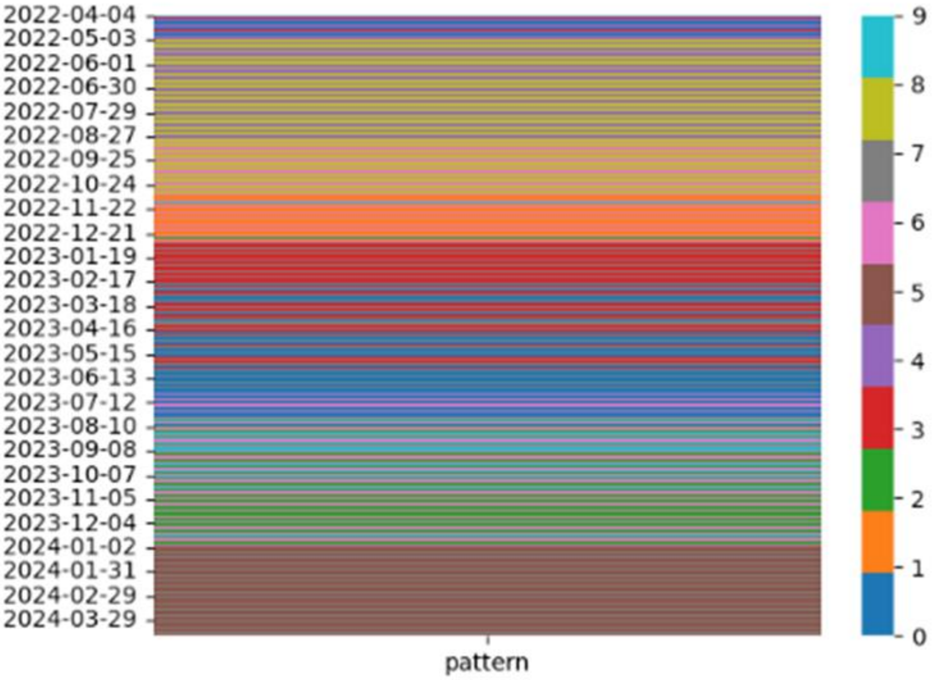


Figure 15 Daily patterns using clustering

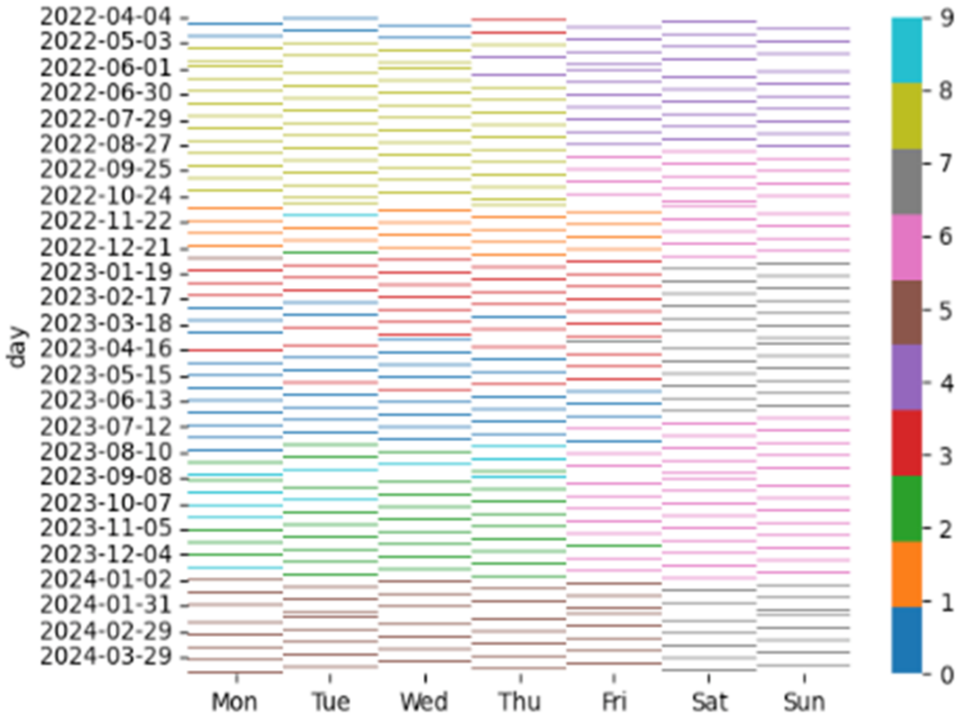


Figure 16 Weekday patterns using clustering

Similar patterns on Fridays and weekends are observed versus Monday to Thursday, especially in year 2022, and from July to December 2023.

Next, moving averages with and without exponential smoothing were explored (cf. arima.ipynb). Moving averages smooth time series data by removing random variations. As equal weights are assigned to all historical data points, this technique is usually preferred when trends and seasonality are absent, which seemed to be the case in PEDAL data. As this technique neglect actual ups and downs in data, results need to be carefully interpreted and customized to the actual reality of the problem.

Figure 16 shows the output of a simple moving average using the previous ten days to each forecasting horizon of 5 days. The plot compares real data in blue to the forecast in orange which is conservative as expected and does not fit well to days with very high or low demand.

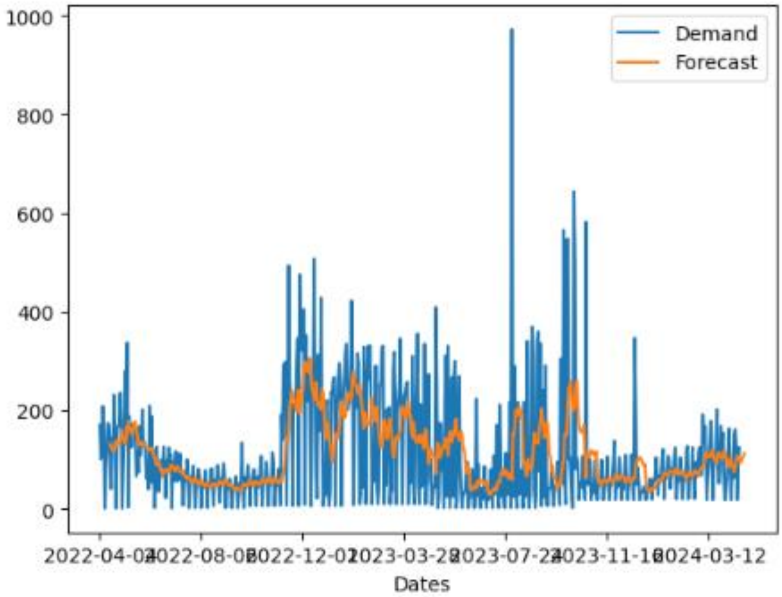


Figure 17 Results using a simple moving average

Figure 17 shows the output of exponential smoothing using an alpha of 0.4, which seems to be a promising method due to the high variability of the data. It should be noted though that high peaks are not anticipated using this method.

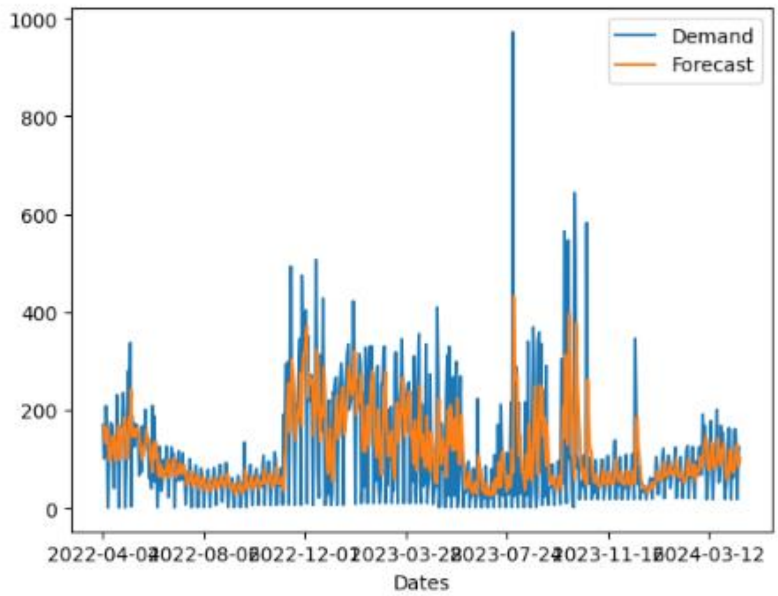


Figure 18 Results using exponential smoothing

Two machine learning techniques have been tested as the most popular regression methods for time-series forecasting: random forest and XGBoost regressors (cf. basic_ml.ipynb). The dataset has been split into train and test using 01-03-2024 as turning point. Both algorithms use the past 10 days as features for the forecast and predict 5-time horizons ahead. Figure 18 shows the compared output of these two methods to the truth data for horizons 1 to 5 respectively for each timestamp.

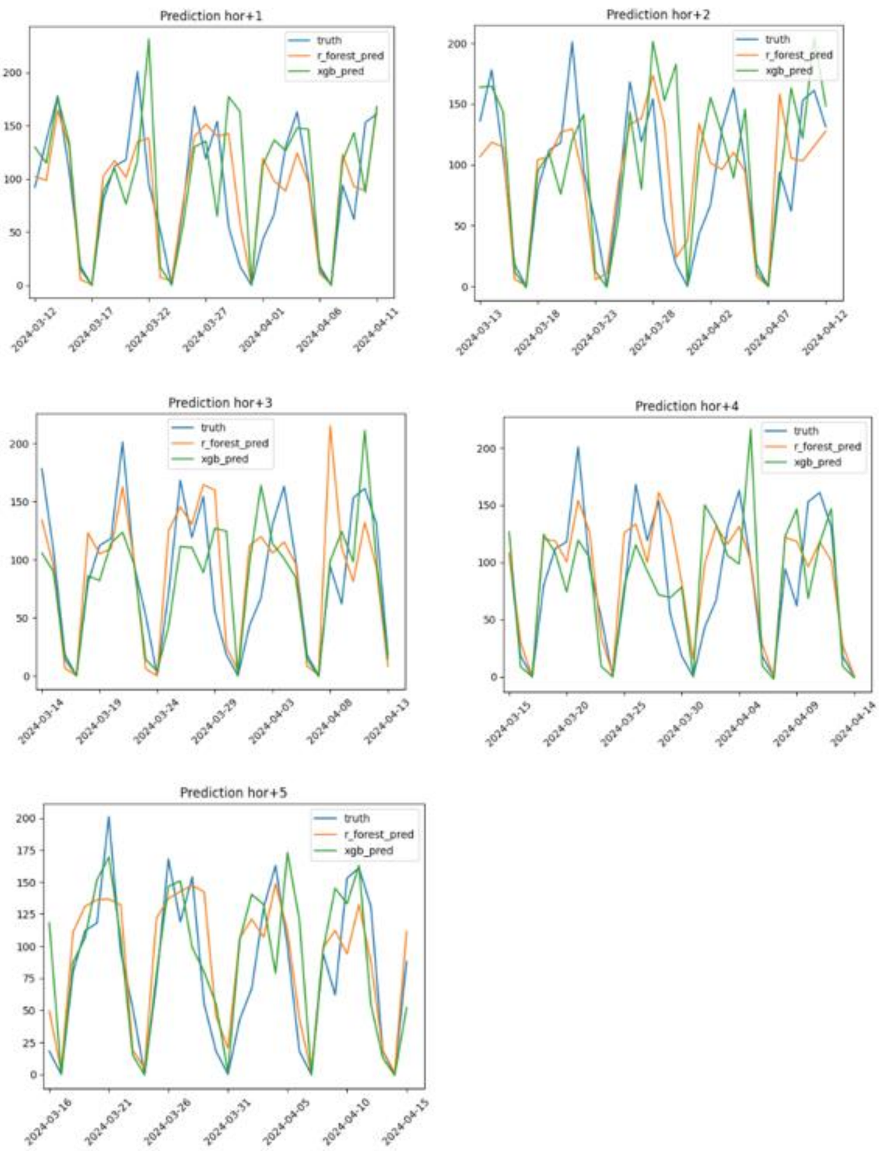


Figure 19 Comparison of results using random forest and XGboost regressors at 5-time horizons

Finally, we have tested a deep learning (DL) framework even though the amount of data is quite limited for DL methodologies. We have used a long-short term memory (LSTM) architecture with 36 units and than activation (cf. lstm_1.ipynb). Figure 20shows the output of the forecast.

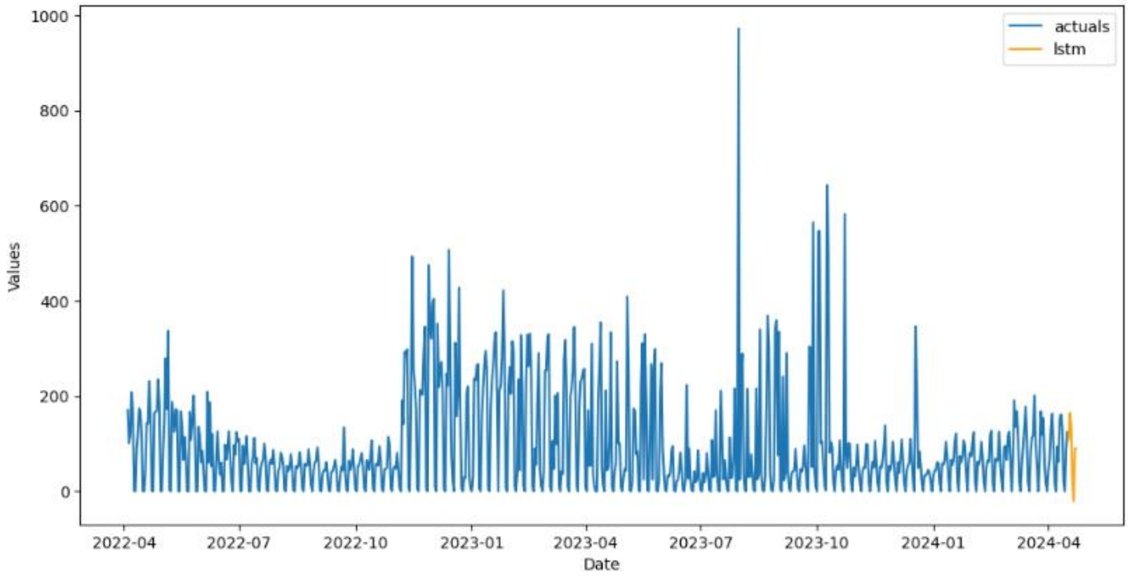


Figure 20 lstm prediction results

As DL architecture are data-hungry algorithms, we decided not to split the dataset. To have a more accurate interpretation of these results, and as the project lasts long enough to be able to collect more data, we expect that in the future once more data is available it can be used to analyze the accuracy and appropriacy of all these methods.

3.6 Conclusion and Future Work

In this work by UoW and AIMSUN, we have designed and studied logistics demand datasets with real-life characteristics, and then tested different efficient demand models and analysed their performance on our designed datasets.

We have run several preliminary experiments on the available data from PEDAL to get insights on the problem that needs to be tackled and start having an overview of different ways to provide insights which may be useful for delivery services. We need to, first, confirm that all data gathering from PEDAL is automated so that the implementation of a potential analytics service can be done. Then, once data providers allow communication of data feeds, a protocol needs to be established which complies with GDPR standards. Once data verification is successful, experiments need to be re-run to have a complete view and analysis of their outcome.

In this preliminary report, we have observed data inconsistencies which can be interpreted as data gaps from manual interference or merging of data sources or actual shifts in demand that could be recurrent. In case the second hypothesis applies, collecting more data from historical periods could prove this trend or reject it and hence a decision on whether peaks should be removed needs to be taken.

With respect to statistical and machine learning methodologies, preliminary results show that traditional regression techniques such as random forest or XGboost are adequate to tackle aggregating demand forecasting for 5-day time horizons. Moving averaged and deep learning are discarded due to their limited capabilities or demanding requirements.

We also had a detailed summary of our main inquiries about the current and desired states of demand prediction within the project, and then conducted an in-depth discussion with our logistics partner PEDAL, our project team, and other partners. In the next step, based on our discussion with our project partners, we will do further improve our logistics demand models and generated demand datasets in order to make it further closer to the real and complex environment of our project, and also carry on more investigations in order to gain better understanding about the operational details, demand patterns, forecasting needs, data limitations, and metrics, and further refine our demand prediction model effectively, and continue to maintain close communication with our project partners and obtain a further insight of both the current operational state and the aspirations post-project completion, facilitating a more targeted and efficient approach to demand prediction modelling in our logistics demand project, and we further apply our built model on real-life big dataset and test our prediction model on the real-life PEDAL dataset.

Moreover, we will do a detailed analysis of the PEDAL dataset and separate important information from useless information inside the large-scale dataset and do some adjustment of our prediction model, in order to further improve its performance on the real-life large-scale demand dataset.

4 Micro-Consolidation Centres (MCC) deployment and load pooling strategiser

4.1 Background / State of the Art

The constant effort of Logistics Service Providers (LSPs) and local authorities to minimise travel times of last-mile urban operations has conceived novel concepts in facility location strategies. Given that the capacities of the vehicles which conduct last-mile operations are limited, each one of them performs multiple trips from inventory facilities and back; at the same time, large depots are usually located in a considerable distance from the centre of cities, which concentrates the majority of demand. To tackle the challenges of this problem, recent studies in location-routing problems are examined [REF-04]. Several of them focus on the concept of mobile depots, as presented in the Mobile Facility Routing problem [REF-02].

The combined problem of regular routing with the integration of mobile depots is presented in literature as Vehicle Routing Problem with Time Windows and Mobile Depots (VRPTWMD). The concept does not significantly differ from a regular routing setting: the vehicles start from a location, namely the depot, from which the parcels to be delivered are picked-up, then each vehicle performs a route to deliver the loaded parcels. After the delivery of all parcels is completed, the vehicles return to the depot to load new parcels or drop-off unserved parcels. It is clearly seen that if this depot is mobile, then it can be transferred to locations which are close to clusters of demand, hence minimising the travel times of the vehicles. However, the applicability of the concept is far from trivial, since the mobile depot will remain at a certain location for a restricted time window, meaning that all local operations must be carried out in limited time.

Beyond regular routing, the capabilities of mobile depots extend the applicability of new transport modes on last-mile operations. We highlight the usage of mobile depots as drones stations for last-mile deliveries, as presented by [REF-04], and the combination of mobile depots with crowd shipping models [REF-06] as indicative novelties which have been brought forward recently.

For the use-cases of the Living Labs of Green-Log, we assume that a fixed location, namely the Micro-Consolidation Centre (MCC), is loaded with the parcels of the daily last-mile operations. A (fleet of) mobile depot(s) will start from the MCC in the beginning of the daily shift, loaded with as many parcels as its capacity allows. Each stop of a mobile depot has a predefined waiting time, which is depended on the volume of demand which will be locally served. In particular, a fleet of last-mile operations will pick-up parcels from the mobile depot, then each vehicle will complete the deliveries of the parcels to the clients, before returning to the mobile depot for reloading, for drop-offs of unserved parcels or parcels which should be picked-up by different locations than the MCC. These operations should be carried out within the indicated time window of the mobile depot in the local stop. If the capacity of the mobile depots is not adequate to cover the entire daily demand in a single run, then these should return to the MCC for reloading.

Access to the software code for T2.3 is available online at: https://github.com/aueb-greenlog/opt-service

4.2 Methodological Approach

4.2.1 Partitioning approach for VRPTWMD

The complexity of the VRPTWMD is too high to allow the execution of a holistic mathematical formulation which addresses to the entire problem. In response, we opt for a partitioning approach, meaning that the optimisation problem is divided into two stages: the MCC deployment stage, in which the mobile depots are scheduled, and the parcels are assigned to selected stops, and the (re)routing stage, in which local last-mile operations are scheduled for each stop. The MCC deployment stage, which is examined in this deliverable, is applicable on most of the use-cases of Green-log; for the rest of them, a direct implementation of the (re)routing stage is designed (see D3.1 for details).

Each stage is solved by different optimising methods; therefore, the methods which address to each stage should interact: the outcome of MCC deployment should be provided to the (re)routing service, then the routing plan should modify the MCC deployment schedule, if additional delays appear. The following scheme demonstrates the internal flow diagram of exchange of knowledge between the optimisation stages for the Day-to-Day optimisation module:

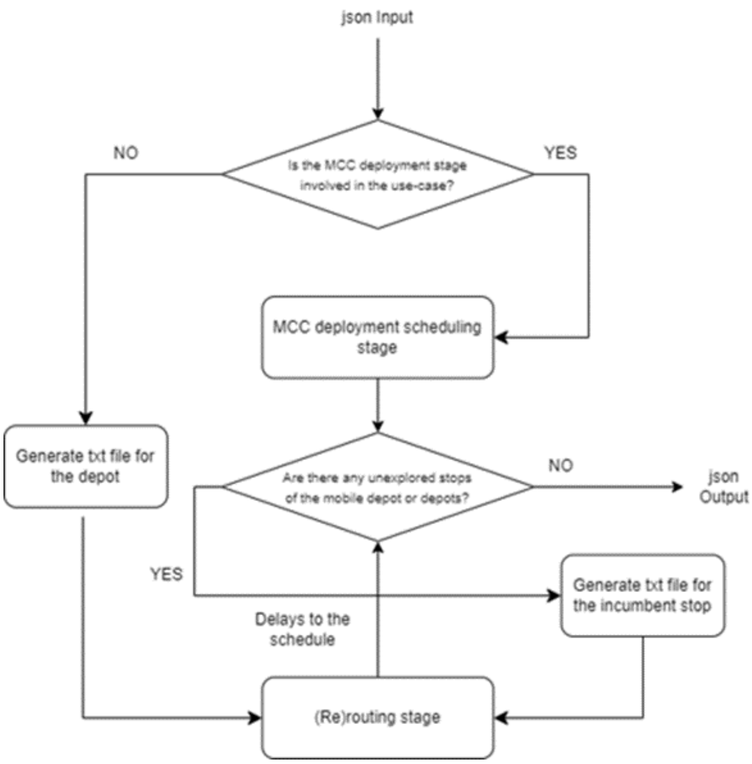


Figure 21 Flow diagram of internal interactions in the Day-to-Day optimization module

If the MCC deployment stage is involved, then for each stop of the mobile depot, a txt file of a predefined format is generated. The txt file includes the input which is required for the (re)routing stage (i.e., travel times between all involved nodes, capacities of vehicles, time windows etc). Since these details mostly concern the (re)routing stage, they are not elaborately described in this deliverable, which is focused on the MCC deployment stage. The MCC deployment stage is expected to be implemented on the use-cases of Oxfordshire, one use-case of Barcelona, one use-case of Athens, and the use-cases of Ispra.

4.2.2 Optimisation method

The MCC deployment stage is a part of the Day-to-Day optimisation service. Therefore, it is expected that the required input data will be available several hours before the actual implementation of the daily schedule, thus allowing us to consider an exact-based approach, using a Mixed-Integer Linear Program (MILP). This capability is critical for the efficiency of the service, since the MCC deployment schedule is strictly constrained, therefore the integration of all restrictions would not allow us to conceive a heuristic method.

For the MILP, the following mathematical annotations will be used:

Table 2 Mathematical annotations of MILP

Sets		
I	Orders	
J	Locations	
D	MCC locations	
H	Candidate stops	
K	Mobile depots	
C^k	Copies of the same mobile depot k	
Parameters		
t^{MD}, t^{LM}	Type of vehicle (MD for mobile depot, LM for last-mile)	
t^T_{ijt}	$\forall i \in J, j \in J, T \in \{MD, LM\}$	Travel time of trip $i \rightarrow j$, using a vehicle of type T
i^p, i^d	$\forall i \in I$	Pickup (for p) or Delivery (for d) location of order i
$minInterval$	Minimum time interval (e.g., if $minInterval = 15$, then all times are integer products of 15)	
t^r_i, t^l_i, t^u_i	$\forall i \in I$	Release (r) time, lower bound (l) of time window, upper bound (u) of time window for order i
V	A very large number	
s	Service time	
w_i, v_i	$\forall i \in I$	Weight and volume of order i
W_k, V_k	$\forall k \in K$	Maximum weight and volume of mobile depot k
$minW$	Minimum waiting time for reloading	
Variables		
$l_{ijk} \in \{0,1\}$	$\forall i \in I, j \in J, k \in K$	1 if mobile depot k does the trip $i \rightarrow j$, 0 otherwise
$xPickup_{ijk}, xDelivery_{ijk} \in \{0,1\}$	$\forall i \in I, j \in J, k \in K$	1 if order i is picked-up/delivered at stop j of mobile depot k, 0 otherwise
$y_{jk} \in \{0,1\}$	$\forall j \in H, k \in K$	1 if candidate stop j is visited by mobile depot k, 0 otherwise
$Arrival_{jk}, Departure_{jk}, Waiting_{jk}$ integers	$\forall j \in J, k \in K$	Arrival, departure and waiting time of mobile depot k at stop j
$t_k \in \{0,1\}$	$\forall k \in K$	1 if mobile depot k is deployed, 0 otherwise

The mathematical model of the MILP is the following one:

$$
\begin{array}{lll}
\min z & & \\
\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} xPickup_{ijk} = |I| & & (1)\\
\sum_{i\in I}\sum_{j\in J}\sum_{k\in K} xDelivery_{ijk} = |I| & & (2)\\
\sum_{j\in J}\sum_{k\in K} xPickup_{ijk} \leq 1 & \forall i \in I & (3)\\
\sum_{j\in J}\sum_{k\in K} xDelivery_{ijk} \leq 1 & \forall i \in I & (4)\\
xPickup_{ijk} + xDelivery_{ijk} \leq 1 & \forall i \in I, j \in J, k \in K & (5)\\
xDelivery_{ijk} \leq y_{jk} & \forall i \in I, j \in H, k \in K & (6)\\
xPickup_{ijk} \leq y_{jk} & \forall i \in I, j \in H, k \in K & (7)\\
minInterval \cdot Arrival_{jk} + t_{ji^{d}t^{LM}} - t_i^u \leq V \cdot (1 - xDelivery_{ijk}) & \forall i \in I, j \in J, k \in K & (8)\\
minInterval \cdot Waiting_{jk} \geq (t_{i^{d}jt^{LM}} + t_{ji^{d}t^{LM}} + s) \cdot xDelivery_{ijk} & \forall i \in I, j \in J, k \in K & (9)\\
minInterval \cdot Waiting_{jk} \geq (t_{i^{d}jt^{LM}} + t_{ji^{d}t^{LM}} + s) \cdot xPickup_{ijk} & \forall i \in I, j \in J, k \in K & (10)\\
t_i^l - t_{ji^{d}t^{LM}} - minInterval \cdot Departure_{jk} \leq t_i^l \cdot (1 - xDelivery_{ijk}) & \forall i \in I, j \in J, k \in K & (11)\\
t_i^r + t_{i^{p}jt^{LM}} - minInterval \cdot Departure_{jk} \leq V \cdot (1 - xPickup_{ijk}) & \forall i \in I, j \in J, k \in K & (12)\\
Departure_{ik} - Arrival_{jm} \leq V \cdot (2 - xPickup_{ilk} - xDelivery_{ijm}) & \forall i \in I, j \in J, k \in K, l \in J, m \in K & (13)\\
l_{jjk} = 0 & \forall j \in J, k \in K & (14)\\
\sum_{i\in H} l_{k^{0}ik} = t_k & \forall k \in K & (15)\\
\sum_{i\in H} l_{ik^{0}k} = t_k & \forall k \in K & (16)\\
\sum_{i\in J} l_{ijk} = y_{jk} & \forall j \in H, k \in K & (17)\\
\sum_{i\in J} l_{jik} = y_{jk} & \forall j \in H, k \in K & (18)\\
Departure_{jk} \geq Arrival_{jk} + Waiting_{jk} & \forall j \in H, k \in K & (19)\\
minInterval \cdot Departure_{jk} + t_{jit^{MD}} - minInterval \cdot Arrival_{ik} \leq V \cdot (1 - l_{jik}) & \forall j \in J, i \in J \setminus \{j\}, k \in K & (20)\\
minInterval \cdot Arrival_{ik} - (minInterval + 1) \cdot Departure_{jk} - t_{jit^{MD}} \leq V \cdot (1 - l_{jik}) & \forall j \in H, i \in J \setminus \{j\}, k \in K & (21)\\
t_k \geq y_{jk} & \forall j \in H, k \in K & (22)\\
\sum_{i\in I}\sum_{j\in J} w_i \cdot xPickup_{ijc} \leq W_c & \forall k \in K, c \in C_k & (23)\\
\sum_{i\in I}\sum_{j\in J} v_i \cdot xPickup_{ijc} \leq V_c & \forall k \in K, c \in C_k & (24)\\
Departure_{jc} \geq Arrival_{jc-1} + minW & \forall j \in D, k \in K, c \in C_k & (25)\\
minInterval \cdot Departure_{jc} \geq minInterval \cdot Arrival_{jc-1} + minInterval \cdot Waiting_{jc} & \forall j \in D, k \in K, c \in C_k & (26)
\end{array}
$$

with the objective function being:

$$
z \geq \sum_{i\in J}\sum_{j\in J}\sum_{k\in K} t_{ijt^{MD}} \cdot l_{ijk} + \sum_{i\in J}\sum_{j\in J}\sum_{k\in K} (t_{i^{p}jt^{LM}} + t_{ji^{p}t^{LM}}) \cdot xPickup_{ijk} + \sum_{i\in J}\sum_{j\in J}\sum_{k\in K} (t_{i^{d}jt^{LM}} + t_{ji^{d}t^{LM}}) \cdot xDelivery_{ijk}
$$

The objective function is the minimisation of an auxiliary variable, which is greater/equal than the sum of travel times of the mobile depots and the estimated travel times for the pickups/deliveries of the assigned parcels. Constraints (1) and (2) ensure that the total pickup and delivery operations will be equal with the number of parcels, and constraints (3)-(4) ensure that each order will be picked-up and delivered at exactly one location by exactly one mobile depot. The combination of constraints (1)-(4) will imply the assignment of each order to one location/one mobile depot for its collection and its drop-off. Constraints (5) prevent the assignment of both pickup and delivery operations for the same parcel to the same stop of a mobile depot. This restriction will be deprecated in the near future; it is currently included because the (re)routing service does not support the required precedence constraints (i.e., if the pickup and delivery of the same parcel are assigned to the same stop, the pickup should precede of the delivery). Constraints (6)-(7) ensure that if any pickup or delivery is assigned to a location (except for MCC locations), then this location will be visited by the respective mobile depot.

Constraints (8) ensure that each parcel will be assigned to a stop for which its delivery can be completed before the upper bound of its time window. Constraints (9)-(10) set an adequate waiting time of the mobile depots at each stop: the time interval is sufficient for all last-mile vehicles to start from the stop, pick-up or deliver any parcel and come back to the stop, before the mobile depot departs. Constraints (11)-(12) ensure that the lower bound of the time window of each delivery will not be violated. Precedence constraints (13) assign the pickup of each parcel to a stop which is visited earlier than the stop which is involved with the respective delivery.

Constraints (14) restrict a trip between identical locations. By (15)-(16), if a mobile depot is deployed, then it will start and end its route from its current location (which is the MCC location). By constraints (17)-(18), if a location (except for the MCC) is visited by a mobile depot, then this location will be the origin and the destination of a trip. Constraints (19)-(21) define the departure and arrival times of each mobile depot to any stop, according to the travel times of trips and the determined waiting time. By (22), if any location is visited by a mobile depot, then this mobile depot is deployed. (23) and (24) are capacity constraints for the total weight and total volume of the mobile depots. Last, constraints (25)-(26) ensure that multiple trips of the same mobile depot will not overlap, and that the reloading procedure has a minimum duration.

4.2.3 Warm-start solution

The MILP presented above is too constrained. If the time windows of deliveries are too tight and diverse, the computation of a near-optimal, or even any feasible, schedule in reasonable time is challenged. To ensure that the optimisation service will always provide a daily schedule, regardless of the complexity of the input dataset, we compute a warm-start solution, by solving an easier optimisation problem: the maximisation of covered demand.

We replace the objective function with the following expressions:

$$\max z$$

$$z \geq \sum_{i \in I} \sum_{j \in J} \sum_{k \in K} (xPickup_{ijk} + xDelivery_{ijk})$$

As we notice, the new objective attempts to maximise the total number of served orders. This objective allows us to omit constraints (1)-(2), which are critical in the increase of the complexity of the problem. If all orders can be feasibly served, then a feasible schedule can be easily obtained, hence being induced to the original MILP as a warm-start solution. If there are any orders which cannot be served (e.g., the time windows are too tight to allow a completion of the delivery in time), then the objective of the warm-start MILP will imply a value which is less than the total number of orders. The new value will be considered in constraints (1) and (2), allowing the computation of a feasible daily schedule, replacing $|I|$.

4.3 Component / Service / Model specification

4.3.1 Use-cases

The use cases of the shell are listed below:

- Use Case 1 (UC1): Authenticate user.
- Use Case 2 (UC2): List the available optimisation services.
- Use Case 3 (UC3): Submit a new optimisation job.
- Use Case 4 (UC4): Get the solution of a completed optimisation job.
- Use Case 5 (UC5): Store the solution of a completed optimisation job.

The listed functional requirements are also demonstrated in the image below:

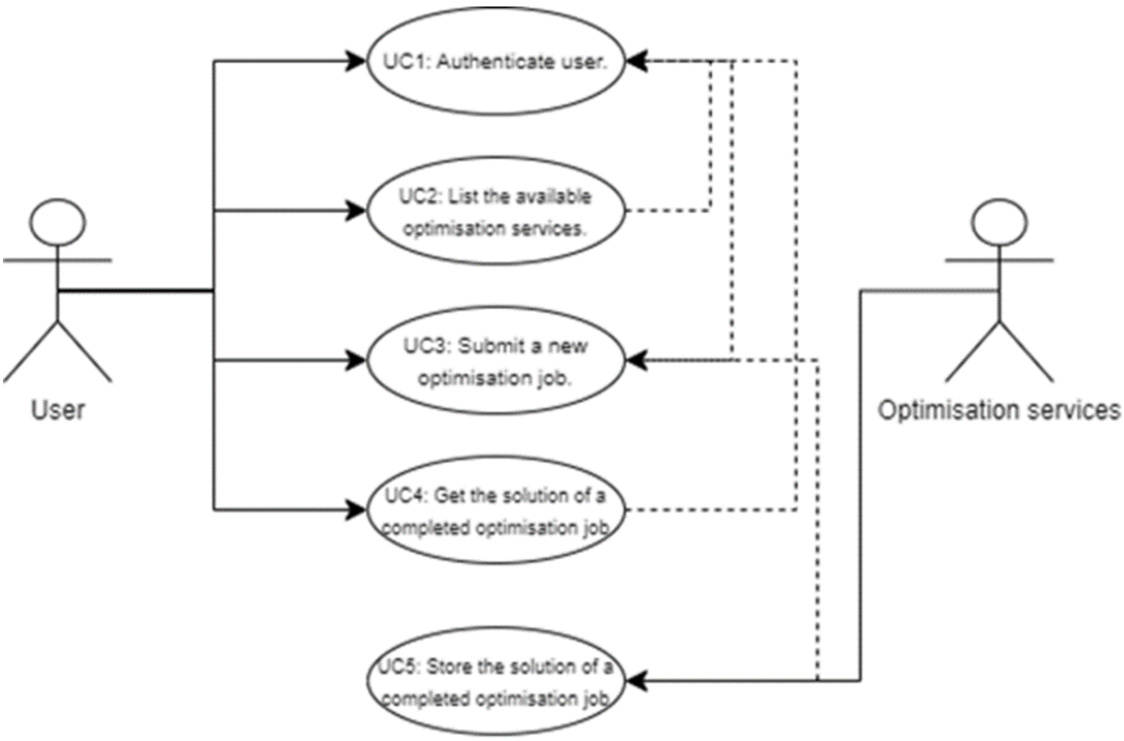


Figure 22 optEngine Functional Requirements

The following class diagram demonstrates the data requirements of *optEngine*:

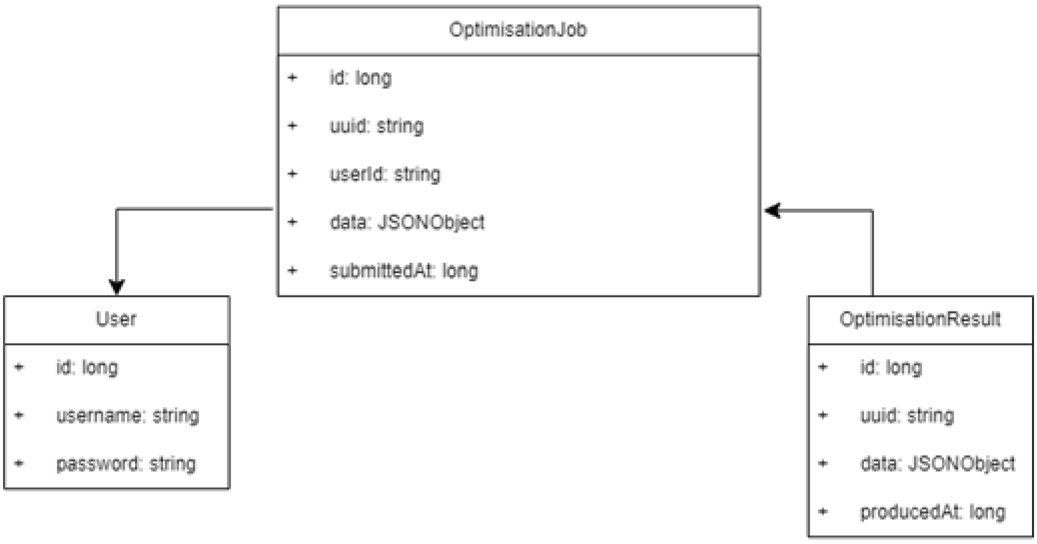


Figure 23 optEngine Data Requirements

4.3.2 Architectural view

OptEngine works as a shell around the optimisation. Its architecture follows an asynchronous approach and is agnostic to optimisation-specific data requirements. That is, *optEngine* receives and forwards the optimisation data to the optimisation service requested by the end-user, then it stores the computed solution to the Dataspace.

Optimisation requests along with the respective data are received via a web API. The communication with the API requires authentication, is encrypted (https) and asynchronous, meaning that the user does not have to wait for the completion of a submitted optimisation job.

Data stores within *optEngine* work in a twofold manner:

Permanent storage via a dataspace: the requests and the related data are permanently stored or retrieved and updated, when necessary, from a Dataspace. For example, the Day-to-Day optimisation module stores the acquired solution to the Dataspace. If a disruption occurs during the implementation of the daily schedule, the Event-triggered optimisation module retrieves the stored solution to update it.

Temporal storage via the use of queues: optimisation data are stored to the Dataspace until they are retrieved from the optimisation services that read these queues.

Regarding the optimisation data, both the dataspace and the queues are data-agnostic following a general json schema. This allows the storage, permanent and temporal, of different data structures required from different optimisation services.

The employed queues allow the asynchronous processing of an optimisation job. Additionally, by being durable they ensure that when *optEngine* or an optimisation service fail, the job along with data are available in the respective queue. This means that *optEngine*, upon reception of a new optimisation job, forwards it to the requested optimisation service via a queue. Each optimisation service listens for a new optimisation job to a specific queue and writes status/progress updates to another queue. Last, *optEngine* listens to (a) a queue for status/progress updates and (b) multiple queues for optimisation results.

The flow of data and the architectural approach of *optEngine* are depicted below:

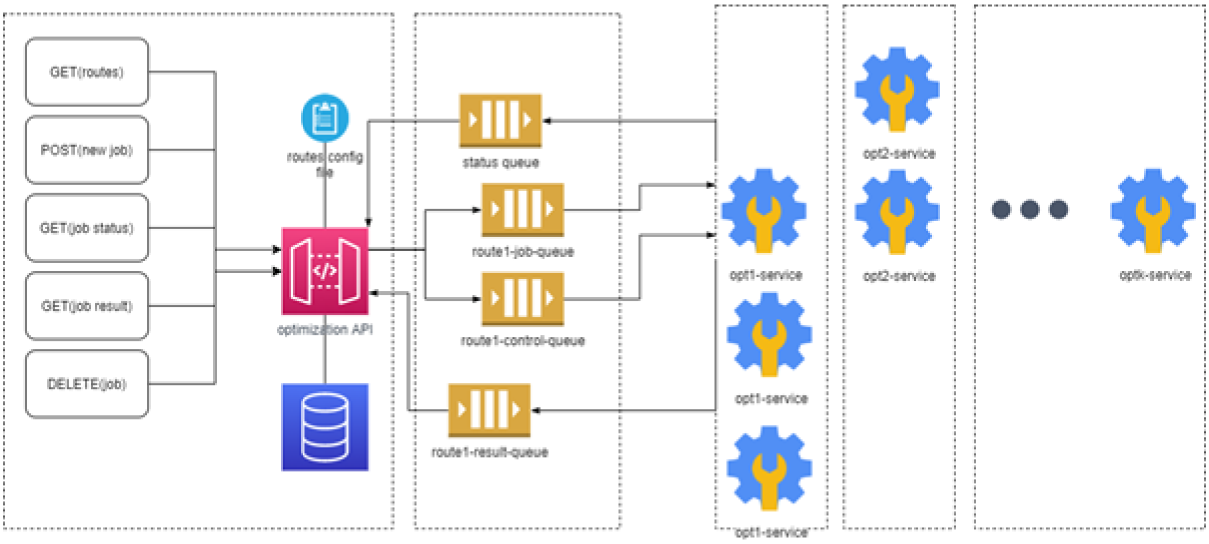


Figure 24 optEngine Data Flow

4.3.3 Technology stack

The following technologies are used for the development of *optEngine*:

Table 3 optEngine Data Stack

Java 8	Java 8
Spring-boot 2.4.5	spring boot
Springdoc openAPI 1.5.2	Open API Specification Swagger
Hibernate 1.0.0	HIBERNATE
PostgreSQL 11	PostgreSQL
RabbitMQ 3.8.16	RabbitMQ

Docker 18.09.7	

The optimization engine has been originally designed for the 2020 Horizon project FACTLOG (Grant agreement: 869951). An elaborate description of *optEngine* is presented in D5.1[2].

4.4 Implementation Approach / Description / Results

4.4.1 Athens, Greece

The optimisation engine will be implemented in Athens for both indicated use-cases, but the MCC deployment stage is involved in the second one only. The white-label van is used as a mobile depot, which will follow a route of stops during the daily shift, starting from Klafthmonos square. The schedule of the mobile depot is determined by the MCC deployment stage; hence it is out of the scope of this deliverable. Nevertheless, for each stop of the mobile depot, the (re)routing service will compute the routes of the vehicles which are responsible for last-mile operations (assumingly, scooters).

To address each parcel to the responsible operator, we generate a pair of duplicate txt files for each stop of the mobile depot; each copy will consider demand points and vehicles of only one operator. When the routing plan of each stop is completed for both operators, the results are consolidated to compute the actual delay of the mobile depot (i.e., the maximum delay of the two routing plans).

As mentioned above, the MCC deployment stage is implemented on the second use-case. We consider a dataset with the following attributes:

Area of deployment: Historical centre of Athens

Location of MCC/Depot: Klafthmonos square

Number of deliveries: 100 – randomly assigned to ACS (70% probability) or Fedex (30% probability)

Number of mobile depots: 1 (i.e., the white-label van is the mobile depot).

Number of last-mile vehicles: 10 scooters – randomly assigned to ACS (70% probability) or Fedex (30% probability).

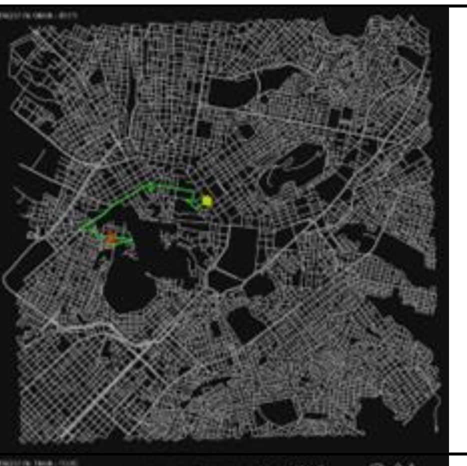
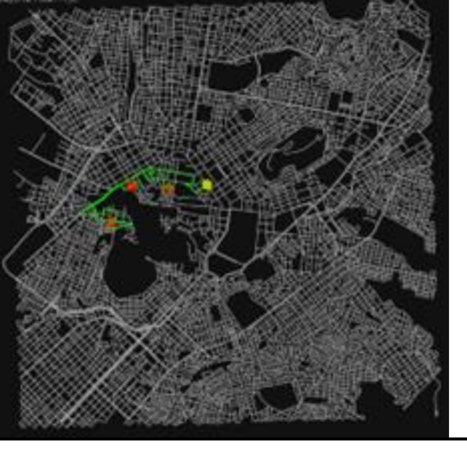
Capacities of vehicles: Estimated for realistic assumptions per type of vehicle

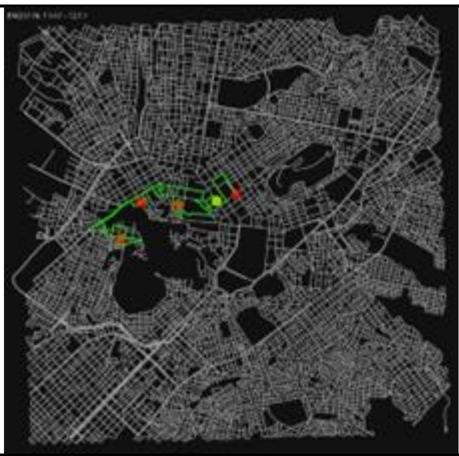
Candidate stops: 10 randomly selected and well-distributed within the area of interest

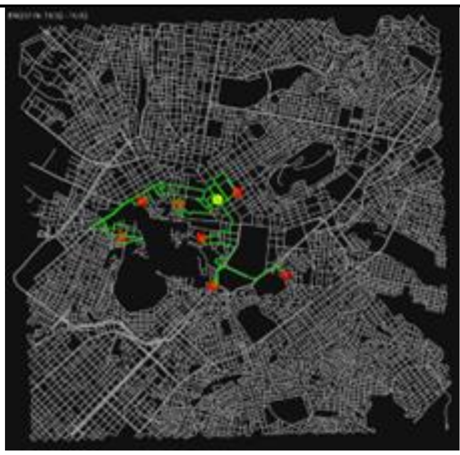
Travel times: Obtained by OpenStreetMap, using the OSMNX [REF-03], for a radius of 3000 metres from the centre of Athens

The schedule of the mobile depot in Athens for the above dataset is the following one:

Table 4 MCC deployment results for the second use-case in Athens

Time window	Daily route
08:00 (Start of shift)	
09:15 – 10:00	
10:30 – 11:00	

11:30 – 11:47	
12:17 – 13:02	
13:12 – 13:47	
14:17 – 14:32	

15:02 – 15:32	

The red nodes indicate the stops of the mobile depot, and the yellow node is the location of the MCC. It is noted that the schedule of Table 3 has also considered the (re)routing stage which succeeds (see D3.1 for more details). If the determined time window of a stop is not adequate for all pickups/deliveries to be completed in time, then it will be extended. In this case, the time window of the fourth stop (11:30 – 11:47) has been originally set to be terminated at 11:45, but a delay of 2 minutes has been added later.

4.4.2 Barcelona, Spain

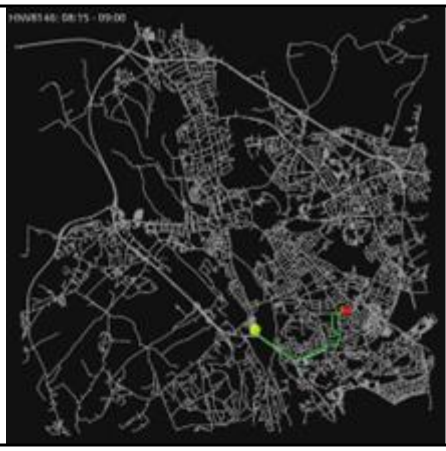
In Barcelona, we examine the combination of last-mile delivery operations using public transport with regular routing. The use-cases are divided into “All-the-way” routing using the fleet of VANaPED, multimodal delivery using the rail network of FGC, and a combination of these two. For this stage of implementation, we examined the third scenario, as it is inclusive for all applied cases.
Using public transport is expected to imply additional travel time: each carrier should either walk or ride a bike to the station in Plaça de Catalunya (for the latter case, it is possible that additional delays will be brought forward if there are no free parking spaces for bikes), then use the next trip to the area of interest (meaning that external delays in the timetable will affect the daily plan), and finally complete the deliveries on foot. Nevertheless, there are several reasons why examining the multimodal deliveries may bring forward potential profits for the operators:
no resources are consumed: the fleet of vehicles remain available for other delivery requests accessibility to distant districts: the elevation of the city does not ensure accessibility of all districts by bike. Also, distant areas (e.g., in Terrassa) are not easily accessible by electric vehicles, as the recharging requirements will be higher.
We observe that the MCC deployment stage is not required for any use-case. However, we can use the MILP of MCC deployment stage to assign as many parcels as possible to multimodal trips, if we assume that each carrier is a different mobile depot, each station of FGC is a candidate stop and the depot of VANaPED is the MCC location. Slight modifications are integrated to include the walk catchment area constraints: if the distance of a delivery point from a candidate station is larger than the maximum walk catchment, then the assignment is not possible.

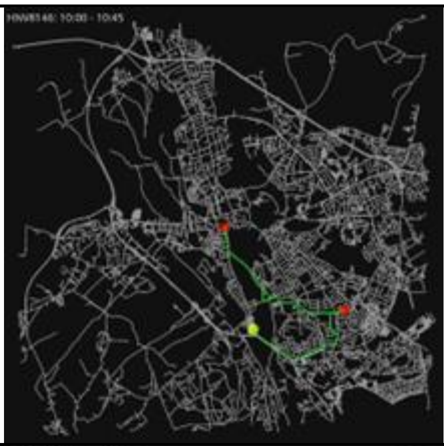
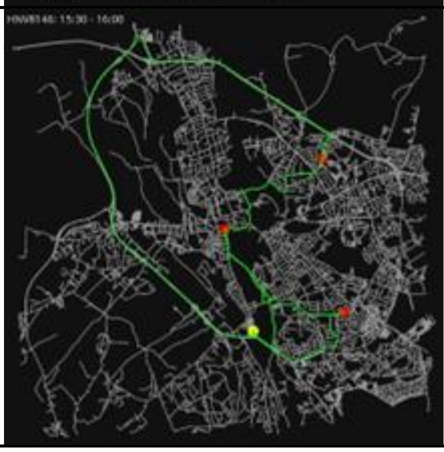
The results are elaborately presented in D3.1, with the routing of carriers and the other available types of vehicles.

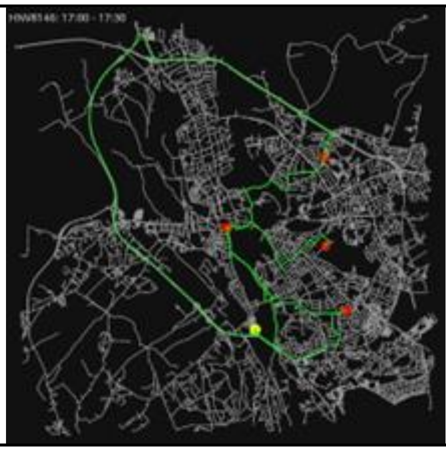
4.4.3 Oxfordshire, UK

In Oxfordshire, we examine the potential benefits of the usage of a mobile depot (namely, the Mobile Delivery Hub – MDH) to minimise the travel times of last-mile operations. It is expected that all parcels are stored in the MCC, which is assumingly the Redbridge Park-and-Ride. A mobile depot will collect parcels, then it will be deployed in the network of the city to facilitate the completion of last-mile operations. If the capacity of the mobile depot cannot cover the entire daily demand at once, then it will return to the MCC to replenish its load.
The first iteration of implementation in Oxfordshire considers the following dataset:
Area of deployment: Oxford
Location of MCC/Depot: Redbridge Park-and-Ride
Number of deliveries: 100 – picked up from the MCC (75% probability) or a random node in the network (25%), delivered to the MCC (25% probability only if not picked up from the MCC) or a random node in the network (75% probability if not picked up from the MCC, 100% probability if picked up from the MCC).
Number of mobile depots: 1
Number of last-mile vehicles: 10 cargo-bikes
Capacities of vehicles: Indicated by the stakeholders of the Living Lab
Candidate stops: 10
Travel times: Partially obtained by AIMSUN, completed by OpenStreetMap using the OSMNX [REF-03], for a radius of 3000 metres from the centre of Oxford
The schedule of the MDH is summarised above:

Table 5 MCC deployment results of Day-to-Day Optimisation in Oxford

Time window	Daily route
08:00 – 08:15 (Start of shift)	

09:00 – 10:00	
10:45 – 12:00	
12:30 – 14:15	
15:00 – 15:30	

16:00 – 17:00	

The yellow node indicates the MCC location, and the red nodes are the stops of the mobile depot.

4.4.4 Ispra, Italy

In Ispra, the bike is considered as a mobile depot, which will be scheduled by applying the MCC deployment stage. For each stop of the bike in the intermediate point, the fleet of autonomous droids will pick-up the lunchboxes from the bike and deliver them to the receivers. After the completion of each round of deliveries, the bike will depart from the intermediate point, re-load in the canteen and repeat the procedure.
Due to the fact that the area of deployment is the enclosed campus of Joint Research Centre (JRC), there are no open-source geographic data. After the network of interest has been indicated by the stakeholders, a network has been constructed from scratch. The speed of the droids has been estimated using historical data of e-Novia.
The rest of the parameters are as follows:
Area of deployment: JRC in Ispra
Location of MCC/Depot: Canteen
Number of deliveries: 30 – randomly assigned to the four candidate demand points
Number of mobile depots: 1 – the bike
Number of last-mile vehicles: 3 Yape droids
Capacities of vehicles: Indicated by the stakeholders of the Living Lab
Candidate stops: 1 – the intermediate point
Travel times: Estimated using input from e-Novia

The daily plan of the bike is the following one (Table 6).

Table 6 MCC Deployment results of Day-to-Day Optimisation in Ispra

Time windows	Daily route

11:00 (Start of shift)	
11:15 – 12:25	
12:40 – 12:50	
13:05 – 14:00	

4.5 Conclusions and Future Work

The first iteration of implementation is focused on the minimisation of travel times, since this objective relates to most of the key-performance indices of the project, e.g., the minimisation of emissions or the minimisation of operational costs. The following iterations will focus on more metrics of interest, as indicated by the stakeholders of the Living Labs. For example, the implementation will examine multiple candidate locations for the MCC or additional mobile depots for the Living Labs which will be facilitated by one mobile depot (i.e., Oxfordshire and Athens). In addition, to exploit the short duration of the current method, an extension of the methodological approach will consider an iterative procedure between the counterparts of optimisation service (i.e., between the MCC deployment and the re-routing services), meaning that the exchange of knowledge between the two stages will be executed iteratively, to examine multiple schedules and their impact on the minimisation of travel times.

5 Micro and meso simulations for urban freight

5.1 Background / State of the Art

Simulation methods are often adopted in analysis and assessment of transportation systems, such as urban last-mile logistics, mainly due to their ability to capture such systems' intrinsic complex, stochastic and non-linear dynamics [REF-22] – dynamics which analytical methods cannot easily and computationally efficiently address. Traditional and state-of-the-art optimization methodologies may in principle produce high quality solutions for complex problems in urban last-mile logistics systems (e.g., fleet scheduling/routing, facility locations, fleet sizing, etc.), however the combination of optimization and simulation may enhance the solution model's reliability and accuracy, reducing the bias stemming from often deterministic and static system variable assumptions [REF-23]. Several such approaches can be found in the pertinent literature either in the form of two-stage optimization-simulation [REF-25] [REF-26], [REF-27], or in simulation-based approaches [REF-29]

The simulation of a last-mile logistics system requires, apart from the service itself, the simulation of the underlying network and traffic dynamics, which directly affect the performance of the fleet, and eventually the logistics service itself. Modelling background traffic dynamics for last-mile logistics simulations relies usually on traditional traffic and network modelling techniques which may vary from macroscopic, mesoscopic, microscopic and hybrid network loading models coupled with traffic assignment methodologies. An extensive review of such approaches can be found in [REF-29] and more recently in a report springing from the H2020 SETA project, entitled "Exploring Prediction Perspectives".

Towards an efficient and robust last-mile logistics *planning service* for optimal analysis, management and assessment of sustainable and multimodal last-mile delivery systems, GREEN-LOG adopts a generic optimization-simulation methodology. More specifically, the optimization service described in Section 4 is coupled with and enhanced by a multimodal mesoscopic fleet-traffic simulation service, which relies on Aimsun Ride [REF-30], [REF-31], in an interoperable and seamless manner towards the delivery of a dual-purpose optimization-simulation framework for last-mile logistics. Either from the point of view of a public authority responsible for policy making, regulation and infrastructure investment or from the point of view of an actual Logistics Service Provider, the goal is to develop and deploy a joint optimization-simulation service for both i) what-if scenarios analysis (the long-term planning context) and day-to-day operations planning (the operational context).

In the sections below, the envisioned joint service's framework is described, emphasizing particularly on:

- ✓ The methodological approach and framework architecture
- ✓ The network models, simulation components and interface descriptions
- ✓ Initial results from the simulation service prototyping
- ✓ And the future work.

5.2 Methodological Approach

The GREEN-LOG methodology for the joint optimization-simulation service framework relies on two distinct and purpose-dependent approaches, i.e.:

- ✓ A one-off two-stage approach suitable for what-if scenarios and long-term planning from either the LSP or an authority's perspective, and
- ✓ An iterative two-stage approach computing equilibrated fleet service conditions for day-day service fleet management (i.e., fleet scheduling)

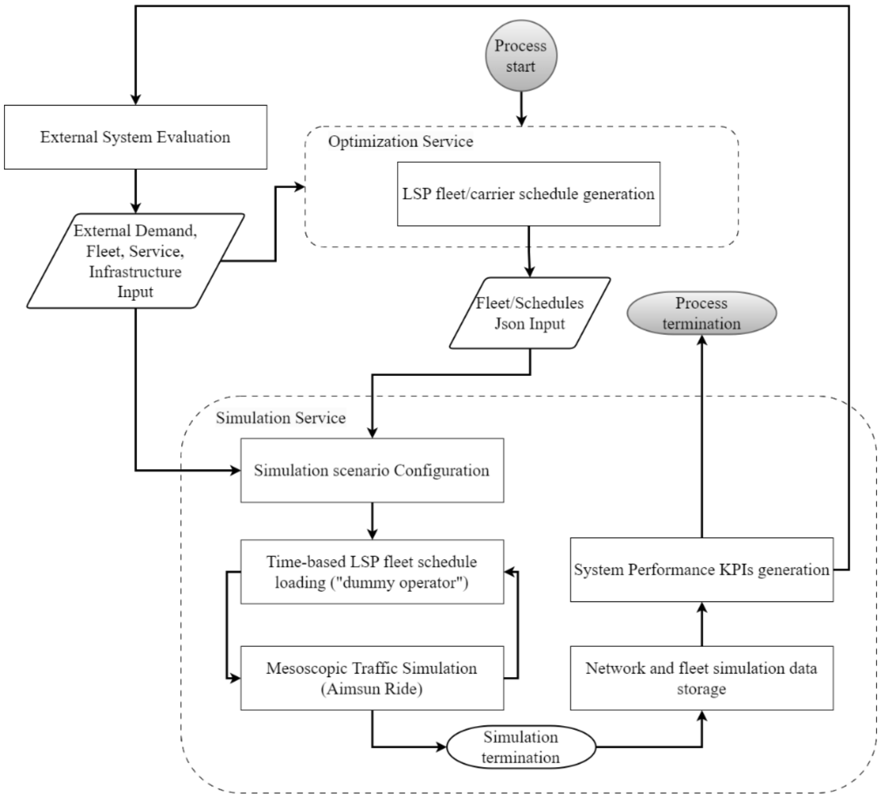


Figure 25 Flow Diagram of the two-stage one-off optimization-simulation methodology for last-mile logistics long-term planning applications (emphasis on simulation service)

Figure 24 illustrates the flow and sequence of the required optimization-simulation processes for the long-term last-mile logistics service planning applications. At the first stage, the optimization service co

mputes the optimal fleet schedules for satisfying a given external parcel demand, given different fleet, service and infrastructure information (e.g., fleet sizes, MCC locations, vehicle types and capacities, service constraints, service characteristics). The resulted schedules are the main input to the optimization service tasks that follow. First, the simulation scenario is configured given external input and the scheduling results produced by the optimization service. The simulation of the configured last-mile logistics service scenario entails the concurrent triggering of two disparate components/processes which rely on a time-based communication pattern.

The schedule loading process is handled by a "dummy operator", i.e., a component that is responsible for representing and managing an LSP operator's actions in the simulation environment. The "dummy" characterisation indicates that all operator's actions – in this case, fleet scheduling/routing – are pre-defined offline (not computation at simulation runtime) by the optimization service and are just executed in the proper sequence to serve the parcel demand at simulation runtime. A fleet schedule represents the daily sequential trips from point to point (parcel pick-up, parcel drop-off, depot and parking locations) that a vehicle is performing within a day (see Section 5.3 for more details). The fleet schedules are, thus, loaded into a mesoscopic traffic simulation environment, i.e., Aimsun Ride, at the departure time of a fleet vehicle's first trip of the day. The simulation process is responsible for:

- routing the fleet vehicles from point to point based on a shortest path function considering the road infrastructure available and the prevalent pre-trip traffic conditions.
- Capturing the interactions of the fleet vehicles with existing background traffic which is modelled using a mesoscopic network modelling methodology[7]

Once the simulation has terminated, fleet and network performance data are stored and a dedicated KPI generation process is triggered to produce performance indexes of interest to any end-user (e.g., travel times, vehicle kilometres travelled, emissions). The optimization-simulation process is terminated once the KPIs have been provided to users' external system evaluation processes capable of revising the system variables for the generation and simulation of more what-if scenarios.

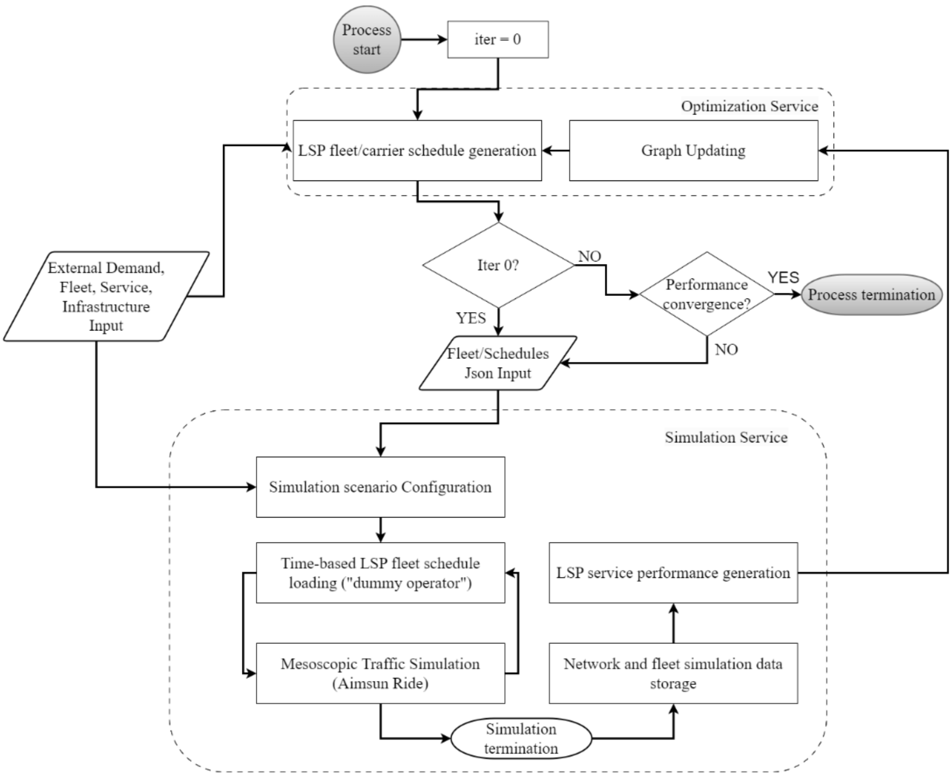


Figure 26 Flow Diagram of the hierarchical iterative optimization-simulation methodology for last-mile logistics day-to-day fleet scheduling operations (emphasis on simulation service)

[7] https://docs.aimsun.com/next/24.0.0/UsersManual/MesoDiscreteSimulation.html

An iterative, equilibrium-seeking approach – which still relies on the same services and components – is adopted for the deployment of the optimization-simulation framework in day-to-day fleet scheduling operations for LSPs, as illustrated in [REF-23]. In last-mile logistics, parcel demand's majority (pick-up and delivery requests) is known in advance, given that the demand-responsiveness of such services is not as high as in passenger and mobility systems. Same-day deliveries are in fact a good example of demand responsiveness in last-mile logistics systems, but good predictions of the spatiotemporal dimensions of same-day requests can be produced by models presented in Section 3. As such, optimal next day fleet operations require a methodology which will ensure the maximisation of the efficiency by which an LSP's existing fleet will be dispatched and scheduled to serve the required demand, in the most sustainable way, while considering representative uncertainty and dynamism in traffic conditions of urban environments. Given the additionally higher complexity of stochastic and dynamic variations of the optimization problems formulated in Section 4, optimization with static and deterministic network performance assumptions is coupled with simulation and run iteratively until convergence, i.e., the fleet performance remains relatively unchanged between different process iterations. Simulated fleet trips and their respected travel times are provided to the optimization service in each iteration and a graph updating process is responsible for updating the point-to-point network performance assumed by the optimization service. After each optimization service run, model system convergence is evaluated given a criterion and pre-defined thresholds. The analysis and final definition of the convergence criteria shall be presented in the final version of the D2.3 in M36.

5.3 Component / Service / Model specification

5.3.1 AIMSUN Ride

Aimsun Ride is a simulation platform for planning towards new – and primarily demand-responsive – **Mobility and Last-mile Logistics deployments** in urban and peri-urban environments. It is, specifically, an agent-based demand-supply interaction framework for multimodal and multi-operator fleet-based service systems, coupled (as a plug-in) with the state-of-the-art Aimsun Next's multi-class mesoscopic traffic simulator.

- ✓ Agent-based: both demand and supply within a service ecosystem are represented as (synthetic) agents (e.g., travellers, fleet vehicles, operators) with certain physical and behavioural characteristics capable of performing centrally controlled or autonomous actions; the agent-based design enables the integration of Aimsun Ride with both activity-based and 4-step demand models,

- ✓ Demand-supply interaction: towards a realistic replication of mobility and logistics ecosystems, both system's demand and supply agents communicate via flexible APIs and are designed with diverse and configurable behavioural (travellers, drivers) and operational (mobility operators) facets,

- ✓ Multimodality and multi-operator: the system enables the simulation of multimodal networks (mobility markets with different available services) where each service is managed by an operator, while different competing operators may exist for the same service type; any intermodal journey, including first-/last-mile to public transportation, can be simulated.

Aimsun Ride has been designed to enable end-users such as transport planners, public authorities, mobility/logistics operators and researchers to test, analyse, optimize and evaluate – either concurrently (multi-service, multi-operator) or in isolation (single service, single operator) – different mobility and logistics service/system designs. Table 7 below gives an overview of the system/service types that Aimsun Ride has the capacity to replicate, while Table 8 refers to the possible system/service design elements which Ride enables testing via proper scenario and input file configuration:

Table 7 Possible Aimsun Ride Mobility and Logistics services/systems use-cases

Demand-responsive ride services	Demand-responsive shared vehicle services and micro-mobility	Mobility-as-a-Service and Multimodality	Freight
E-hailing / Ride-hailing	Car-sharing	Multimodal Integrators and Journey Recommendation Systems	Micro-freight services (vans)
Ride-sharing / Ride-pooling	Bike-sharing	Feeder Transit Services	Micro-consolidation hubs
Car-pooling	Scooter-sharing		Cargo bikes
Combination of the above (Uber, Lyft, etc.)			Cargo scooters/motorbikes

Table 8 Possible Aimsun Ride scenario design features

Demand and Behaviour	Service Design	Service Infrastructure	Fleet Management	Traffic resolution
Service user menu (mode/route) choice model	Round-trip vs One-way	Vehicle Types and Capacities	Fleet sizing and composition	Mesoscopic network loading with simulated costs
Configurable synthetic agent socio-demographic and socioeconomic attributes	Fixed route vs flexible routes	Vehicle Fuel Types	Fleet assignment and scheduling	Mesoscopic network loading with free flow conditions
Driver behavioural facets (trip acceptance, etc.)	Station-based vs Free-floating	Autonomous vs human-driven	Vehicle Relocations	External (historical) travel times

	Reservation-based vs on-demand booking	Infrastructure (Hub, Charging stations) positioning and design	Pricing	
	Fixed timetable vs demand-responsive		Personalized recommendations and journey planning	

Aimsun Ride can be a valuable tool for any end-user aiming to either identify optimal settings for designing and operating a mobility or last-mile logistics service or consider the impact of one or more new mobility and last-mile logistics services on urban and peri-urban planning decisions for a city or a metropolitan area. In either case, demand, service, and supply models are required within a unified modelling framework to conduct a thorough service operations design or impact study. Figure 27 below illustrates an Aimsun Ride-based (simulation-based) architecture of such a generic unified modelling framework, focusing primarily on highlighting the main I/O (Input/Output) requirements between Aimsun Ride and other required external components.

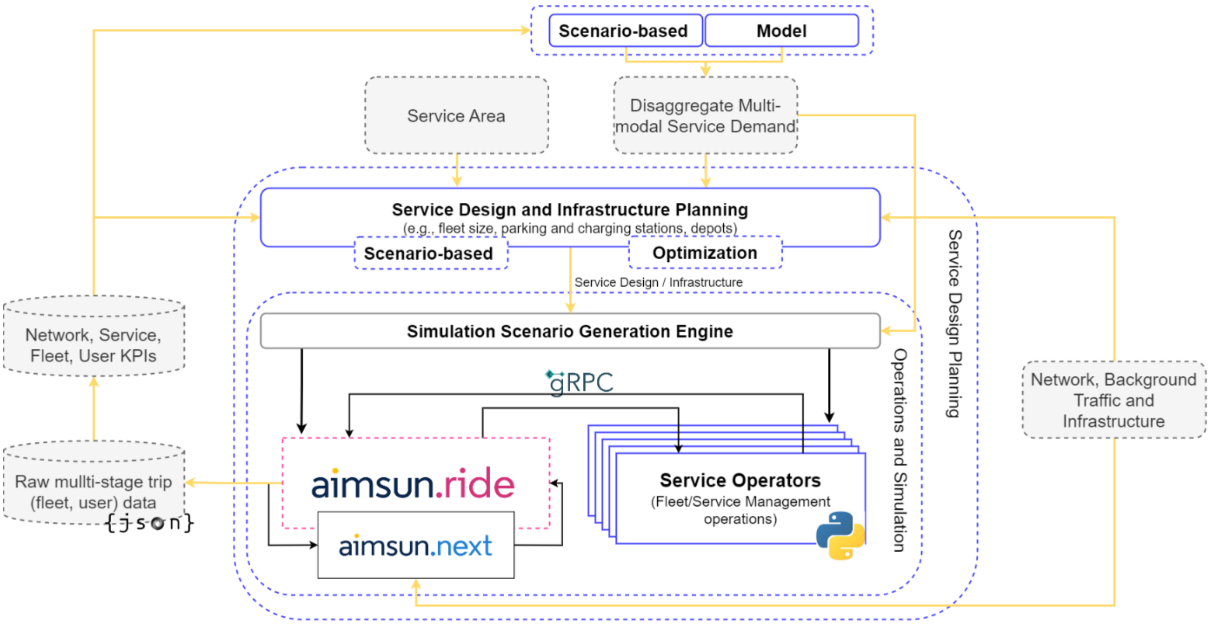


Figure 27 I/O Communication between Aimsun Ride and external components within a generic simulation-based unified service modelling framework

5.3.2 Aimsun Ride Inputs

The main input required by Aimsun Ride is related to i) demand, ii) service and iii) network information. From the demand perspective, Aimsun Ride receives as input a request .JSON file containing the entirety of the disaggregate (agent-based) demand for any mobility or logistics service to be simulated for a given scenario along with agents' socio-demographic and socio-economic variables. The request file contains practically fields related to trip information (origin point, destination point, departure time, etc.). as well as several optional user-defined fields (accessibility constraints for specific individuals, walking time constraints, etc.). The demand and population data may be derived either from end-user defined scenarios or from actual demand (e.g., mode choice, subscription choice) and synthetic population models.

From a service perspective, Aimsun Ride also receives as input an operator .JSON file containing service and fleet design information such as the service type, the fleet size along with the type of vehicles and their fuel type, vehicle capacity, original location and, finally, infrastructure-specific data (charging parking stations and locations, capacities, etc.). Service and infrastructure design elements are strategic and tactical decisions which may be either produced by sophisticated external (offline) optimization processes or via scenario and sensitivity analysis-based processes performed by an end-user.

Finally, from a network perspective, Aimsun Ride requires supply information such as the service area of operations, non-service-related infrastructure (such as other parking locations or public transport stops) and the underlying traffic dynamics (background traffic conditions generated from conventional transport modes). For the latter and as already mentioned on Table 5-2, Aimsun Ride requires an .ANG file (Aimsun Next network format) which encapsulates and defines the background traffic resolution dynamics for each simulation scenario, i.e., mesoscopically simulated traffic dynamics, free-flow conditions or externally-loaded historical travel times.

Towards the harmonization of all the required input data into a Ride-compatible format, a Simulation Scenario Generation Engine (offline process) is responsible for converting formats from different sources into the Aimsun Ride-compatible formats for all the three (3) aforementioned input categories.

5.3.3 Aimsun Ride – Service Planner/Operator Integration Architecture

An important design consideration in Aimsun Ride's architecture lies on its ability to communicate interoperable and flexibly with any type of service operator (direct communication) or/and service design planner (indirect communication) as part of a simulation-based system/service analysis process. The frequency, type and flows of such communication – and its underlying required architecture – depend on:

- ✓ The end users' assumptions on the impact scale of a region's overall and prevailing network traffic conditions on the efficiency of both the service design and operations.
- ✓ The dynamism of service operations and the dependency with dynamic stochastic traffic conditions; operations like fleet scheduling might be impacted on an event or a time basis given different levels of variability in travel time distributions over time (congested vs non-congested network, peak vs off-peak periods)
- ✓ The demand-responsiveness level of the demand; some mobility services operating with a pre-booking system and most last-mile logistics services know the demand in advance (e.g., next-day delivery) and can, thus, plan fleet schedules and flows offline. In highly demand-responsive cases though, like e-hailing and bike-sharing, fleet operation approaches are based on frequent re-evaluation based on the prevailing system status.

Due to such communication requirements, Aimsun Ride offers a TCP/IP interface, i.e., a gRPC-based (google Remote Procedure Call framework) API, which enables flexible server-client implementations with external Python modules that define and establish the communication types and patterns between service operators and Aimsun Ride. Figure 28below illustrates the two (2) traditional types of Ride-Operator communication, i.e., event-based and time-based communication. For the sake of simplicity, a fleet scheduling operation for a e-hailing taxi operator is selected to demonstrate the difference between the 2 communication approaches.

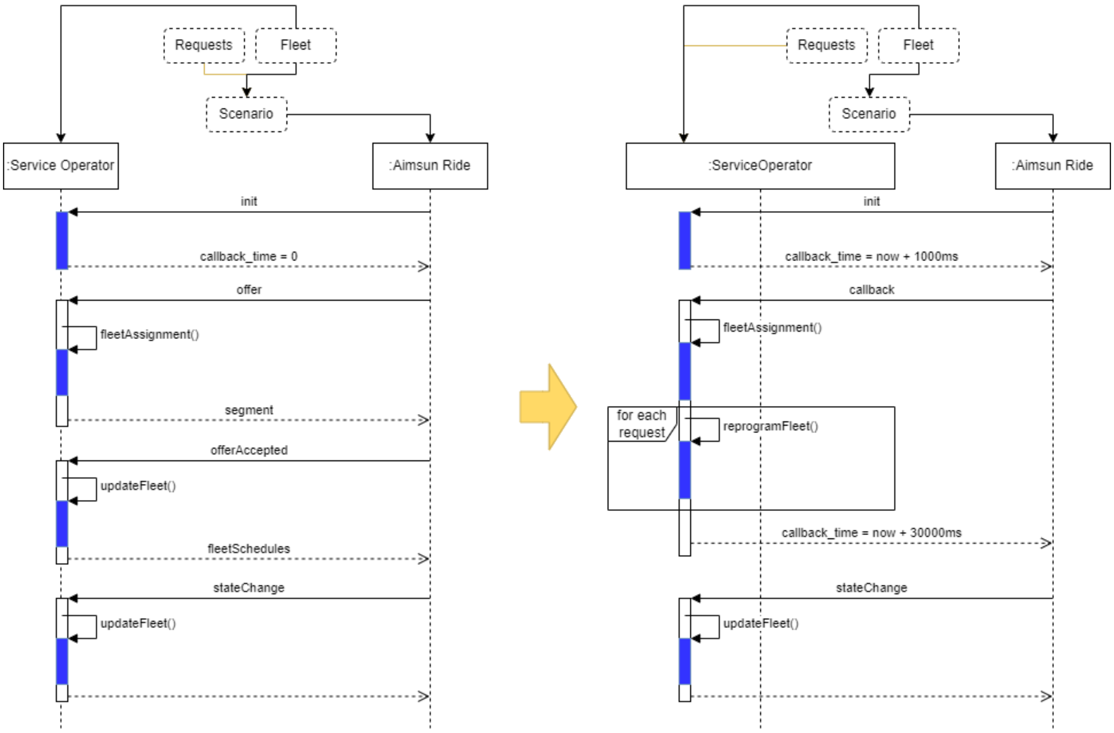


Figure 28 Service Operator - Aimsun Ride event-based and time-based sequence diagrams

The event-based communication design is illustrated on the left side of Figure 28 and the time-based on the right. On the event-based case, all requests (demand) and service fleet information are passed through a scenario file to Aimsun Ride. Given that Aimsun Ride is fully aware of the service demand, whenever a user is expected to make a trip request from A to B (event), Ride requests (at trip departure time) an "offer" from the service operator. If the offer is accepted[8], then the operator sends the final schedule to the vehicle in Aimsun Ride. Whenever there is an update in the status of the fleet vehicle (event), Aimsun Ride again calls a "stage change" function informing the operator of the change and asking for any subsequent action.

On the time-based communication, all requests are passed to the service operator instead of Aimsun Ride. The reason is that Aimsun Ride is called by the service operator at fixed time intervals (in this instance 30 sec) and doesn't need, thus, to be

[8] In a single service and single operator environment only- one offer would have been sent to Aimsun Ride and it would thus be accepted. However, in a multi-service or multi-operator environment Aimsun Ride would ask for an offer from each operator and would then decide via a logit model which is the optimal one. This auction-like approach is configurable.

aware of when requests for a service take place. Every 30 sec a callback is sent from Aimsun Ride to the Service Operator asking for an action plan. The Service Operator batches the trip requests made in the last 30 sec, plans the vehicle assignment for all and communicates the final schedule for each vehicle-request pair to Aimsun Ride. As shown, a concurrent event-based and time-based communication is possible for different types of operations; an event-based "state change" message is sent from Aimsun Ride to the Service Operator, while the fleet assignment remains a time-based process.

5.3.4 Aimsun Ride Outputs

After an execution of a simulation scenario, Aimsun Ride produces mainly three types of outputs:

1. An event log file (binary file that can be converted to a JSON format) containing a complete list of all the events that transpired during the simulation of each service and are related specifically to travel states (user, vehicle), agent positions, vehicle capacities and the simulated paths for each trip stage.

2. An optional recording file to visualize the simulation results within Aimsun Next's Graphical User Interface (GUI).

3. Statistical measures about network performance (network-wide, section, intersection, lane, vehicle) as traditionally produced by Aimsun Next.

Such raw output data (referring to 1) can be post-processed to extract Key Performance Indicators (KPIs) on three different categories: i) service efficiency, ii) user satisfaction, iii) fleet performance, which can iteratively be fed to offline service design planning processes for design dimensions re-evaluations. An example of such KPIs is illustrated in the Table 9 below.

Table 9 Indicative service-related KPIs extracted from Aimsun Ride outputs

	Service Efficiency	User satisfaction	Fleet Performance
Average distances/travel times per vehicle			x
Total average distances/travel times			x
Average waiting times per traveller		x	
Average VKT (vehicle kilometres travelled) and/or VHT (vehicle hours travelled)			x

Number of satisfied and unsatisfied requests	x	x	
Average vehicle utilization rate per time interval (trips per vehicle)	x		x
Average walking times per traveller	x		
Average vehicle occupancy per trip	x		x
Average number of vehicles queueing for charging	x		

5.3.5 GREEN-LOG Last-mile Logistics Simulation Framework Architecture

The integration of Aimsun Ride within GREEN-LOG's joint optimization-simulation service framework and its deployment based on the methodology described in Section 5.2, has necessitated ad-hoc adjustments on both the Ride-based generic service simulation and integration frameworks described in Figure 28 and Figure 29 respectively. Generally, the main adjustments and approaches which have been adopted, implemented and tested are:

1. A new "standardized" input file (JSON) format for communicating externally and offline computed fleet schedules (i.e., not at simulation runtime)
    a. The standardization terms implies that this format is generally deployed in all Ride use-cases[9] where within-day service fleet operations are computed offline from an external component and are input to the operator-Ride system.

2. The development of a new "dummy" operator ("fleet schedule loader") based on a variation of the time-based communication approach described in Figure 29 (right-side).

3. A new "standardized" output file (JSON) format which is compatible with the design of the fleet schedule input file format mentioned above, and contains the simulated travel times for each fleet vehicle trip (i.e., schedule item)

[9] including Ride applications in other Horizon Europe projects, such as CONDUCTOR, SYNCHROMODE, etc.

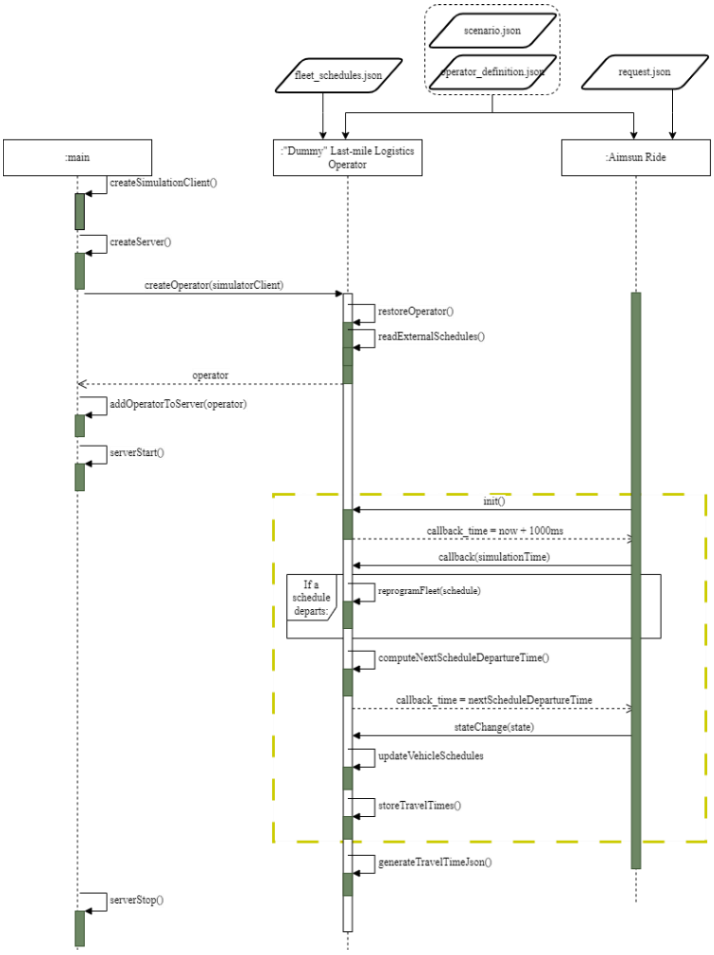


Figure 29 The GREEN-LOG Last-mile Logistics Simulation Framework Architecture

More specifically, the GREEN-LOG last-mile logistics simulation framework architecture and the adopted integration approach is illustrated in detail by the sequence diagram of Figure 29. A "main" script, acting as the operator-simulation system orchestrator is responsible for establishing the creation of the server along with the main objects and the ports in which each object is communicating with. This process is responsible for conducting the following sequential steps: i) creating the simulation client, ii) creating the server, iii) creating the "dummy" operator, iv) adding the operator to the server, v) starting the server, and vi) terminating the server. The two main components instantiated during in this process is the "Dummy" Last-mile Logistics Operator object and the simulator object (which establishes the connection to a port in which Aimsun Ride is sending and receiving messages to and from respectively). The instantiation of the "dummy" operator includes reading and storing in memory the externally generated (from the optimization service of Section 4) fleet schedules (JSON format). Table 10below describes the main attributes of the data contained by the fleet schedule file, while Figure 30 illustrates an example of it.

Table 10 Fleet Schedules JSON File Attributes

Attribute Name	Attribute Description	Attribute Value Type	Comments

name	Name of the input file	string	-
vehicles	The fleet vehicles to be simulated	list	All fleet vehicles, regardless of whether they have schedules
vehId	The id of the fleet vehicle	string	Same fleet vehicle ids as the ones included in the operator_definition.json file; ids should be unique, i.e., UUIDs
schedule	The schedule of a vehicle for the entire computation period; it consists of ordered schedule items that indicate the type of operation sequences that will be simulated	dict	-
scheduleStartTime	The departure time of the vehicle from the initial location (e.g., station, depot, hub) in the simulation environment	miliseconds	Time since epoch
scheduleItems	An item of the schedule incapsulating information for the operation	list	-
itemId	The identifier of the schedule item	string	-
requestId	The identifier of the operation request (passenger or parcel)	string	Optional; it can be used to map trip-specific simulation outputs with the associated passenger or parcel operation request
itemOperationType	The type of the operation the vehicle will perform in this item; Values are: “PickUp”, “DropOff”, “Collection”, “Delivery”, “Rebalancing”,	string	-

	“GoingToCharge, “GoingToPark”		
itemOperationDuration	The time the operation will last (e.g., how much time to charge, how much time to park and remain idle)	milliseconds	Optional
itemOperationDestination	The destination location for the specific operation in a schedule item	dict	-
x	Location longitude	float	-
y	Location latitude	float	-

```json
{
    "name": "Ride Fleet Schedule Input for Oxford, England",
    "vehicles": [
        {
            "vehId": "1523dfec-697c-4bfc-80fc-8290d4f64342",
            "schedule": {
                "scheduleStartTime": 30600000,
                "scheduleItems": [
                    {
                        "itemId": "2b357496-c23c-4796-b079-3daf91cac863",
                        "requestId": "2b357496-c23c-4796-b079-3daf91cac863",
                        "itemOperationType": "Relocation",
                        "itemOperationDuration": 5700000,
                        "itemOperationDestination": {
                            "x": 453749.489446069,
                            "y": 205644.842760308
                        }
                    },
                    {
                        "itemId": "2a9f5650-059e-4f73-9d4b-9b441fb73211",
                        "requestId": "2a9f5650-059e-4f73-9d4b-9b441fb73211",
                        "itemOperationType": "Relocation",
                        "itemOperationDuration": 600000,
                        "itemOperationDestination": {
                            "x": 452021.171557753,
                            "y": 203640.086649804
                        }
                    },
                    {
                        "itemId": "9e38d419-f63c-47a7-99b5-bd3b5b285e1a",
                        "requestId": "9e38d419-f63c-47a7-99b5-bd3b5b285e1a",
                        "itemOperationType": "Relocation",
                        "itemOperationDuration": 3900000,
                        "itemOperationDestination": {
                            "x": 453749.489446069,
                            "y": 205644.842760308
                        }
                    },
                    {
                        "itemId": "9761a61c-71b9-4df6-abe6-65edfefa520a",
                        "requestId": "9761a61c-71b9-4df6-abe6-65edfefa520a",
                        "itemOperationType": "Relocation",
                        "itemOperationDuration": 4200000,
                        "itemOperationDestination": {
                            "x": 453770.865675632,
                            "y": 207825.297489134
                        }
                    },
```


Figure 30 Example of a fleet schedules JSON file generated by the optimization service

The main process of the operator is illustrated within the yellow-dashed box in Figure 30. Aimsun Ride is sending an init() message to the operator to ensure the initialization of the server-client communication with the operator. In the init() function a callback is requested immediately to be called by Aimsun Ride after 1 sec. Once the first call-back is made the operator is evaluating if any schedule is about to depart at this timeframe. If yes, then the simulator's reprogramFleet() function is called which sends the vehicle schedule to Aimsun Ride for simulation. Once all vehicle schedules have been checked and, if necessary, loaded, the operator is computing the departure time of the next vehicle and requests a callback from Aimsun Ride at this time. Once a schedule item (e.g., a collection trip or a delivery trip), a stage message is called by the simulator, and the operator is i) updating the fleet schedules store in the operator's memory, and ii) logging the travel time that was experienced for the schedule item (trip). Upon simulation and server termination, the operator is generating, among others (see Section 5.3.1) the JSON output file containing the simulated schedule travel times for each schedule item. The data attributes and a representative example are described and illustrated respectively in Table 11.

Table 11 Schedule Travel Times JSON File Attributes

Attribute Name	Attribute Description	Attribute Value Type	Comments
scheduleItemId	The identifier of the schedule item	string	-
startTime	Start time of the trip	int	UTC time since epoch
endTime	End time of the trip	int	UTC time since epoch
tt	Total experienced travel time of the schedule item	int	seconds

```json
{
    "1523dfec-697c-4bfc-80fc-8290d4f64342": [
        {
            "scheduleItemId": "2b357496-c23c-4796-b079-3daf91cac863",
            "startTime": 30600.0,
            "endTime": 31115.0,
            "tt": 515.0
        },
        {
            "scheduleItemId": "2a9f5650-059e-4f73-9d4b-9b441fb73211",
            "startTime": 36815.0,
            "endTime": 37335.0,
            "tt": 520.0
        }
    ],
    "7496dead-b097-46b8-9c42-8a56ee4d7f4e": [
        {
            "scheduleItemId": "93e83067-78fe-4514-b9d7-cd6e6e54638e",
            "startTime": 32400.0,
            "endTime": 32910.0,
            "tt": 510.0
        },
        {
            "scheduleItemId": "620a2b18-a50d-41e0-8259-89614ae8f691",
            "startTime": 32915.0,
            "endTime": 33135.0,
            "tt": 220.0
        },
        {
            "scheduleItemId": "28f5fe3d-9119-403d-8193-ed5e864c3e4e",
            "startTime": 33975.0,
            "endTime": 34480.0,
            "tt": 505.0
        },
        {
            "scheduleItemId": "e15b50fd-a5b9-42a5-b548-b817179ec14f",
            "startTime": 34660.0,
            "endTime": 34880.0,
            "tt": 220.0
        },
        {
            "scheduleItemId": "8e50c76d-c589-4a38-998e-9c741cde67f1",
            "startTime": 35060.0,
            "endTime": 35265.0,
            "tt": 205.0
        },
```


Figure 31 Example of the schedule travel time JSON file generated by the simulation service

5.3.6 Network Models

The main goal of the simulation component is to replicate the stochastic and dynamic interactions between the fleet vehicles and existing background traffic, compute service fleet vehicle routes and produce realistic fleet operation performances. As such, a network graph development and modelling exercise is also required to produce calibrated network models replicating habitual (weekday, weekends) underlying traffic conditions for the area under investigation. A network model development process usually consists of a series of sequential and iterative processes to ensure a reliable model development (Figure 31). A more detailed analysis of the workflow process and the general data requirements have been presented in [D7.1: Land Network Models]() for the HARMONY H2020 project.

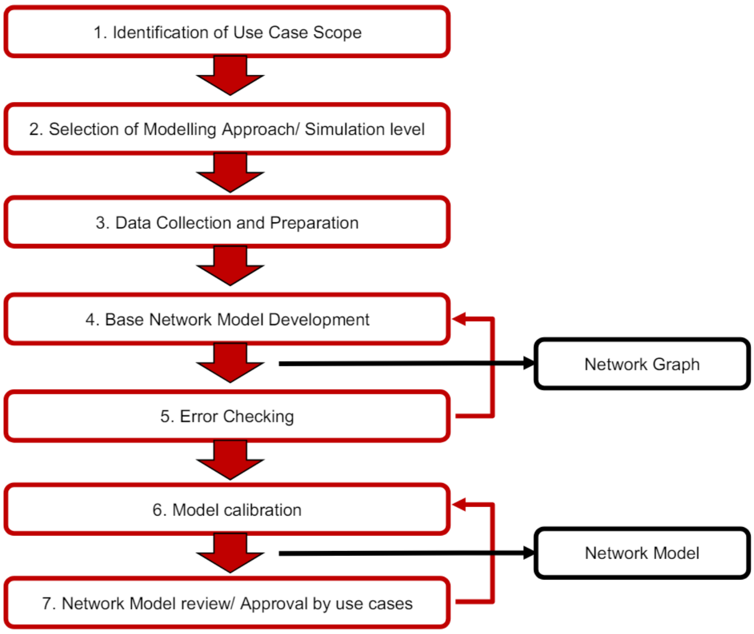


Figure 32 Workflow Process for Network Graph and Network Model Development

In GREEN-LOG the simulation service is planned to be deployed in the following Living Labs and for the following cases:

- ✓ Oxford, UK;
  - Planning case: simulation service deployment for what-if scenario analysis and assessment of different multi-purpose fleets (cargo-bikes) and MCC locations
  - Operational case: simulation service deployment for day-to-day service fleet management for PEDAL (local Logistics Service Provider)

- ✓ Barcelona, ES;
  - Planning case: simulation service deployment for what-if scenario analysis and assessment of multimodal transit-based last-mile logistics for the VANaPED Logistics Service Provider, considering different demand levels and sustainable multi-class service fleet vehicle types

- ✓ Athens, GR;
  - Planning case: simulation service deployment for what-if scenario analysis and assessment of white-labelling movable depot and MCC locations

5.3.6.1 Oxford Model

The urban ring road region of Oxford city has been selected as the area of interest for modelling given that it’s the main operations area for PEDAL, including the main local transport system infrastructure.

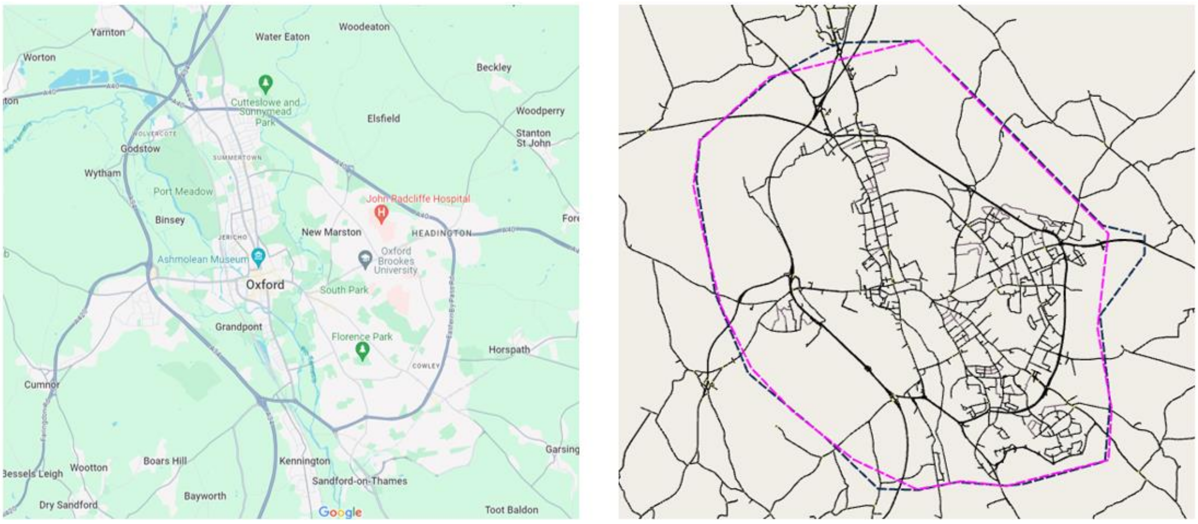


Figure 33 Oxford Study Area

The starting point for this model was an existing Aimsun Next model (Figure 32) which covers the study area and has an existing demand and traffic signals (see D7.1: Land Network Models from the HARMONY H2020 project). Due to the model's application in an operational setting, several model extensions have been applied using supply, control and demand data from 2023, to ensure high model quality and sophistication.

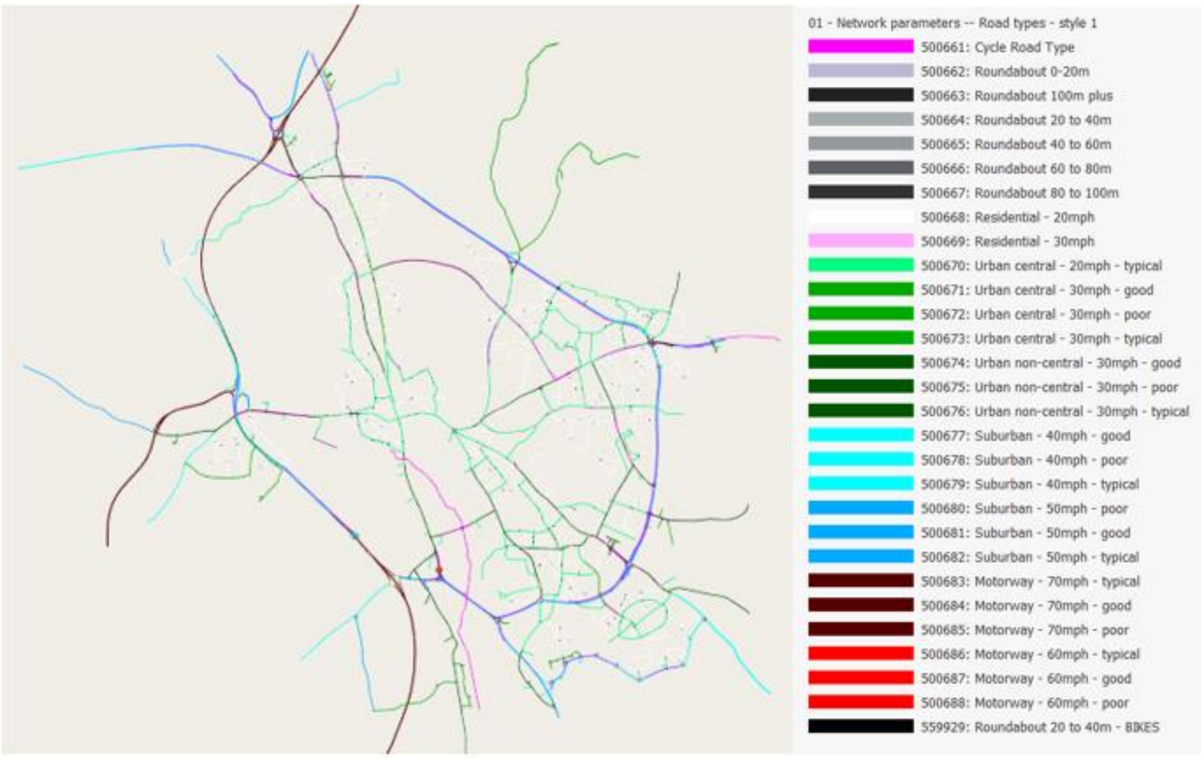


Figure 34 Existing Network Graph from HARMONY H2020 project

Supply Extensions

An extensive geometry refinement exercise has been deployed focusing mainly on:

- ✓ Geometry refinements (Figure 33)
    - Changes in network layouts, and
    - Network coding enhancements

- ✓ Cycling Infrastructure Representations (for cargo-bike simulations)
    - OSM (Open Street Map) geometry, which mentioned cycling facilities, was imported as a background to guide the manual coding of cycling infrastructure, which was not previously present in the model (Figure 34-Figure 36).

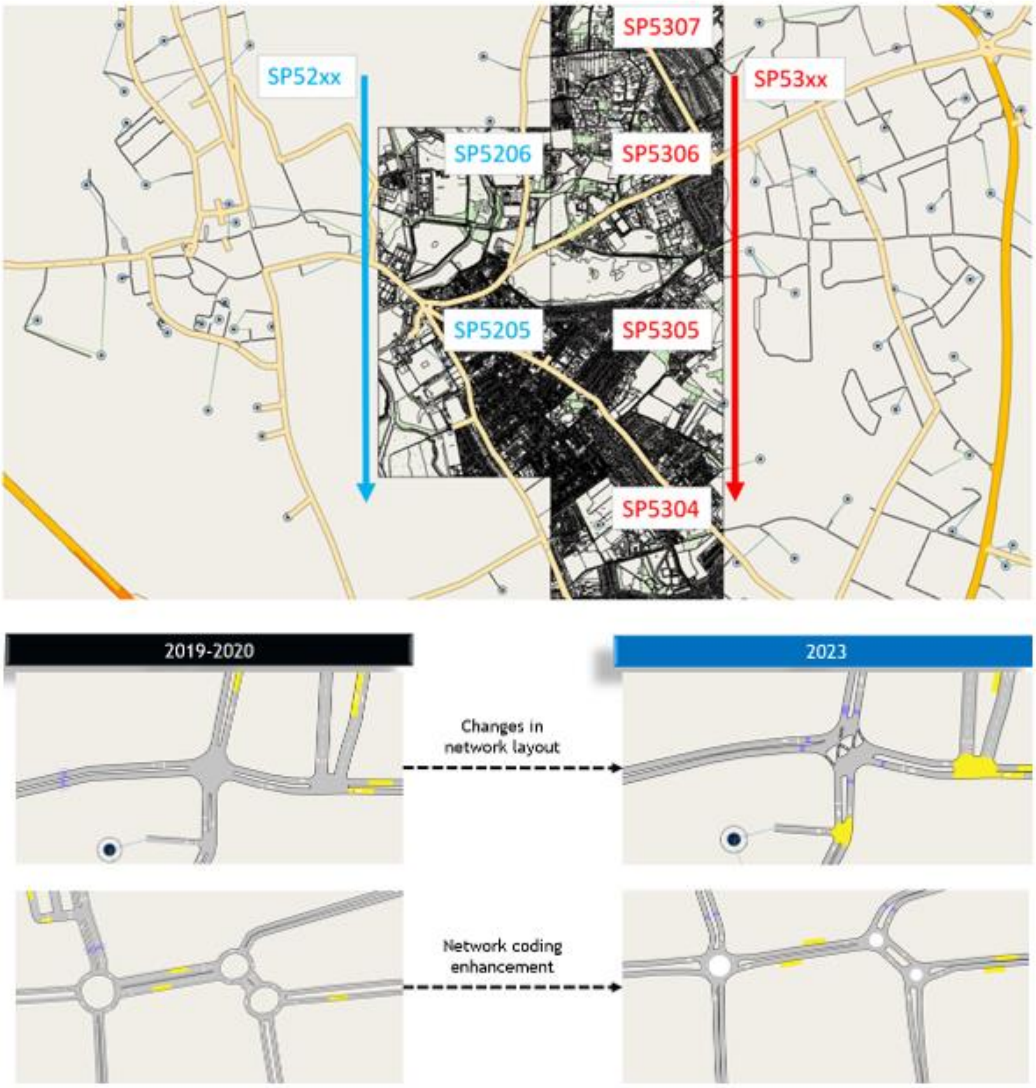


Figure 35 Network Graph Geometry Refinements

Figure 36 Import of OpenStreetMap cycling infrastructure (red)

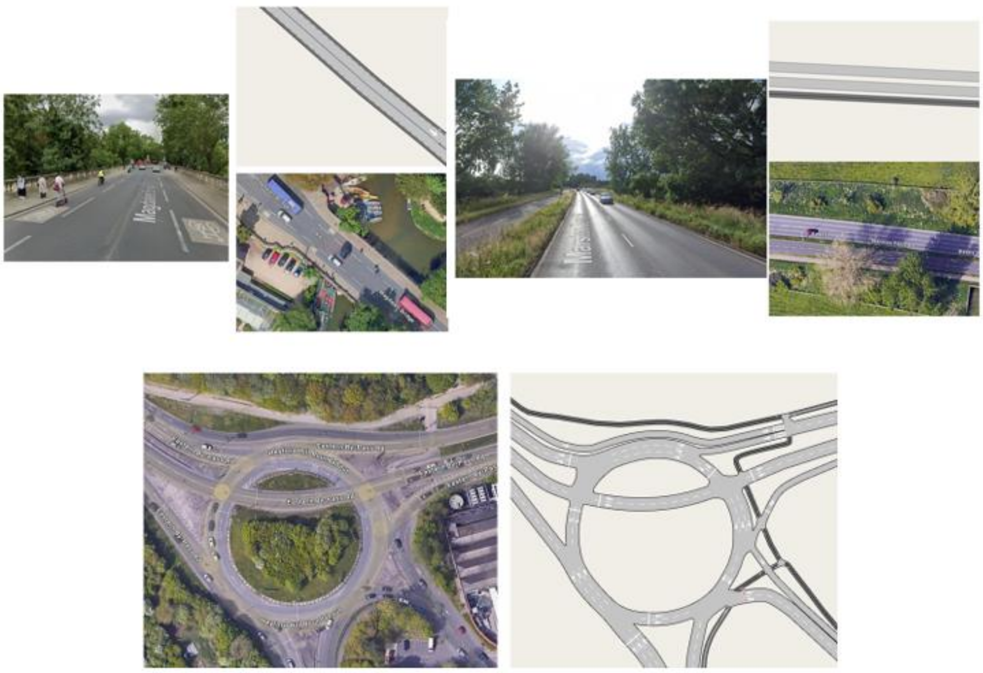


Figure 37 I) Cycle lanes right and left from road; II) Bidirectional cycle track left of road; III) Junction Crossings

The resulted network graph has the following attributes:

- Area: 92.53 km2
- Perimeter: 36.82 km
- Total Section Length (in km): 607
- Total Lane Length: 771
- Sections: 3897
- Nodes: 1643
- Centroids: 252
- Total Cycle Lane Length: 134.25 km

5.3.7 Demand Extensions

The provision of a background traffic demand (considering Cars and Heavy Goods Vehicle types) relied on a demand re-adjustment process based on real data detection points (Figure 37) for the study area for Morning, AM, IP, PM and Night

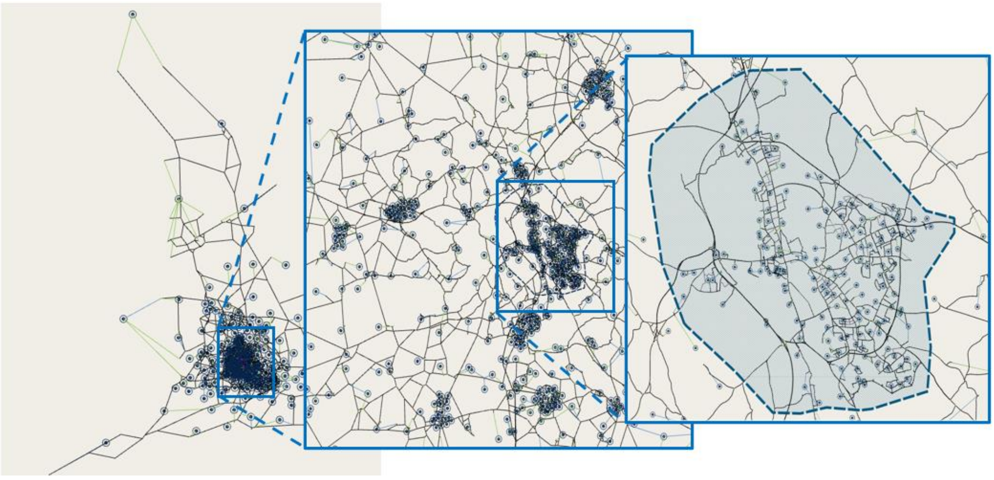


Figure 38 Oxford Model Centroids

periods (weekdays and weekends).

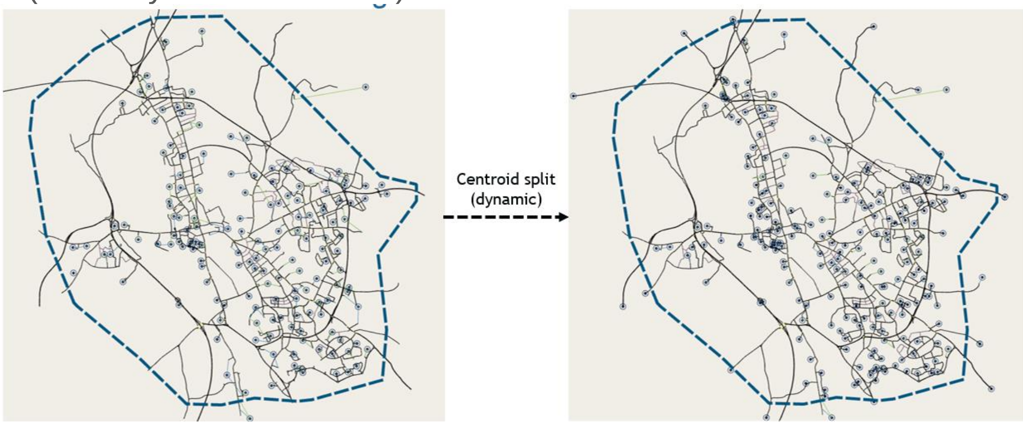


Figure 39 Model Cordoning

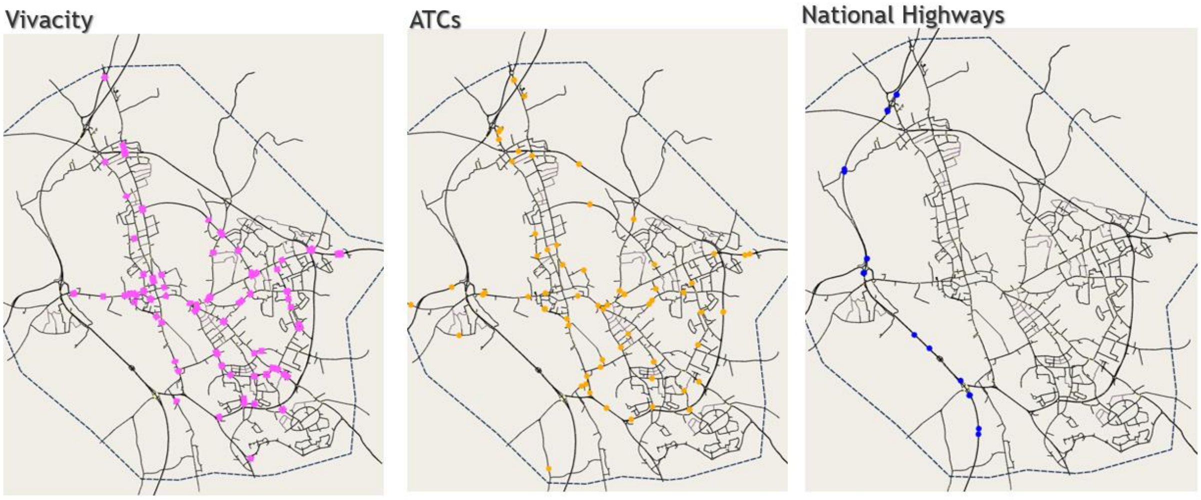


Figure 40 Detection data used for demand calibration and re-adjustment

An example of the resulted profiled demand suitable for simulations is illustrated in the figures below for the AM and PM peak hours.

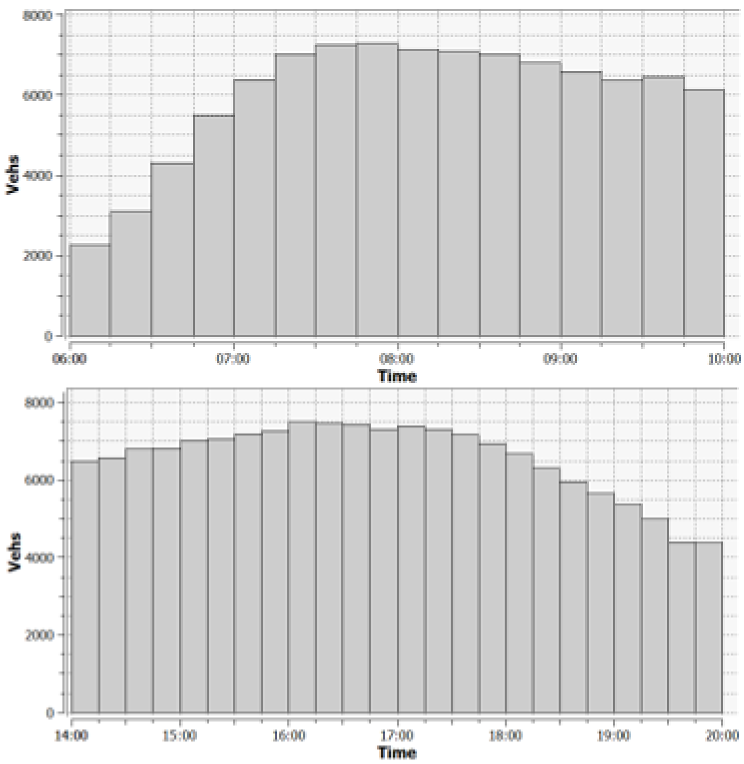


Figure 41 Example profile of traffic demand for AM and PM demands

5.3.7.1 Barcelona Model

The Barcelona City has been selected as the area of interest for modelling in the Barcelona LL, given that it's the main operation area for VANaPED, and it's the priority area for the LSP's cargo-bike fleet expansion. The developed model is intended to be used for planning purposes, including multimodal transit-based deliveries to areas not reachable easily by cargo-bikes (due to high altitude), and optimal co-ordination of multi-class vehicle fleet. For the model development, given open data availability and 24-hour OD and skim matrices from ATM, the following steps have been implemented.

5.3.8 Strategic Demand

Centroids were provided by ATM from their strategic model, along with a 24-hour vehicle demand. The private vehicle matrix was then loaded into the model.

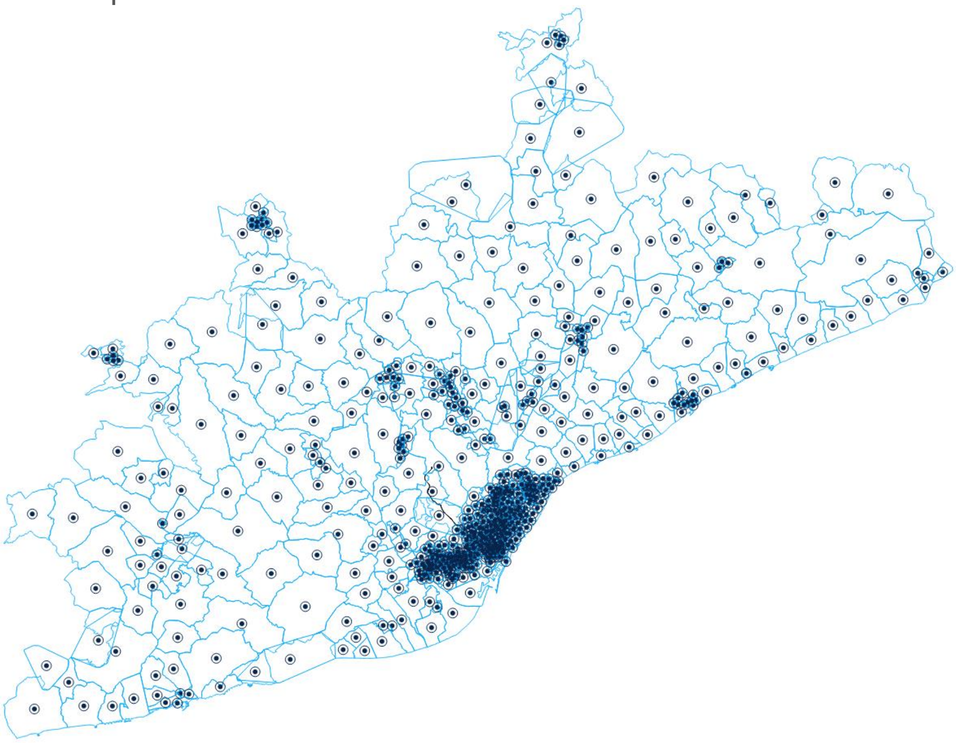


Figure 42 Barcelona Metropolis Strategic Model Centroids

Strategic Supply

For a polygon enclosing the strategic model's centroids, OSM highway data has been downloaded for the area then imported into the modelling software.

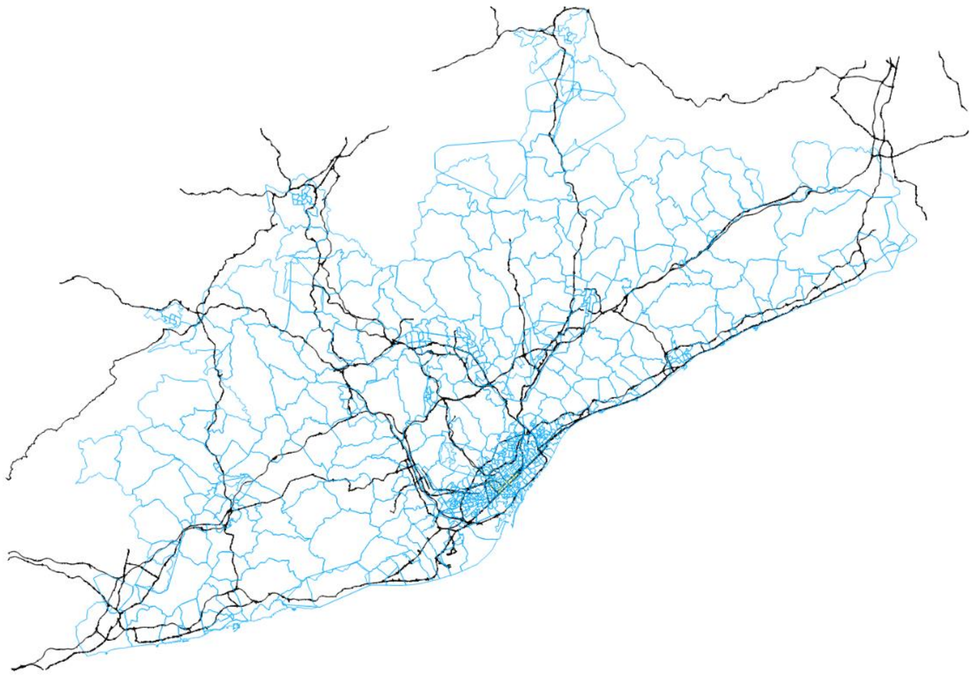


Figure 43 Barcelona Metropolis Strategic Model's Highways

5.3.9 Strategic Assignment

The centroids have been connected into the model and vehicles assigned on the network using a static assignment process.

Figure 44 Barcelona Metropolis Strategic Highway Model's Assignment Results

Subnetwork Definition

Administrative boundaries were then downloaded from OSM, to ensure that all locations within the selected St Gervasi region as the target neighbourhood were included. Then based on visual inspection, a subnetwork was defined including all reasonable network infrastructure which would normally be used to access locations between the Estacion de França (VANaPED depot) and St Gervasi (service extension area), including the cycling infrastructure.

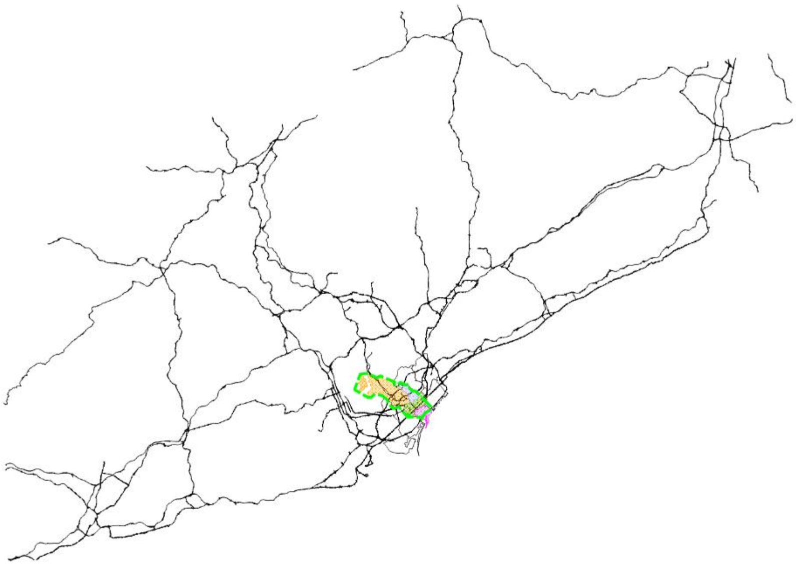


Figure 45 Location of the Barcelona subnetwork in context of wider AMB model area.

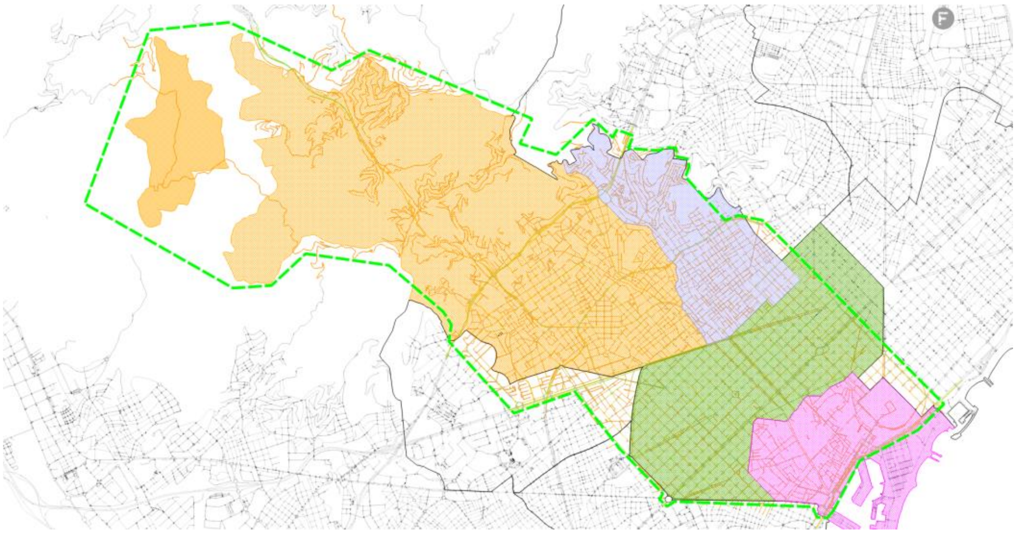


Figure 46 Barcelona subnetwork in context of major administrative areas (St Gervasi, Gracia, Eixample and Ciutat Vella)

Given the potential sensitivity of slope, altitudes were imported from a DEM (Digital Elevation Model) source for the subnetwork area.

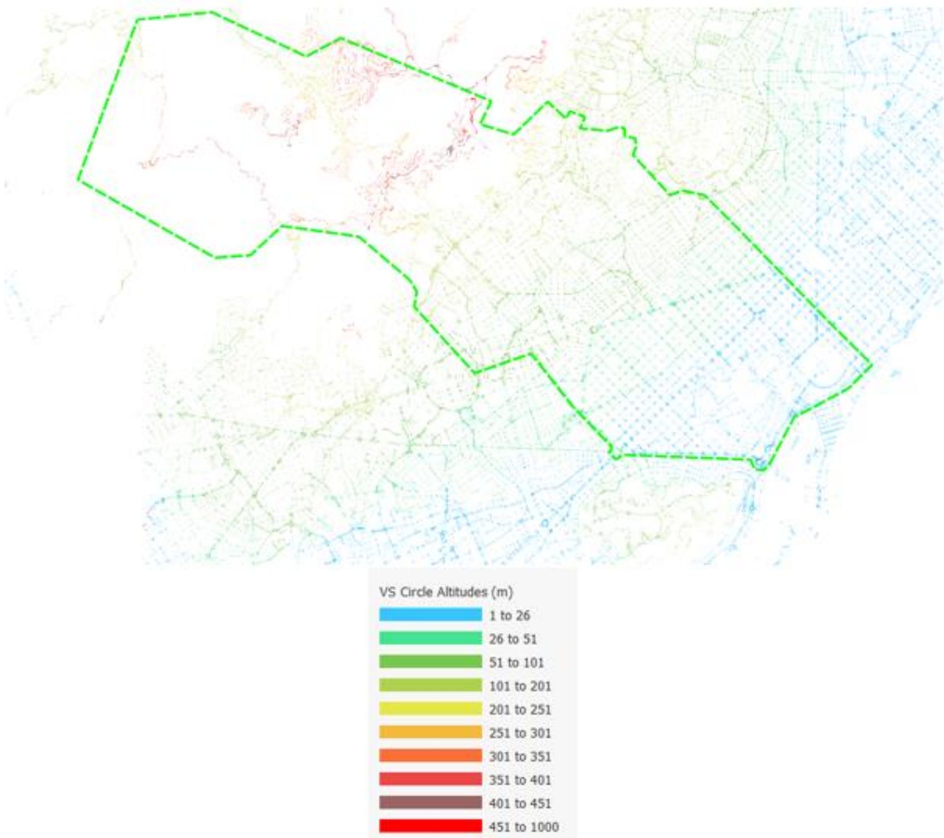


Figure 47 Elevation information in the network, blue indicating lower elevations, red indicating higher elevations.

5.3.9.1 Subnetwork Supply

Additionally, the original XML tags associated with the nodes, ways and relations were imported into the model from the OSM file, and then used to improve the quality of the network inside the subnetwork, adding details such as way and lane reservations, yield indicators, ways where reverse cycle passage is permitted, pedestrian crossing locations and simplify some signalized intersections.

5.3.9.2 Subnetwork Summary

The resulted subnetwork graph has the following attributes:

- Total Section Length (in km): 894,
- Total Lane Length: 1206
- Sections: 28636
- Nodes: 14074
- Centroids: 317 (in 1 Centroid Configuration)

5.3.9.3 Subnetwork Supply Transit

To support scenarios where FGC metro services are used for parcel deliveries, the FGC (Ferrocarrils de la Generalitat de Catalunya) network was manually coded into the model, and then services (stops, routes and timetables) imported from a GTFS data source (https://www.fgc.cat/en/opendata/).

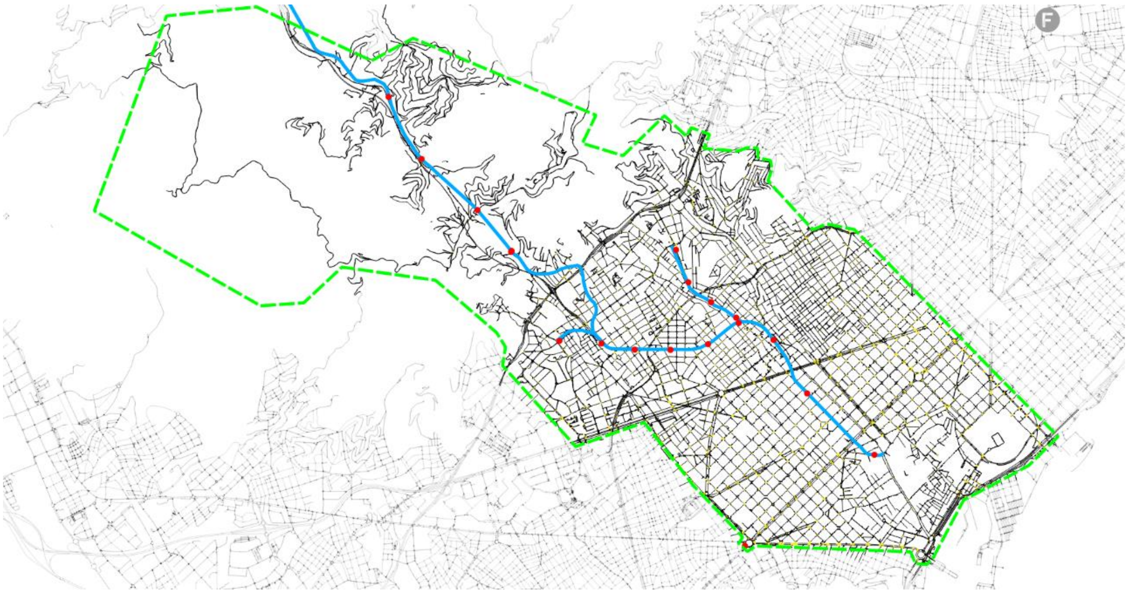


Figure 48 FGC network and Stops

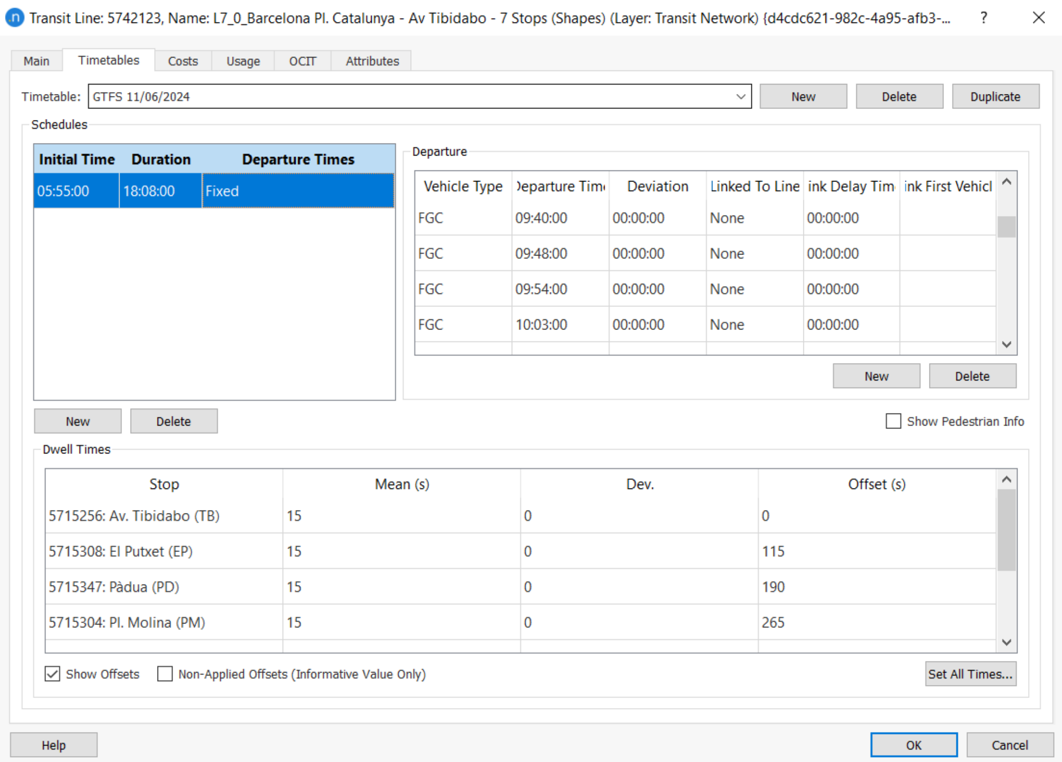


Figure 49 Timetable information loaded for a FGC line (typical weekday)

5.3.9.4 Subnetwork Demand

To provide conditions of background demand for the simulation, a traversal operation was applied (using the strategic assignment result) to provide a flat subnetwork demand of private vehicles. This demand was then profiled using values from an open report[10].

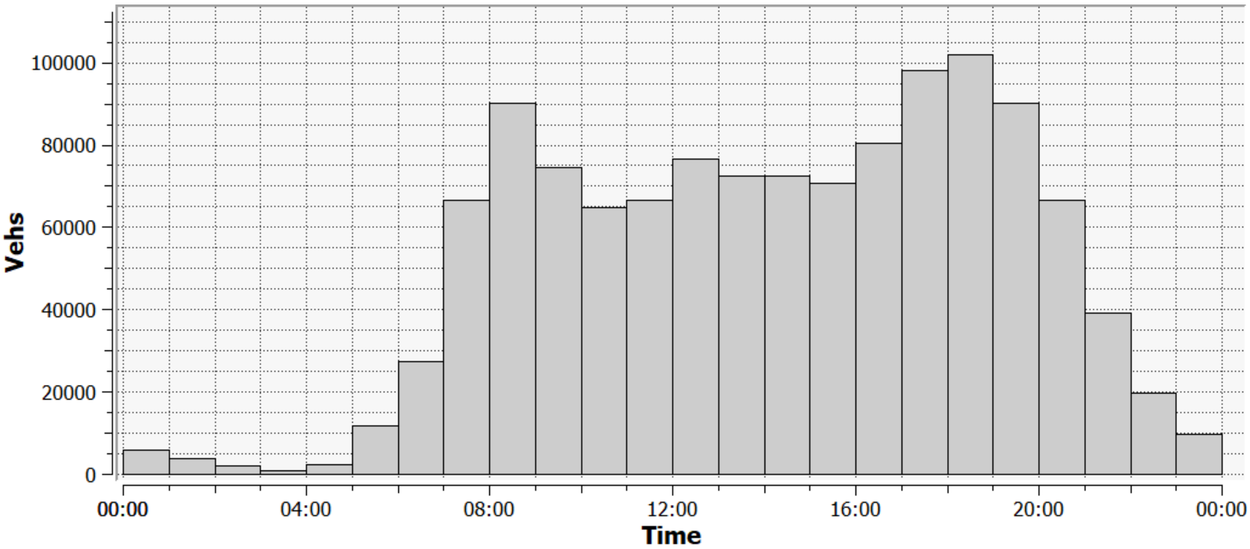


Figure 50 Profile of traffic demand applied to the network

Each hour was then assigned separately, and the resulting link speeds stored as traffic conditions, so that these could be used in the simulation to represent background traffic conditions.

[10] p.23 of Figure Distribución horaria de los desplazamientos según el motivo del desplazamiento. https://doc.atm.cat/es/_dir_pdm2020-2025/Memoria_2020.2025_castellano.pdf ¿CÓMO NOS MOVEREMOS? PLAN DIRECTOR DE MOVILIDAD 2020-2025

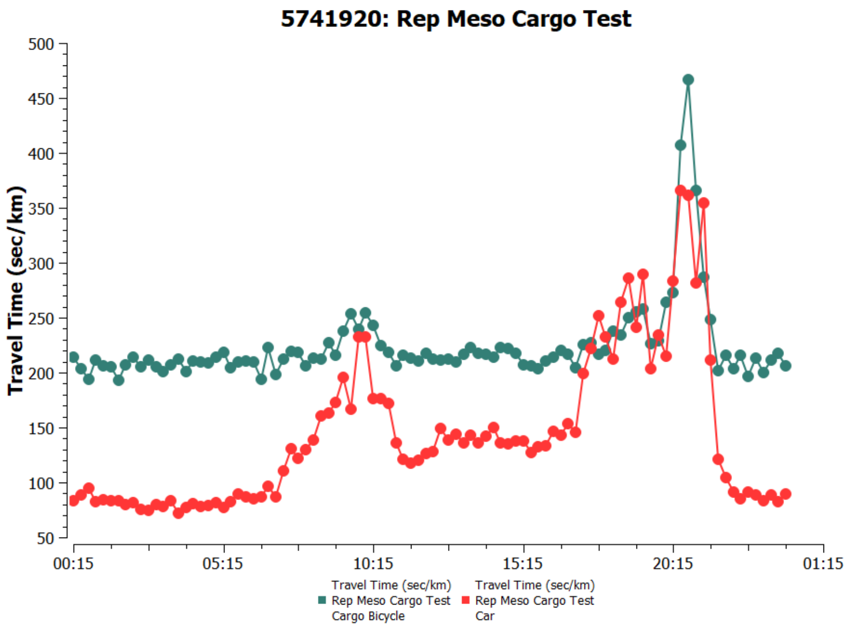


Figure 51 Resulting travel times on initial assignment of trial demand using traffic management to reproduce macro network conditions.

Since no traffic signal data was available, nor would be practical to code for this application, for each traffic signal location as indicated in the network, a fixed delay was applied to relevant turnings. Similarly, a separate delay was applied at each stop or yield location. These values may then be interpreted by the mesoscopic simulator if enabled.

5.3.9.5 Athens Model

The Athens City Inner Ring (Daktylios) has been selected as the area of interest for modelling in the Athens LL, due to the alignment with the boundaries of the Low Emission Zone and the city agenda to increase sustainability in the region. The developed model is intended to be used for planning purposes, including white labelling mobile deport schemes and MCCs for multi-service provider collaboration. For the model development, only open data was available which limits the model's sophistication and representativeness. However, given the strategic use-case scope, the model has been rendered adequate for a first system analysis. The following steps have been implemented.

5.3.9.6 Supply

As a starting point, OSM highway data was exported for a wider Athens area and a subnetwork was selected, given the scope of the Athens use-case.

Figure 52 Athens subnetwork in context (green)

Figure 53 Detail of Athens subnetwork

Additionally, the original XML tags associated with the nodes, ways and relations were imported into the model from the OSM file, and then used to improve the quality of the network inside the subnetwork, adding details such as way and lane reservations, yield indicators, ways where reverse cycle passage is permitted, pedestrian crossing locations and simplify some signalized intersections.

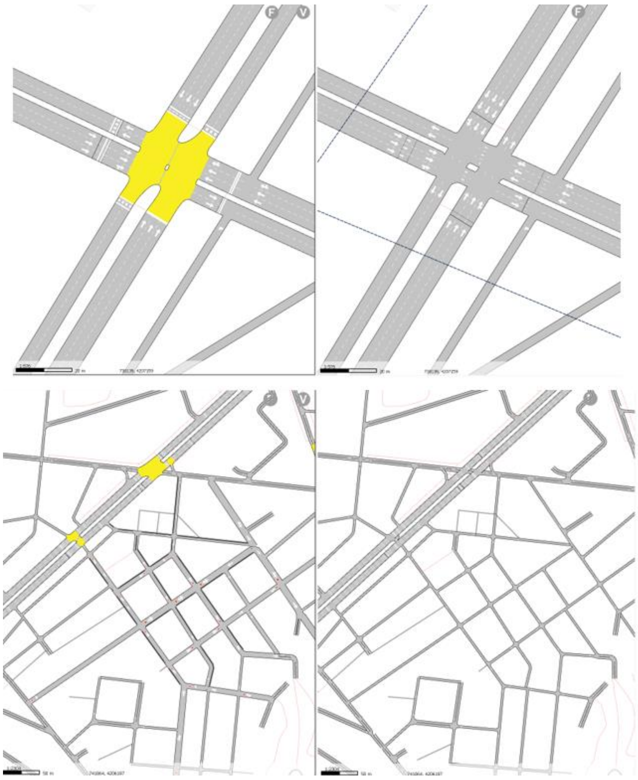


Figure 54 Examples of detail added to network (left) compared with original import (right)

Subnetwork Summary
The resulted subnetwork graph has the following attributes:

- Total Section Length (in km): 341,
- Total Lane Length (in km):
- Sections: 9311
- Nodes: 4387

No Centroids: no demand data so free flow conditions

5.4 Implementation Approach / Description / Results

The simulation service components are currently being integrated into GREEN-LOG's Augmented Intelligence Modelling Platform, which will be described in D2.2 in M21 – the simulation service and joint optimization-simulation service deployment is ultimately envisioned to take place via the platform for the Oxford, Athens and Barcelona LLs' planning and operational applications. However, till M18, the joint optimization-simulation service has been partly prototyped offline and in a non-automated way.
More specifically, a single full iteration of the integrated optimization-simulation components has been performed (given the methodology presented in Section 5.2) for the Oxford LL and the use-case described in Section 4.3.3. Given the service fleet

schedules computed by the optimization service described in Section 4, the simulation service is deployed as follows:

Step 1: Launching the (main) operator-simulator orchestrator Python script

This process can be done in two different ways:

1. via a Terminal/Command Prompt application by running the following command:

python orchestrator.py --operator orchestrator_definition.json

where, orchestrator.py is the orchestrator Python script and orchestrator_definition.py is the configuration of the main "dummy" operator fleet attributes, or

2. via an Integrated Development Environment (IDE), e.g., Visual Studio Code (preferred way)

Step 2: Launch aconsole, i.e.,, an application that runs the simulator in parallel with the orchestrator

"pathToAimsunInstallationDirectory\aconsole" -log --verbose --project oxfordModel.ang --command execute --target replicationId --mod-scenario scenario.json

---

The concurrent launching process is also illustrated in Figure 55 and Figure 56. Once the simulation process has terminated its execution, the fleet schedule travel times are written automatically in JSON file and a recording file (.ARF) is created which enables the visualization of the service simulation in the Graphical User Interface of Aimsun Ride (Next) as illustrated in Figure 57.

Co-funded by
the European Union

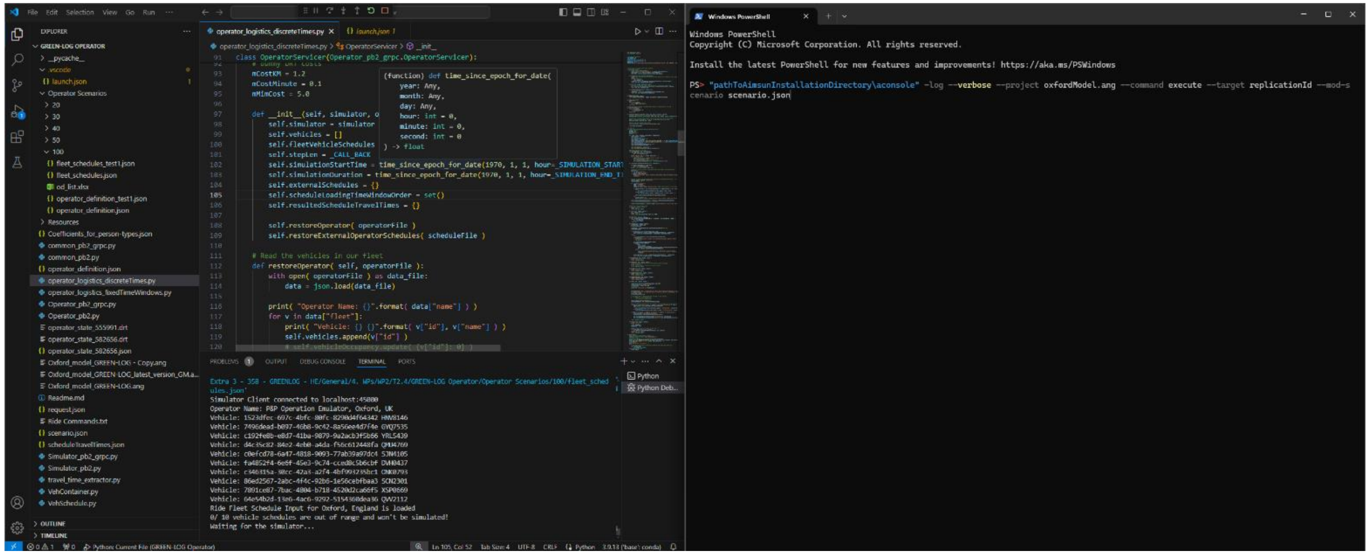


Figure 55 Launching orchestrator Python File and the aconsole (simulator) application

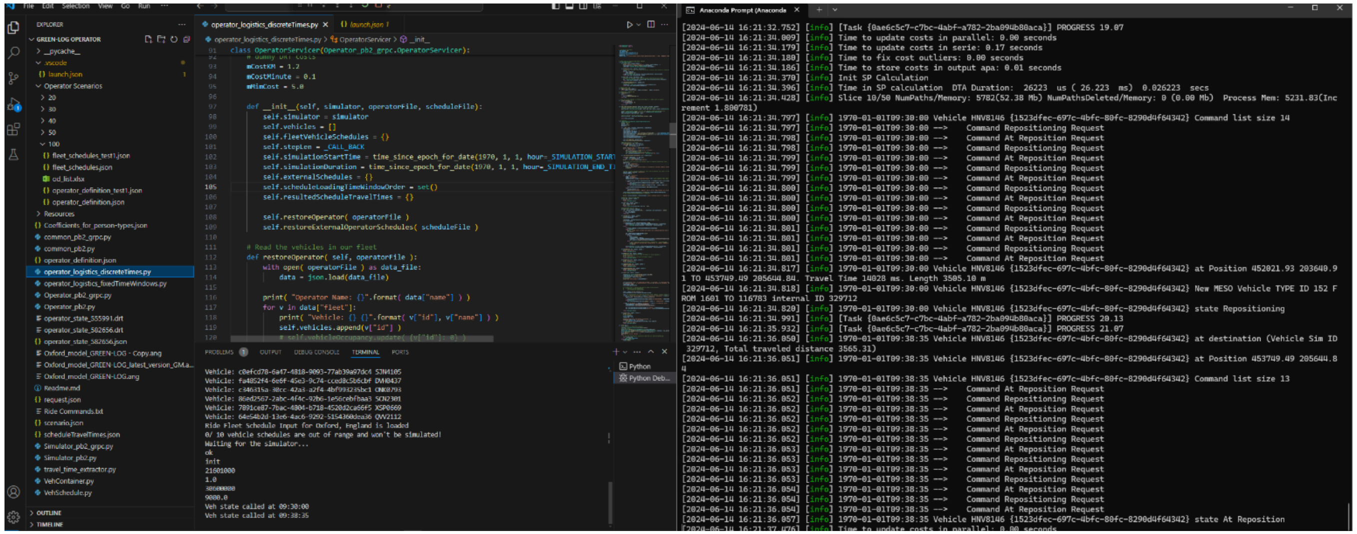


Figure 56 Concurrent simulation service execution

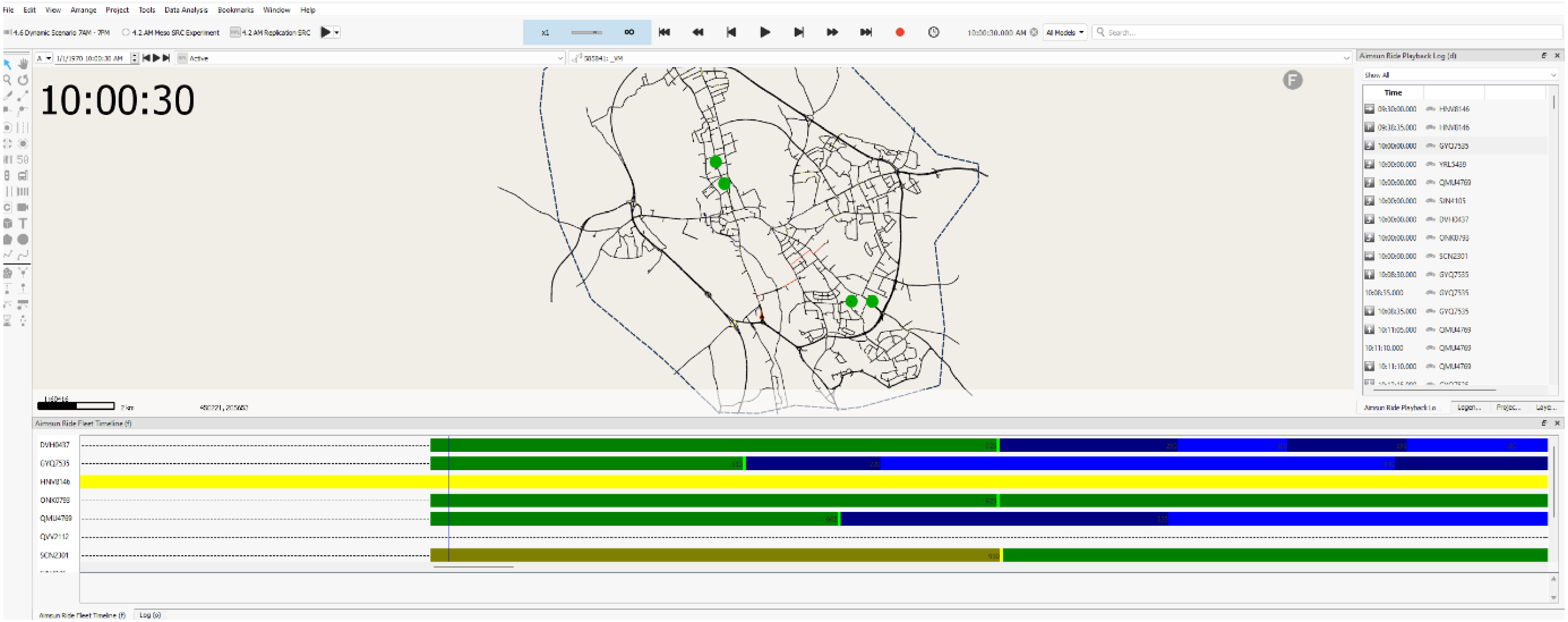


Figure 57 Visualization of Fleet Vehicle Movements from the Graphical User Interface of Aimsun Next

Aimsun Ride may produce a wide range of raw simulation data and KPIs either regarding the last-mile logistics service/fleet or network performance and sustainability indicators.

5.5 Future Work

For the simulation service, future work will focus on four main activities, which are the following:

1. Network Modelling
    a) While the Oxford and Athens network models have been largely completed (minor ad-hoc adjustments might have to be made if needed), the Barcelona model is not yet complete. Priority till M21 will be given to enhance the model with geometry hacks which will allow is to increase the realism of the traffic conditions given very limited data availability
2. Aimsun Ride enhancements
    a) The internal routing functionalities of Aimsun Ride will be enhanced to ensure that all schedule items regardless of origin and destination locations will be simulated with reasonable and feasible routes/paths given the highly disaggregate spatial resolution of the parcel delivery simulation problem with cargo bikes and dedicated cycling infrastructure.
3. Simulation Service Integration into GREEN-LOG's Augmented Intelligence Modelling Platform and Testing
    a) A close collaboration has been established with relevant partners and MOSAIC FACTOR in particular to ensure an interoperable and seamless integration of the simulation service and the communication with other components (e.g., optimization service or data space).
4. Last-mile logistics Simulation Scenario Analysis and Assessment
    a) For all LLs (Oxford, Athens, Barcelona) where the simulation service will be deployed along with the optimization service, extensive scenario analysis and evaluation based on pre-defined KPIs will be implemented to ensure either efficient operations for LSPs in an operational setting (Oxford and PEDAL) or planning and analysis for hypothetical infrastructure and concept scenarios (Barcelona and VANaPED, Athens).

6 Conclusion

Initial work has been presented which covers the specification and prototyping of the technical components D2.1 "Services for orchestrated deployment of new delivery methods" and D2.2 "Augmented intelligence modelling platform".

We have designed and studied logistics demand datasets with real-life characteristics, testing various efficient demand models and analysing their performance on our designed datasets, reflecting our project's objectives and insights. We also had a detailed summary of our main inquiries about the current and desired states of demand prediction within the project, and then conducted an in-depth discussion with our logistics partner PEDAL, our project team, and other partners. In the next step, based on our discussion with our project partners, we will do further improve our logistics demand models and generated demand datasets in order to make it further closer to the real and complex environment of our project, and also carry on more investigations in order to gain better understanding about the operational details, demand patterns, forecasting needs, data limitations, and metrics, and further refine our demand prediction model effectively, and continue to maintain close communication with our project partners and obtain a further insight of both the current operational state and the aspirations post-project completion, facilitating a more targeted and efficient approach to demand prediction modelling in our logistics demand project, and we further apply our built model on real-life big dataset and test our prediction model on the real-life PEDAL dataset. Moreover, we will do a detailed analysis of the PEDAL dataset and separate important information from useless information inside the large-scale dataset and do some adjustment of our prediction model, in order to further improve its performance on the real-life large-scale demand dataset.

We have run several preliminary experiments on the available data from PEDAL to get insights on the problem that needs to be tackled and start having an overview of different ways to provide insights which may be useful for delivery services. We need to, first, confirm that all data gathering from PEDAL is automated so that the implementation of a potential analytics service can be done. Then, once data providers allow communication of data feeds, a protocol needs to be established which comply with GDPR standards. Once data verification is successful, experiments need to be re-run in order to have a complete view and analysis of their outcome.

In this preliminary report, we have observed data inconsistencies which can be interpreted as data gaps from manual interference or merging of data sources or actual shifts in demand that could be recurrent. In case the second hypothesis applies, collecting more data from historical periods could prove this trend or reject it and hence a decision on whether peaks should be removed needs to be taken.

With respect to statistical and machine learning methodologies, preliminary results show that traditional regression techniques such as random forest or XGboost are adequate to tackle aggregating demand forecasting for 5-day time horizons. Moving averaged and deep learning are discarded due to their limited capabilities or demanding requirements.

The implementation of a data space based on the IDS-RAM specification, is implementing a previously agreed standard group of components.

7 References

Reference	Name of document
[REF-01]	GREEN-LOG. D6.1: Project Reference Manual. 28-02-2023.
[REF-02]	Halper, R., Raghavan, S., “The Mobile Facility Routing Problem” Transportation Science. 45:3, pp. 413-434, 2011.
[REF-03]	Hof, J., Schneider, M., “Intraroute Resource Replenishment with Mobile Depots” Transportation Science. 55:3, pp. 660-686, 2021.
[REF-04]	Prodhon, C., Prins, C., “A survey of recent research on location-routing problems” European Journal of Operational Research. 238, pp. 1-17, 2014
[REF-05]	Li, H., Wang, H., Chen, J., Bai, M., “Two-echelon vehicle routing problem with time windows and mobile satellites” Transportation Research Part B. 138, pp. 179-201, 2020.
[REF-06]	Mousavi, K., Bodur, M., Roorda, M.J., “Stochastic Last-Mile Delivery with Crowd-Shipping and Mobile Depots” Transportation Science. 56:3, pp. 612-630, 2022.
[REF-07]	Boeing, G., “OSMnx: New Methods for Acquiring, Constructing, Analysing, and Visualizing Complex Street Networks.” Computers, Environment and Urban Systems. 65, pp. 126-139, 2017.
[REF-08]	Abbasimehr, H., Shabani, M., Yousefi, M. “An Optimized Model using LSTM Network for Demand Forecasting”, Computers & Industrial Engineering. 143, pp. 106435, 2020.
[REF-09]	Dou, Z., Sun, Y., Zhang, Y., Wang, T., Wu, C., Fan, S. “Regional Manufacturing Industry Demand Forecasting: A Deep Learning Approach”, Applied Sciences, 11:13, pp. 6199, 2021.
[REF-10]	Raizada, S., Saini, J. “Comparative Analysis of Supervised Machine Learning Techniques for Sales Forecasting”, International Journal of Advanced Computer Science and Applications (IJACSA), 12:11, pp. 51-54, 2021.
[REF-11]	Wang, J., Liu, G., Liu, L. “A Selection of Advanced Technologies for Demand Forecasting in the Retail Industry”, 2019 IEEE 4th International Conference on Big Data Analytics (ICBDA), pp. 317-320, 2019.
[REF-12]	Thies, P., Johanning, L., Gordelier, T. “Component Reliability Testing for Wave Energy Converters-rationale and Implementation”, Proceedings of the European Wave and Tidal Energy Conference, 2013.
[REF-13]	Reikard, G. “Forecasting Ocean Wave Energy: Tests of Time-series Models”, Ocean Engineering, 36:5, pp. 348-356, 2009.
[REF-14]	Duan, W-Y., Huang, L-M. “A Hybrid EMD-AR Model for Nonlinear and Nonstationary Wave Forecasting”, Journal of Zhejiang University Science, 17, pp. 115-129, 2016.
[REF-15]	Ozger, M., Altunkaynak, A., Sen, Z. “Statistical Investigation of Expected Wave Energy and Its Reliability”, Energy Conversion and Management, 45, pp. 2173-2185, 2004.

[REF-16]	Mahjoobi, J., Etemad-Shahidi, A. “An Alternative Approach for the Prediction of Significant Wave Heights based on Classification and Regression Trees”, Applied Ocean Research, 30, pp. 172-177, 2008.
[REF-17]	Etemad-Shahidi, A., Mahjoobi, J. “Comparison between M5 Model Tree and Neural Networks for Prediction of Significant Wave Height in Lake Superior”, Ocean Engineering, 36, pp. 1175-1181, 2009.
[REF-18]	Berbiu, J., Ocvirk, E., Careviu, D., Loncar, G. “Application of Neural Networks and Support Vector Machine for Significant Wave Height Prediction”, Oceanologia, 59, pp. 331-349, 2017.
[REF-19]	Malekmohamadi, I., Bazargan-Lari, M., Kerachian, R., Nikoo, M., Fallahnia, M. “Evaluating the Efficacy of SVMs, BNs, ANNs, and ANFIS in Wave Height Prediction”, Ocean Engineering, 38, pp. 487-497, 2011.
[REF-20]	Kazeminezhad, M. H., Etemad-Shahidi, A., Mousavi, S. J. “Application of Fuzzy Inference System in the Prediction of Wave Parameters”, Ocean Engineering, 32, pp. 1709-1725, 2005.
[REF-21]	Adamowski, J., Chan, H. F. “A Wavelet Neural Network Conjunction Model for Groundwater Level Forecasting”, Journal of Hydrology, 407, pp. 28-40, 2011
[REF-22]	Law, A.M., Kelton, W.D. and Kelton, W.D., 2007. Simulation modeling and analysis (Vol. 3). New York: Mcgraw-hill.
[REF-23]	Wolf, F., Engelhardt, R., Zhang, Y., Dandl, F. and Bogenberger, K., 2023, September. Effects of Dynamic and Stochastic Travel Times on the Operation of Mobility-on-Demand Services. In 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC) (pp. 5476-5481). IEEE.
[REF-24]	Rabe, M., Gonzalez-Feliu, J., Chicaiza-Vaca, J. and Tordecilla, R.D., 2021. Simulation-optimization approach for multi-period facility location problems with forecasted and random demands in a last-mile logistics application. Algorithms, 14(2), p.41.
[REF-25]	Sawik, B., Faulin, J., Serrano-Hernandez, A. and Ballano, A., 2022, December. A simulation-optimization model for automated parcel lockers network design in urban scenarios in Pamplona (Spain), Zakopane, and Krakow (Poland). In 2022 Winter Simulation Conference (WSC) (pp. 1648-1659). IEEE.
[REF-26]	Costa, P.C., Cunha, C.B. and Arbex, R.O., 2021. A simulation-optimization model for analyzing a demand responsive transit system for last-mile transportation: A case study in São Paulo, Brazil. Case Studies on Transport Policy, 9(4), pp.1707-1714.
[REF-27]	Perboli, G., Rosano, M. and Wei, Q., 2021. A simulation-optimization approach for the management of the on-demand parcel delivery in sharing economy. IEEE Transactions on Intelligent Transportation Systems, 23(8), pp.10570-10582.
[REF-28]	Snoeck, A. and Winkenbach, M., 2022. A discrete simulation-based optimization algorithm for the design of highly responsive last-mile distribution networks. Transportation Science, 56(1), pp.201-222.
[REF-29]	Barceló, J., 2010. Fundamentals of traffic simulation (Vol. 145, p. 439). New York: Springer.

[REF-30]	Yfantis, L., Stebbins, S., Gerostathopoulos, I., Djukic, T., Casas, J., Garcia, D., Kamargianni, M. and Chaniotakis, M., 2021, June. A software-agnostic agent-based platform for modelling emerging mobility systems. In 2021 7th International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS) (pp. 1-6). IEEE.
[REF-31]	Narayanan, S., Salanova Grau, J.M., Frederix, R., Tympakianaki, A., Masegosa, A.D. and Antoniou, C., 2023. Modeling of shared mobility services-An approach in between aggregate four-step and disaggregate agent-based approaches for strategic transport planning. Journal of Intelligent Transportation Systems, pp.1-18.