Progress towards Federated Logistics through the Integration of TEN-T into A Global Trade Network

D2.2 Open EGTN Platform Architecture final version

Document Summary Information

Grant Agreement No	860274	**Acronym**	PLANET
Full Title	Progress towards Federated Logistics through the Integration of TEN-T into A Global Trade Network		
Start Date	01/06/2020	**Duration**	36 months
Project URL	www.planetproject.eu		
Deliverable	D2.2 Open EGTN Platform Architecture Final Version		
Work Package	WP2 PLANET Cloud-based Open EGTN Infrastructure		
Contractual due date	31-Dec (M31)	**Actual submission date**	22-Dec (M31)
Nature	Report	**Dissemination Level**	PU
Lead Beneficiary	INLE		
Responsible Author	Harris Niavis (INLE)		
Contributions from	Gerasimos Kouloumbis (INLE), Andreas Alexopoulos, Aristea Zafeiropoulou, Antonis Mygiakis (KNT), Philippos Philippou (eBOS), Moises Sanchez, John Farren (IBM), Georgia Ayfantopoulou (CERTH), José Luis López Ramón, David Cipres (ITAINNOVA), Claudio Salvadori (NGS)		

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Grant Agreement No 860274.

Revision history (including peer reviewing & quality control)

Version	Issue Date	% Complete[1]	Changes	Contributor(s)
v0.1	10/10/22	10%	Initial Deliverable Structure	Harris Niavis (INLE)
v0.2	30/11/22	30%	Added section 3.5 Modelling and Simulation. Also added content all over the document	Harris Niavis (INLE), José Luis López Ramón, David Cipres (ITAINNOVA)
v0.3	8/12/22	60%	Added section 3.1 section 3.3, 3.5, 4.1, 4.2.	Harris Niavis (INLE)
v0.4	9/12/22	70%	Added section 5.	Harris Niavis (INLE), Andreas Alexopoulos (KNT)
v0.5	12/12/22	90%	Added sections 6, 7.	Harris Niavis (INLE)
v0.6	21/12/22	95%	Integrated feedback from internal review	John Farren (IBM), Aristea Zafeiropoulou (KNT), John Limaxis (INLE)
v0.7	22/12/22	100%	Final version	Harris Niavis, Gerasimos Kouloumpis (INLE)

Disclaimer

Copyright message

1 According to PLANET's Quality Assurance Process

Table of Contents

List of Figures

List of Tables

Glossary of terms and abbreviations used

Abbreviation / Term	Description
ACL	Access Control Lists
BLE	Bluetooth
CI/CD	Continuous Integration / Continuous Development
DAG	Directed Acyclic Graph
EGTN	EU-Global T&L Networks
ETL	Extract Transform Load
GQL	GraphQL
JAAS	Java Authentication and Authorization Service
KG	Knowledge Graph
LL	Living Lab
LTS	Long Term Support
MAMCA	Multi Actor Multi Criteria Analysis
OD	Origin Destination
PoI	Point of Interest
SASL	Simple Authentication and Security Layer
TRL	Technology Readiness Level
T&L	Transport & Logistics
UC	Use Case
UI	User Interface

1 Executive Summary

The Cloud-based Open EGTN Infrastructure Architecture deliverable reports on the creation of an open-source blueprint that aims to empower organisations to build upon and implement Transport & Logistics (T&L) design tools, collaborative logistics and new eCommerce models underpinned by data-driven supply chain insights. The deliverable aims to inform any stakeholder or consortium of stakeholders involved or interested in the design of innovative, cross-organisational EU-Global T&L networks.

The document explains the reasoning behind the development of the features provided, by mapping the requirements presented in the previous version with the final functionalities offered by the EGTN Platform. It materialises the platform specification defined in WP1 to an integrated architecture and a cloud-based instantiation of it on the cloud. A description of the components of the platform is presented together with detailed deployment strategies aiming to ensure that the platform can be easily adopted by any interested T&L party. The governance model of the platform defines processes for data ingestion and processing, integration and management of PI services and onboarding of users.

The EGTN Platform is designed to meet the particular needs of the T&L industry by implementing a customised combination of technologies and models. Data Ingestion is handled by versatile mechanisms responsible for importing heterogeneous data from various external sources in batch and/or in real time in a secure manner. A Decision Support System (DSS) allows the users to make important T&L and PI decisions, such as corridor route optimisations, warehouse time reductions etc. The DSS provides data intelligence and is based on different Machine Learning (ML) models, as well as simulation mechanisms. The results of these models and simulations are the basis for the decision-making process. Intelligent pallet and container volume forecasting and route optimisation is used with the ultimate purpose of achieving the PI roadmap, while the use of smart contracts facilitates automated and paperless negotiations. Blockchain interoperability aims to overcome the silos of the different Blockchain systems/partners, and, finally, user-accessible dashboards within the Human Machine Interface offer a visual frontend to all stakeholders. Most of the underlying technical solutions used throughout the implementation of the EGTN Platform are industry standard, production proven and open source.

The combination of these technical advancements plays an instrumental role in uniquely positioning the EGTN Platform in terms of technical enablement of the Physical Internet (PI) paradigm. In that context, the EGTN Platform empowers T&L stakeholders by offering them a means to access tools and PI services for routing, node optimisation, shipping, and encapsulation, and so on, as well to collaborate with other T&L actors across borders and organisations in a self-determined and secure way.

2 Introduction

2.1 Mapping PLANET Outputs

Purpose of this section is to map PLANET's Grant Agreement commitments, both within the formal Deliverable and Task description, against the project's respective outputs and work performed.

Table 1: Adherence to PLANET's GA Deliverable & Tasks Descriptions

PLANET GA Component Title	PLANET GA Component Outline	Respective Document Chapter(s)	Justification
DELIVERABLE			
D2.2 Open EGTN Platform Architecture Final Version	Final version of Open EGTN Platform Architecture complying to the updated requirements from T1.5	Chapter 3 -7	Chapters 3 -7 present the final version of the EGTN Platform architecture, how the offered functionalities map with the requirements listed in the previous version of the deliverable and also present the value to the T&L industry.
TASKS			
T2.1 Cloud-based Open EGTN ICT Infrastructure Architecture	The objective of this task is to specify, design and prototype an open Cloud-based platform that will provide stakeholders with a low entry cost (open) collaboration platform for sustainable integrated multimodal freight transport.	Chapter 3	The design and architecture of the EGTN Platform can be found in chapter 3, which describes the components, internal and external APIs, governance model and aspects for its deployment on the Cloud.
	The task will create an open-source blueprint that enables organisations to build upon and to implement T&L design tools, collaborative logistics and new eCommerce models underpinned by Big Data driven Supply Chain insights.	Chapter 7	Chapter 7 describes the architectural blueprint as one of the outputs of T2.1, together with its innovation potential and its value to the T&L industry that enables collaboration between actors and new eCommerce models.
ST2.1.1 Specification of a Cloud Platform for sustainable integrated multimodal	Specification of a Cloud Platform for sustainable integrated multimodal freight transport: Based on stakeholder needs analysis (Task 1.5) and experience gained from other logistics collaboration platforms from	3.1 – 3.5	The design and the architecture of the EGTN Platform can be found in sections 3.1 to 3.5.

freight transport	other Horizon 2020 projects (SELIS & Aeolix), this task will provide an architectural blueprint for a Cloud based platform.		
	To maximize acceptance and applicability, different Cloud models such as Public, Private, Hybrid and federated will be considered and supported.	3.6	Deployment aspects of the EGTN Platform and the flexibility of the architecture are explained in section 3.6.
ST2.1.2 Design for service integration, governance, privacy and cybersecurity	The platform will be implemented from the ground up to be secure, provide seamless integration of logistics services, and be based on an explicit governance model for onboarding of users, data & services, and providing a customized environment for specifying privacy policies.	Chapter 4, 6 3.5	Chapter 4 lists the PI Services provided by the EGTN Platform while section 3.5 describes the governance model for onboarding users, data and integrating services. Chapter 6 presents privacy and security considerations of the platform.
	Key assets from SELIS, CHARIOT, ICONET AND FENIX will help accelerate implementation.		The previous version of this deliverable described how assets developed in other projects inspired the design and development of the EGTN architecture.
ST2.1.3 Platform Implementation and Deployment	This subtask will prototype the Platform and make it available for integration of data and services developed by other WP2 tasks, and deployment across PLANET's Living Labs.	3.3, 5.3	Section 3.3 presents the EGTN Infrastructure components on the Cloud that enable other WP2 tasks to integrate services and data. Section 5.3 describes use cases of data residing in the platform, showing the complete flow of the data from ingestion to the platform to visualization in the EGTN Dashboard.

2.2 Deliverable Overview and Report Structure

The deliverable is organised in separate chapters as follows:

- **Chapter 3** provides an overview of the EGTN Platform, starting with mapping the requirements defined in the first version of the deliverable, namely D2.1, with functionalities provided by the platform. Subsequently, it describes the design principles followed and the architecture of the system together with a description of the different infrastructure components. This chapter also discusses the governance model for data, users and services, aspects of deploying an instantiation of the architecture on the cloud and finally describes how the EGTN infrastructure enables the development of PI services.
- **Chapter 4** presents the EGTN Services, dividing them into i) infrastructure utilities for enabling secure data ingestion, users onboarding and PI services integration and orchestration and ii) PI Services which

are the software developed by the tasks T2.2, T2.3, T2.4, T2.5, T2.5 for values such as increased visibility in the supply chain, optimised routing and more efficient last mile delivery.

- **Chapter 5** focuses on the EGTN Data Lake where heterogenous data from multiple sources are stored and made available to the PI Services.
- **Chapter 6** analyses security and privacy considerations for the architectural blueprint proposed by the project, while it sets up more specific recommendations for secure data integration and data management.
- **Chapter 7** recounts the state of the art in modern T&L networks and illustrates how the EGTN innovations take a step beyond the current state of the art and finally describes its value to the industry.
- **Chapter 8** summarises the deliverable and highlights its key outputs.

3 The EGTN Cloud-based Platform

The EGTN Platform integrates infrastructure such as an Event Streaming Service, a Data Lake, monitoring interfaces, a dashboard with a User Interface (UI) as well as PI services i.e., predictive analytics for logistics operations, PI route optimisation, decision support systems, and provides seamless integration of these components based on an explicit governance model for onboarding users and data.

This section presents the mapping of the requirements to functionalities provided by the Platform, the high-level design and architecture of the EGTN Platform, components description including the modelling and simulation components, the platform governance, deployment on the cloud details and the value that the EGTN infrastructure brings to the EGTN PI Services.

3.1 Mapping of Requirements

The functional and non-functional requirements of the EGTN Platform have been defined and analysed based on feedback from WP1 and have been presented in the first version of this deliverable, namely D2.1, using a a Usage-Actors-Requirements approach. The mapping of those requirements to functionalities of the EGTN Platform are presented in Table 2 through Table 7.

Business Scenario	ID	Functional Requirement	Respective EGTN Platform functionalities
A. Add / Remove data from the platform	A.1	Add data in any agreed format to the platform	The Data Aggregator of the EGTN Platform provides the necessary interfaces for the ingestion of data of any format and structure, namely live data through the Event Streaming Service and/or batch datasets in MongoDB and/or InfluxDB for timeseries data, from T&L stakeholders (see section 3.3 and section 5).
	A.2	Provide metadata for every file and storage (e.g. sensitivity, keyword and tags, short description, owner, uploader)	Metadata collection in MongoDB for every kafka-topic, collection, measurement (influxDB).
	A.3	Provide volumetric information for validation purposes (see B)	Volumetric information in the form of object count (in Mongo) events count (Kafka) and point count (InfluxDB) stored in the MongoDB metadata collection (see A.2).
	A.4	Securely add data following the security and data encryption requirements on a case-by-case basis (most stringent applies, min AES128)	Any data stored in the EGTN servers are encrypted using strong cryptographic algorithms (AES-256) to enhance the accountability of the platform.
	A.5	Support continuous streaming data	The Event Streaming Service provides support for continuous streaming of data e.g. live IoT data from

			sensors or events from blockchain systems. (see section 5.1.1)

Table 2: Usage Scenario A: Add / Remove data from the platform

Business Scenario	ID	Functional Requirement	Respective EGTN Platform functionalities
B. Pre-processing data ready for analysis	B.1	Ability to extract, transform and load from a wide variety of file and stream formats	The Apache Airflow enables this feature in the platform (see more in section **Error! Reference source not found.**).
	B.2	Ability to automate the pre-processing of the data from multiple, varied sources (structured)	The Apache Airflow enables this feature in the platform (see more in section **Error! Reference source not found.**).
	B.3	Ability to automate the pre-processing of the data from multiple, varied sources (unstructured)	The Apache Airflow enables this feature in the platform (see more in section **Error! Reference source not found.**).
	B.4	Ability to respect data sensitivity attached to data sources and to label transformed outputs appropriately (see D3.1)	The metadata collection in MongoDB, mentioned in requirement A.2 facilitates this functionality, since the output metadata are also stored in this collection.
	B.5	Maintain metadata for pre-processing and log and report associated pre-processing activity	Apache Airflow provides metadata information for each pre-processing ETL activity.
	B.6	Ability to validate data (and report exceptions) based on metadata and volumetric information provided in scenario A	The Data Aggregator component enables the validation of data based on their metadata and volumetric information.
	B.7	Securely clean-up all workspaces, VM's and mount points	Kubernetes facilitates the clean-up of workspaces, VM's and mount points.

Table 3: Usage Scenario B: Pre-processing data ready for analysis

Business Scenario	ID	Functional Requirement	Respective EGTN Platform functionalities
C. Visualise and analyse data	C.1	Ability to run supervised and semi-supervised analytics and models	The platform offers Zeppelin notebooks that can be used for the development of supervised models which run using the EGTN data as input. The notebooks have

			access to the EGTN data either through the Kafka topics or from the filesystem itself with volume mounts on the notebook pods (see section 3.3.3).
	C.2	Ability to run unsupervised / unattended analytics and models	The platform offers Zeppelin notebooks that can be used for the development of unsupervised models which run using the EGTN data as input. The notebooks have access to the EGTN data either through the Kafka topics or from the filesystem itself with volume mounts on the notebook pods (see section 3.3.3).
	C.3	Export reports in a range of formats (e.g., pdfs, images, Office) from unattended models	The platform offers Zeppelin notebooks that can be used to export reports in different formats (see section 3.3.3).
	C.4	Run classical statistics and models based on many, varied sources (unstructured and unstructured)	The platform offers Zeppelin notebooks that can be used for the development of classical statistics and models which run using the EGTN data as input. The notebooks have access to the EGTN data either through the Kafka topics or from the filesystem itself with volume mounts on the notebook pods (see section 3.3.3).
	C.5, C.6	Ability to fully integrate with off-site training facilities for model creation (e.g., EC2)	The integration with an EC2 instance in the Amazon Cloud was achieved through Apache Airflow which can offload model creation to EC2. The integration reported in the first version of this deliverable, but the project didn't finally engage the instance since the amount of Living Lab real data collected and analyzed was such, that local computational capacity provided by the service providers could serve it.
	C.6	Ability to run all analytics server side or via EC2	The integration with an EC2 instance in the Amazon Cloud was achieved through Apache Airflow which can offload model creation to EC2. The integration reported in the first version of this deliverable, but the project didn't finally engage the instance since the amount of Living Lab real data collected and analyzed was such, that local computational capacity provided by the service providers could serve it.
	C.7	Ability to perform a wide range of statistical analysis techniques on T&L data and business outcome data	Zeppelin supports a wide range of techniques and models while Grafana supports data exploration and more traditional statistical analysis.
	C.8	Ability to use any open-source visualisation library (subject to license), and to	Grafana extensions/plugins support the creation of new custom visualisations.

		create new, custom visualisations	
	C.9	Ability to create and run agent-based models	Zeppelin supports the creation of agent-based models which are deployed through Apache Airflow.

Table 4: Usage Scenario C: Visualise and analyse data

Business Scenario	ID	Functional Requirement	Respective EGTN Platform functionalities
D. Building algorithms and analytical workflows	D.1	Develop the algorithms and workflows in-situ on the Platform	Apache Zeppelin is deployed to enable the development of algorithms in the platform.
	D.2	Collaborative development of algorithms, workflows and code	Apache Zeppelin enables the collaborative development of algorithms, workflows and code in the platform.
	D.3	Code management for source and executable management	The project's GitLab repository is employed for code management.
	D.4	Respect a deny list (i.e., data objects that a) the user has no permission to see and b) should not be combined for legal, commercial or analytical reasons)	ACLs in Apache Kafka and users in MongoDB
	D.6	Version control the algorithms and workflows	The project's GitLab repository supports version control of the algorithms and workflows.
	D.7	Ability to automate the algorithms and workflows so they can run unattended	Apache Airflow is deployed which enables the unattended execution of the algorithms and workflows.
	D.8	Ability to integrate Bluemix and EC2 into workflows	Apache Airflow is employed for the integration with EC2 (see C.5).
	D.9	Ability for workflows to respect data sensitivity attached to data sources and outputs	The metadata collection in MongoDB enables this functionality (see B.4).
	D.10	Develop workflows and algorithms using multiple tools on the same platform	Apache Zeppelin is used to enable this functionality in the platform.

Table 5: Usage Scenario D: Building algorithms and analytical workflows

Business Scenario	ID	Functional Requirement	Respective EGTN Platform functionalities
E. Automation of analytical workflows	E.1	Constructed pipelines should continuously operate under predefined performance criteria	Kubernetes supports elasticity for deployments based on CPU usage, memory etc.
	E.2	Algorithmic performance issues/timing should be raised	Kubernetes supports this with error reporting and warnings while algorithms should be written with logging.
	E.3	Data availability issues should be raised	Kubernetes supports this with error reporting and warnings while algorithms should be written with logging.
	E.4	Cost-constrained Scale up	Kubernetes provides ways to limit the elasticity based on values metrics such as CPU usage, pod count etc.

Table 6: Usage Scenario E: Automation of analytical workflows

Business Scenario	ID	Functional Requirement	Respective EGTN Platform functionalities
F. Curating and archiving data	F.1	Retain all for the duration of the project	The EGTN servers are maintained for the entire duration of the project.
	F.2	Flag datasets as out of service (e.g., superseded, defective etc)	The metadata collection in MongoDB enables this functionality.
	F.3	Version control datasets	Version control of the datasets is achieved through metadata in MongoDB.
	F.4	Provide daily snapshot backups	Daily backups are provided through Kubernetes cronjobs.
	F.5	Ability to back-up entire platform at regular intervals to tape	Not applied
	F.6	Maintain encryption during archiving (min. AES128)	Already provided by the infrastructure Provider (Hetzner).
	F.7	Ability to extract training sets in a controlled manner	Data can be easily extracted from different databases.
	F.8	Off-site archiving	Not applied

Table 7: Usage Scenario F: Curating and archiving data

3.2 Design and Architecture

The EGTN Platform facilitates data sharing, collaboration and decision making, through the integration of AI models and PI services to be easily implemented that answer analytics questions in real time using a plethora of data stemming from heterogeneous sources. The platform is designed according to the Archimate 3.1 Open Standard [1], further described in D2.1 and follows a "layered" architecture.

The EGTN architecture imports data both in real time and in batch from various sources, such as IoT networks and devices, using publish and subscribe mechanisms. The Data Aggregator is responsible for the provision of all the data to the entire EGTN Platform and to all EGTN actors through open APIs. The data flow is configured based on business stories and data sharing agreements – either in the form of a Knowledge Graph containing the contact semantics for data sharing, data usage and data access or in the form of direct access to the Data Lake of the infrastructure.

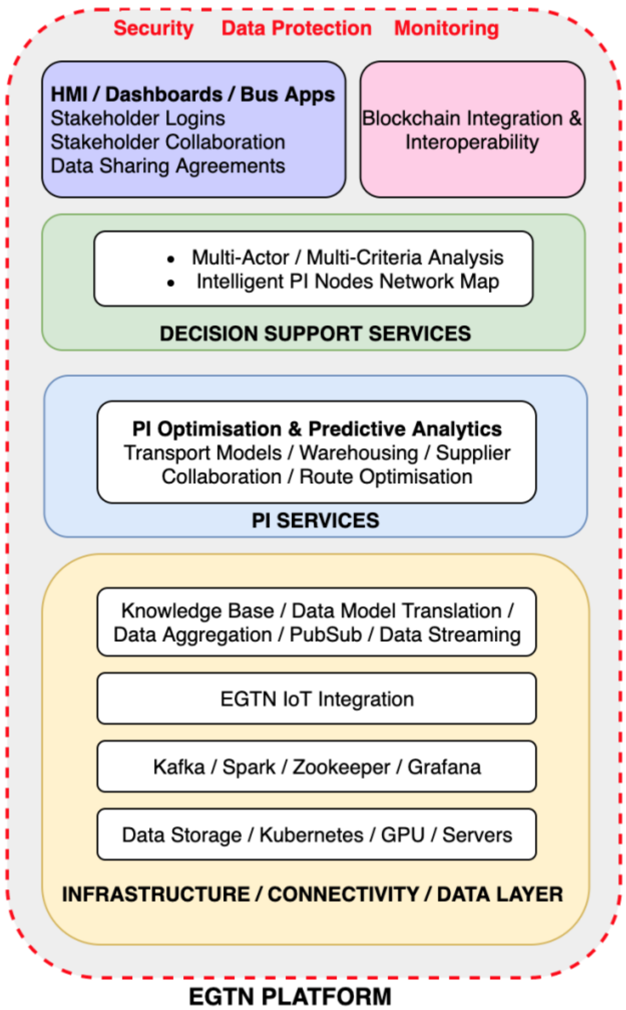

Figure 1: The EGTN Platform High-Level Architecture

As shown in Figure 1, the infrastructure/connectivity/data layer is comprised of the Data Aggregator for the connection to different data sources, the Data Storage, Kubernetes[2] for enabling a container-based infrastructure and open-source tools for enabling big data analytics (Apache Spark[3]), administration of the

[2] Kubernetes system, https://kubernetes.io/
[3] Apache Spark, https://spark.apache.org/

platform (Apache ZooKeeper[4] and Grafana[5]) and workflow management (Apache Airflow[6]). All the software utilities employed are open-source frameworks, supported by a large community of developers that maintain state-of-the-art functionalities and the long-term viability of the platform.

On top of the infrastructure layer are the PI Services layer i.e. decision support systems that consists of predictive and prescriptive models and whose goal is to decide upon the best transportation options. The PI Services empower all EGTN actors with better decision-making tools, so that they can in turn optimise planning and pre-planning, considering certain constrains, goals and limitations, such as lowest-cost route, fastest route, low-carbon route and so on. Following the decision-making process, contract negotiation and execution shall take place, using smart contracts.

Smart contracts are enabled through a Blockchain frontend, that exposes a decentralized network of transactions of multiple backend blockchain systems that connect to the EGTN Platform and share logistics and SLA management events in a transparent, immutable, trusted, and efficient manner. The integration of the Blockchain networks and the smart contracts within the EGTN are presented in D2.16 "Integration and Interoperability of proprietary Blockchain Systems for Seamless Global Trade Workflows final version" and D2.18 "EGTN smart contracts and associated PI motivated workflows in the context of SLA management final version". Finally, the EGTN Platform features a unified interface with custom dashboards that empower actors to manage all TEN-T corridors as well as the PI workflows in real time through a Human Machine Interface. More details on the HMI regarding its scope and implementation plan can be found in D2.20 "Unified HMI implementation and technical documentation Final version".

3.3 Infrastructure Components Description

The EGTN Platform is hosted at dedicated servers offered by Hetzner[7] cloud provider. The cloud provider selection process and the requirements have been presented in D2.1. Based on the project's needs, the servers have been utilized in the following manner:

- Two servers compose the Kubernetes cluster of the EGTN platform; a 12-Core / 24-Thread, 128GB RAM machine is set as the control plane of the cluster, while an 8-Core / 16-Thread 64GB RAM machine is set as a worker node.
- An additional server, with similar processing capabilities as the worker node mentioned above, is used for the deployment of the event streaming platform (Apache Kafka) and the Blockchain component of the EGTN platform.

The servers' infrastructure is considered appropriate for the mostly RAM-heavy, I/O bound workloads that it needs to serve. The latest (at the time of procurement) version of the Ubuntu Long Term Support (LTS) server image (20.04) was used as the operating system of choice. The systems were configured and the necessary packages, such as docker, kubeadm and kubectl, were installed. The cluster was bootstrapped using kubeadm as per the official kubernetes instructions. The control plane node was configured first, while the worker node was later added to the cluster through a kubeadm join command.

Having prepared the cluster, the main components could then be deployed. In the following, the storage services provided by the platform will be presented, the ingestion methods and tools used, the analytics solution and a general overview of the integration process of a PI service.

[4] Apache ZooKeeper, https://zookeeper.apache.org/
[5] Grafana, https://grafana.com/
[6] Apache Airflow, https://airflow.apache.org/
[7] Hetzner Online GmbH, https://www.hetzner.com/

3.3.1 Data Lake

The EGTN Platform Data Lake consists of the event streaming platform whose role is to ingest real time data and the databases that serve different use cases of structured data. Specifically, the event streaming is handled by Apache Kafka, while a MongoDB and an InfluxDB deployments are provided for document-based and timeseries data respectively.

3.3.2 Data Aggregator

To accommodate the variety of data being ingested into the EGTN platform, a Data Aggregator utility has been developed. It is a Python based, containerized command line application that can retrieve data from sources such as an EPCIS server, CSV files, XML documents and others and load them into a Kafka topic, https://gitlab.com/planet-h2020/egtn-data-aggregator. Most importantly, it addresses the requirements A.2 and A.3, where it validates the incoming data based on a collection of metadata stored in the MongoDB. The collection includes metadata i.e., short description, owner, uploader, sensitivity, tags and volumetric information, for each Kafka topic, collection in MongoDB and measurement in InfluxDB.

The Data Aggregator utility has been executed on demand, but it is designed so that it can be periodically executed in the cluster as a Kubernetes CronJob. It currently serves the ingestion of the LL3 events data from the ILIM EPCIS server, the LL1 DHL data provided as CSV files, the CityLogin data also provided as CSV files and the Unified Transport Documents provided by COSCO as XML files.

3.3.3 Analytics

The EGTN platform offers Apache Zeppelin notebooks so that developers can implement and test their solutions directly on the platform. They are used for data exploration and processing and provide an interface to the Apache Spark cluster-computing to the interested actors for big data analytics. The notebooks have access to the EGTN Data Lake either through the Kafka topics or from the filesystem itself with volume mounts on the notebook pods. Models are deployed on the platform as separate services in the Kubernetes cluster that consume data from the available data sources and continuously calculate valuable analytics. As previously mentioned, multiple solutions are offered by the platform for the storage of the generated analytics, which are stored following naming conventions that contain information regarding the concerned LL, the data source, and the data consumer.

From a technology perspective, the modellers develop their AI models in R and Python programming languages through the Zeppelin notebooks and use git to track changes in their code.

3.3.4 Workflow Management

To create, execute, schedule and monitor tasks Apache Airflow has been deployed to the EGTN Platform. It consists of a mature platform to programmatically author, schedule and monitor workflows and is used extensively in the industry for ETL processes.

Airflow comes pre-loaded with features that provide dynamicity and extensibility to the EGTN platform. It uses operators that define templates for tasks that facilitate the connection to the ETGN platform's services such as databases or APIs. Containers can also be executed through Airflow operators further expanding the flexibility offered by the platform. Moreover, it integrates with Kubernetes, creating executor pods for each scheduled task thus taking advantage of the cluster resources and autoscaling features.

As a result, a user can perform data exploration and testing in the previously mentioned Zeppelin notebooks, before implementing an Airflow Directed Acyclic Graph (DAG) that subsequently can be scheduled to be executed periodically. And finally a UI is available for monitoring execution pipelines.

3.4 Modelling and Simulation

PLANET's modelling and simulation capabilities are constituted by different models which have been developed in accordance with simulation requirements (presented in D2.1) and used to evaluate strategic/macro & PI paradigm operational scenarios in the PLANET project. The models estimate transport flows from different perspectives: micro models focus on more operational aspects, assessing the impact of innovations on small-scale processes, while macro models, with a broad scope, focus on the major streams. A detailed description of the models can be found in deliverable D1.3 Modelling & Simulation Capability.

In the context of this project, data harmonization has been employed to bring together the information requirements of the different models used for simulation and analysis. A harmonized dataset enables the combination of information provided by the various use cases and facilitates the usage of the different models. A detailed description of the data requirements for modelling and simulation can also be found in deliverable D1.3.

As these models need to accurately represent the transport processes in the corridors and reflect specific Living Labs requirements, it is necessary for the EGTN platform to have information regarding demand, infrastructure, and available services so the different models can interact with the platform either to exchange information (input and output data) or interact with other models or services.

In the context of LL1, some examples of the interaction of the models with the platform are described in the following.

As shown in Figure 2, the PI Network Simulator in the UC1 of LL1 is used to evaluate how the impact of PI concepts in combination with existing technologies applied in novel ways for T&L (IoT, AI, BC) can improve the processes, operations, and efficiency of transport chains between China and the EU. A description of the interaction between the simulation model and the ETGN platform, as shown in Figure 2, is:

1. The simulation model needs information regarding the containerized cargo demand, and it takes COSCO container data from the MongoDB.
2. The simulation model sends the container data to the PI Port Choice model to get the best container-vessel allocation and determine whether each of the vessels should omit or call at discharge ports.
3. After running the PI Network Simulator, the simulation results are sent to the predictive transport models to make predictions based on transport cost and other operational KPIs.
4. The simulation results are also shown in the dashboard, where the users can explore the impact that PI and new technologies have on different transport processes based on what-if scenarios and key performance indicators.

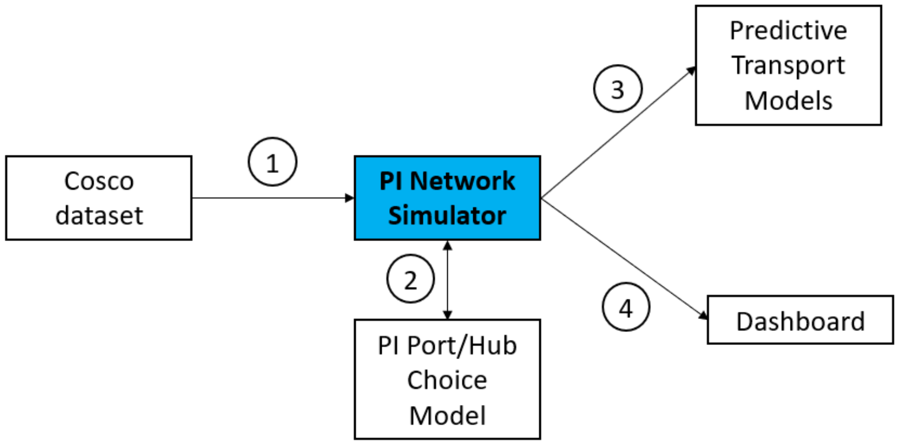

Figure 2: PI Network Simulator interaction with the EGTN Platform

In UC2, see Figure 3, the Last Mile Delivery Model is used to assess how Physical Internet concepts in combination with algorithms for last mile collaboration and the utilization of green vehicles can improve processes, operations, and efficiency of urban logistics. A description of the interaction between the simulation model and the ETGN platform, as shown in Figure 3, is:

1. The simulation model needs information regarding the parcel demand, and it takes CityLogin deliveries data from the MongoDB.
2. The simulation model sends the data to the last mile parcel reshuffling model to get the optimal parcel-vehicle on-the-fly re-allocation during routes.
3. After running the Last Mile Delivery Model, the simulation results are shown in the dashboard, where the users can explore the impact that PI, algorithms for collaboration and green vehicles have on last mile distribution based on what-if scenarios and key performance indicators.

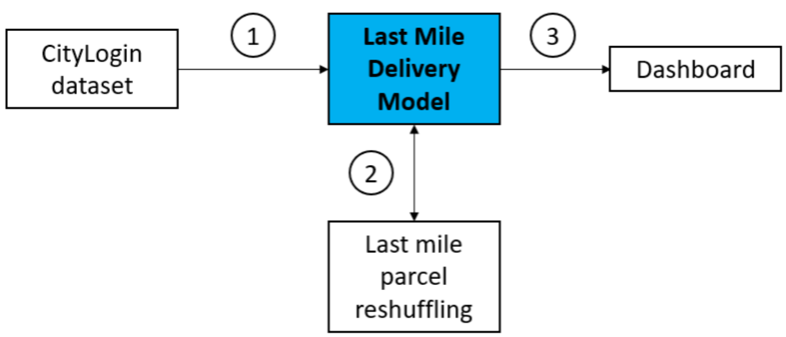

Figure 3: Last Mile Delivery Model interaction with the EGTN Platform

Furthermore, in UC2, as shown in Figure 4, the PI Warehouse Model is used to assess how operations and processes within a warehouse are improved when PI concepts in combination with new technologies such as AI are applied. A description of the interaction between the simulation model and the ETGN platform, as shown in Figure 4, is:

1. The simulation model needs information regarding the palletized demand and the warehouse infrastructure (layout, resources...), and it takes DHL data from the MongoDB.
2. The simulation model interacts with the demand/volume forecast service to get the demand prediction for the considered scenarios.

3. After running the PI Warehouse Model, the simulation results are shown in the dashboard, where the users can explore the impact that PI and AI have on warehouse processes based on what-if scenarios and key performance indicators.

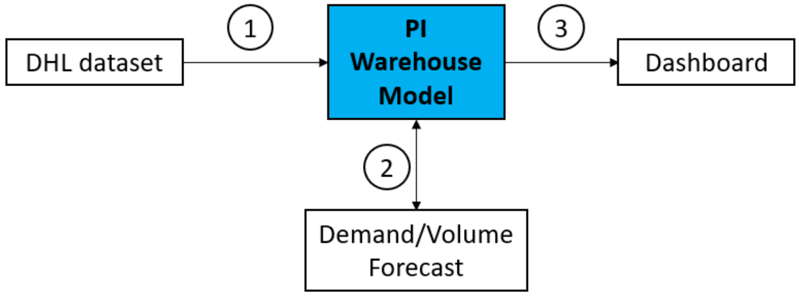

Figure 4: PI Warehouse Model interaction with the EGTN Platform

Thus, from a modelling and simulation point of view, the EGTN platform acts as a tool for the interoperability of models and services, as well as a common place to take input data from data sources in a harmonized way.

3.5 Governance

Platform Governance can be defined as who makes what decisions about a platform. In that regard, a critical challenge is that a platform owner must retain sufficient control to ensure the integrity of the platform while giving away enough control to encourage innovation by the platform's module developers [2]. The EGTN Platform was developed in such a way that it guarantees the security and integrity of the platform to the owner and at the same time enables the seamless integration of logistics services based on certain governance principles for onboarding of users and data.

3.5.1 Data

Regarding the ingestion of data, the Data Aggregator is the entry point for new data, either real time or batch data. The Data Aggregator utility, see section 3.3.2, uses the metadata already stored in the platform to validate the incoming data based on fields such as owner, uploader, sensitivity, related tags and volumetric information. This is part of the automated pre-processing happening on the platform through the Airflow service, which provides data ready for analysis. The pre-processed data are then stored in the corresponding database of the EGTN Data Lake together with logs and reports of the process.

The Event Streaming Service provides the configuration option for adjusting the data retention policy per Kafka topic, which means that data owners can define when their data should be deleted from the Data Lake of the EGTN Platform. Finally, as further described in section 6, the minimisation of data stored in the platform is a GDPR principle [3] that is applied horizontally on the data shared through the platform, from the IoT data to blockchain events. In the case of blockchain events, only pointers and metadata of a transport document included in an event are stored in the EGTN Platform, while the actual documents are stored in external legacy systems - owned and controlled by T&L stakeholders - retrieved by the platform and shown to a user only upon his/her request.

3.5.2 Services

The EGTN platform follows a cloud-native approach that can be applied on either public, private or a hybrid infrastructure. It encourages the usage of containers by the service provider, which are then deployed to the cluster through a Kubernetes manifest or a Helm chart. These microservices are then formed into a loosely coupled system that is resilient, manageable, and observable.

To ensure reproducibility of the platform, all microservices installations have been automated and scripted. These scripts are kept version controlled and belong to a Continuous Integration and Continuous Deployment pipeline (CI/CD).

Users and developers can create tickets on Slack to notify the development team about requests and issues during integration. Depending on the load of tickets, the requested changes are handled primarily based on priority or alternatively in order of ticket creation. If many change requests occur simultaneously, the team performs sprints to ensure that all changes maintain compatibility. This reduces the interoperability challenge down the line. Requests are prioritised based on (but not limited to) the following orders of factors: security (system or data), stability, restrictions in existing functionality, support for new data, support for new functionalities. D2.1

3.5.3 Users

The EGTN Platform is built upon trust with all users, which is based on predefined principles for onboarding anyone that needs to interact with the platform. The strategic decision of limiting the exposed interfaces i.e., data ingestion through standardised tools, system administration and monitoring through Grafana, data processing and analytics through Zeppelin, blockchain events through dedicated connectors and interaction with the PI Services through the EGTN Dashboard, provides a model that enables the explicit

Data providers, data analysts, data modelers, system administrators and T&L stakeholders request registration and enrolment from the EGTN Platform administrator, who creates accounts and enforces strong password restrictions. All the aforementioned interfaces are strictly secured, and user accounts are given on-demand only. Also, there are mechanisms to allow single sign-on on the different interfaces to reduce user confusion. Personal Identification Information (PII) are not stored within the EGTN Platform's databases, while any data stored is encrypted using strong cryptographic algorithms (AES-256) to enhance the accountability of the platform. This gives users full control over their data even through there is a central governance framework.

3.6 Deployment

The EGTN components are containerised and are deployed using two different perspectives i.e., the more infrastructural utilities e.g., Kafka and the Blockchain Interoperability service, follow a different approach than the PI Services. Firstly, the Apache Kafka event streaming platform has been deployed on one server using docker compose stacks. The event streaming platform uses Apache Kafka 2.8.0 and is based on the Confluent 6.2.6 distribution. The Blockchain Interoperability service is separately deployed as a Python container. The server's stacks and services are monitored through a Portainer instance. Separating the above services further enhances the exclusive access and control of these services by the EGTN platform administrators.

Secondly, the Kubernetes cluster is used for the deployment of the storage and ingestion services of the EGTN platform, together with the PI services developed within the PLANET project. At the storage layer of the EGTN platform, as previously mentioned, a document-based and a time-series database have been deployed. The MongoDB database has been deployed using a set of Kubernetes manifests declaring the pod, service, volume, and secrets configuration. The InfluxDB database is deployed using the Helm chart from the official InfluxData repository. An instance of a Telegraf service handles the ingestion of data to the time-series database. It is also deployed using a Helm chart from the official InfluxData repository.

Furthermore, to enable and facilitate the EGTN users to develop their models using the platform's infrastructure, a Zeppelin notebook server has been deployed using Kubernetes YAML manifests. The Apache Airflow platform, deployed through its official Helm chart, further enables the EGTN users to execute workflows in the platform.

Finally, a Grafana visualization web application, deployed through a Helm chart, is used to provide monitoring dashboards.

3.7 Value for the EGTN PI Services

The EGTN Infrastructure provides great value for the development of data-driven T&L collaborative logistics services and new eCommerce models that bring us a step closer to the PI paradigm. All the infrastructure software is open source enabling an open and neutral industry platform and facilitating the engagement of smaller T&L players through the low-cost integration process. At the same time, it provides flexibility and applicability to different Cloud models, such as Public, Private or Hybrid, through its container-based architecture.

The open APIs offered by the architecture of the platform for data ingestion increase the potential for a modern Data Lake that optimises resources and data flows, improves performance and enables big data analytics on shareable data. On the other hand, the integration of the PI Services is managed by well-defined and automated processes that ensure seamless integration and visualisation through the Dashboard. As explained in 3.5, the platform governance model enables a secure and private infrastructure fulfilling the privacy criteria of the business stakeholders for sharing anonymised data that are useful for the development of PI Services.

4 EGTN infrastructure and Services

The EGTN Platform provides the infrastructure, namely a secure and scalable Data Lake, software container orchestration and analytics notebooks for code development that enables the fundamentals for the development of data-driven PI services. Figure 5 shows the integrated technology architecture of the EGTN Platform that includes connectivity components for ingestion of various datasets, spanning real-time data, batch data and blockchain events, a Data Lake for data storage, analytics tools and the EGTN PI Services developed by the WP2 tasks.

Each of the components shown in Figure 5 are further listed and described in the following subsections.

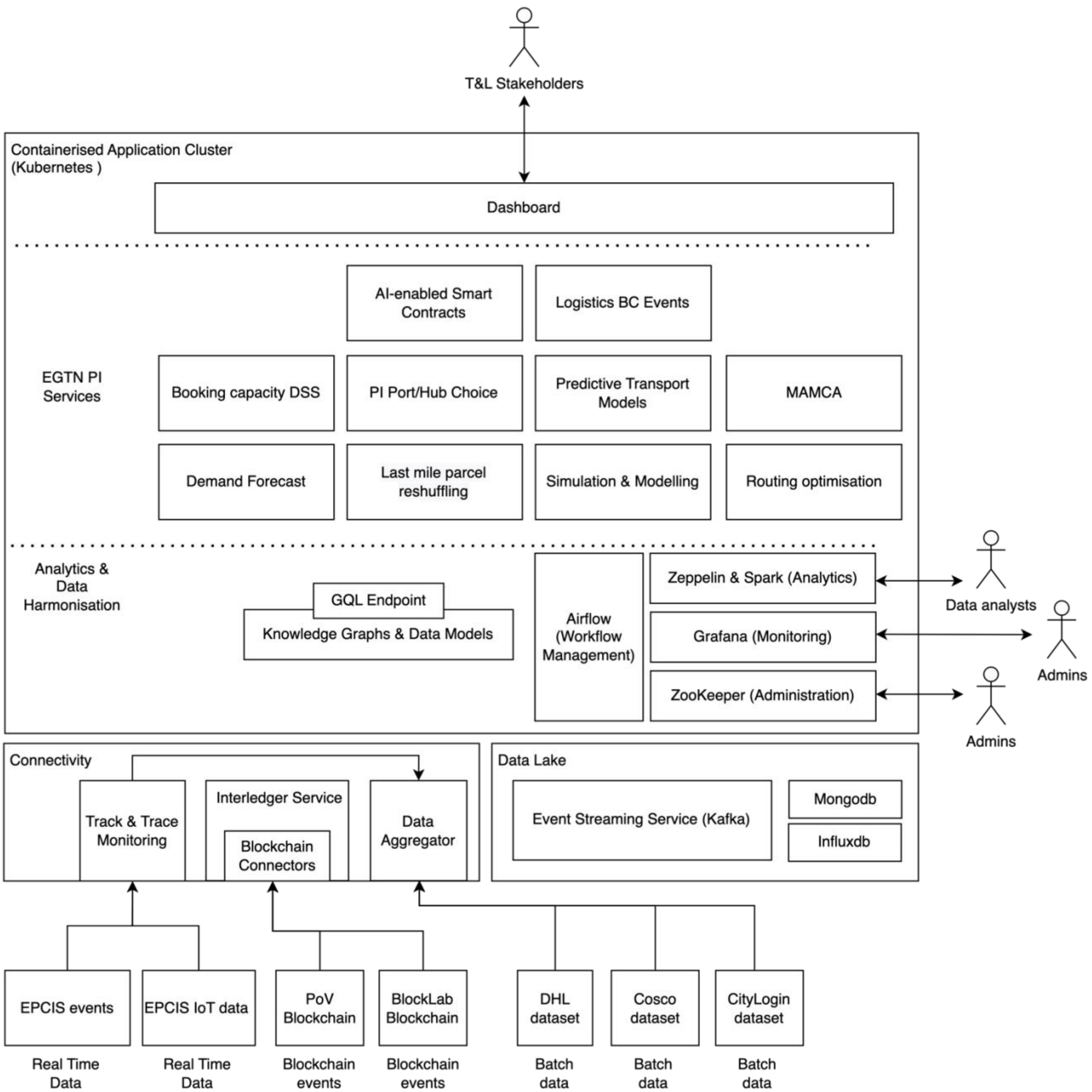

Figure 5: The EGTN Platform - infrastructure and PI Services

4.1 Infrastructure

The components listed in Table 8 through Table 13 were developed in the context of T2.1, they support the deployment of PI Logistics services, listed in section 4.2, and provide an open, secure and scalable infrastructure for integration of both data and services. The tables include a short description for each service together with the stakeholders involved.

Data Lake

Task Mapping	T2.1
Description/Added Value	The EGTN Data Lake comprises of an Apache Kafka service running in the EGTN platform that enables data providers to push their data to the platform, a MongoDB instance for structured data and an InfluxDB for timeseries data.
Stakeholders	Data providers and Developers/Engineers to perform post-processing of data

Table 8: Data Lake

Data Ingestion and Processing

Task Mapping	T2.1
Description/Added Value	The Data Aggregator of the EGTN Platform is used for the ingestion of data from multiple sources and manages the heterogeneity of data types and formats as well as their validation, see section 3.3.2. Finally, Apache Airflow enables the platform users to further ingest, process and store data through user-defined workflows.
Stakeholders	Data providers and EGTN admins

Table 9: Data Ingestion and Processing

Administration and Configuration (ZooKeeper)

Task Mapping	T2.1
Description/Added Value	It enables the efficient configuration and monitoring of the Kafka cluster.
Stakeholders	EGTN admins

Table 10: Administration and Configuration

Analytics Notebook (Apache Zeppelin and Spark)

Task Mapping	T2.1
Description/Added Value	Web-based notebook that enables interactive data analytics and collaborative documents with SQL, Scala, Python and R. It also enables integration with Spark for large-scale data processing. Apache Spark provides an interface for programming entire clusters with implicit data parallelism and fault tolerance.

Stakeholders	Developers and data analysts who connect to the Zeppelin tool to write code for data analytics.

Table 11: Analytics Notebook

Container Orchestration (Kubernetes)	
Task Mapping	T2.1
Description/Added Value	Open-source container-orchestration system for automating computer application deployment, scaling, and management. It brings efficiency and automation in application and service deployment.
Stakeholders	Logistics service providers to deploy their containerised services in the EGTN Platform

Table 12: Container Orchestration

Monitoring and Visualisation (Grafana)	
Task Mapping	T2.1
Description/Added Value	It is a multi-platform open-source analytics and interactive visualization web application that enables monitoring of the EGTN infrastructure. It provides charts, graphs, and alerts for the web when connected to supported data sources.
Stakeholders	EGTN admins, who monitor the Kafka cluster and developers who visualise their data analysis on Zeppelin.

Table 13: Monitoring and Visualisation

4.2 Logistics Services for Multimodal Freight Transport

This section briefly describes the logistics services deployed in the infrastructure of the EGTN Platform, while further details for each one of them can be found in the respective WP2 deliverables. For each service, we present the task in which it was implemented, a short description, the data input and data output of the service.

4.2.1 Track & Trace Monitoring

Track & Trace Monitoring	
Task Mapping	T2.2
Description/Added Value	It enables the real-time visibility of the shipment by proving detailed information, such as position, temperature, luminance, humidity, acceleration.
Data Input	N/A
Data Output	In a Kafka topic: readings from IoT sensors in smart pallets/containers and events following the EPCIS format

Table 14: Track & Trace Monitoring EGTN Service

4.2.2 Metadata & Statistics

Metadata & Statistics

Task Mapping	T2.2
Description/Added Value	It generates statistics and additional information regarding a shipment.
Data Input	It gets as input the sensor readings from the Track & Trace Monitoring (4.2.1).
Data Output	Shipment summary (pdf that summarises all the events in the shipment)

Table 15: Metadata & Statistics EGTN Service

4.2.3 Events Generation

Events Generation

Task Mapping	T2.2
Description/Added Value	It generates notifications for events based on incidents during the shipment.
Data Input	It gets as input the sensor readings from the Track & Trace Monitoring (4.2.1).
Data Output	Arriving to/leaving from a certain point (point-of-interest) in EPCIS format.

Table 16: Events Generation EGTN Service

4.2.4 Consolidation & Deconsolidation

Consolidation & Deconsolidation

Task Mapping	T2.2
Description/Added Value	It allows to bind/unbind a smart BLE device or logistic asset to a shipment.
Data Input	Mobile App and Web Interface (outside of EGTN)
Data Output	Shipment Events (time when consolidation happens)

Table 17: Consolidation & Deconsolidation EGTN Service

4.2.5 KG Query: Weather

KG Query: Weather

Task Mapping	T2.2

Description/Added Value	It makes available through a KG API, historical weather data (and possibly historical forecasts should they be of interest) near Points of Interest such as train stations, border crossing, ports and warehouses.
Data Input	External public source
Data Output	In a GraphQL endpoint: time-series of sensor reading values. Each data point includes time, value and observation type. Data can be obtained over specific time periods and for various collections of platforms, sensors and observations as well as post-processed in the GQL query.

Table 18: KG Query: Weather EGTN Service

4.2.6 KG Query: IoT Readings

KG Query: IoT Readings

Task Mapping	T2.2
Description/Added Value	It makes available historical time-series data from EGTN IoT sensors.
Data Input	It consumes the output of the Track & Trace Monitoring service (4.2.1) through a Kafka topic.
Data Output	In a GraphQL endpoint: time-series of sensors reading values. Each data point includes time, value and observation type. Data can be obtained over specific time periods and for various collections of platforms, sensors and observations as well as post-processed in the GQL query.

Table 19: KG Query: IoT Readings EGTN Service

4.2.7 KG Query: IoT Events

KG Query: IoT Events

Task Mapping	T2.2
Description/Added Value	It makes available historical events related to EGTN IoT sensors such as sensor installation, creation of SSCC, arrival at PoI etc. It is useful primarily as a step towards tracking specific packages or modes of transport.
Data Input	It consumes the output of the Events Generation service (4.2.3) through a Kafka topic.
Data Output	In a GraphQL endpoint: EPCIS events

Table 20: KG Query: IoT Events EGTN Service

4.2.8 KG Query: Route Data

KG Query: Route Data

Task Mapping	T2.2
Description/Added Value	It makes available information about the LL3 train route from China to Poland, such as segments, stations, train compositions, associated weather stations and so on.
Data Input	A mix of real and simulated data based on feedback from LL3 partners.
Data Output	In a GraphQL endpoint: details on route segments, stations, train compositions, train engine specifications, associated weather stations and so on

Table 21: KG Query: Route Data EGTN Service

4.2.9 Demand Forecast

Demand Forecast

Task Mapping	T2.3
Description/Added Value	It forecasts the number of pallets/containers arriving in a warehouse or port, providing confidence intervals for the next 10 days. The Demand forecast service reduces hiring costs, as it enables the booking of vehicles in advance to avoid demand peaks and high prices. The output is consumed by the 4.2.15 Booking Capacity DSS service that produces recommendations for generating smart contracts based on the predictions of the Demand Forecast service.
Data Input	Historical data regarding the inflow of pallets/containers, found in the EGTN Data Lake in various formats i.e., messages in MongoDB and/or excel files.
Data Output	In MongoDB: [time stamp] [Number of pallets/containers] max/min confidence levels

Table 22: Demand Forecast EGTN Service

4.2.10 Routing Optimisation

Routing Optimisation

Task Mapping	T2.3
Description/Added Value	This service addresses some of the most challenging aspects in delivering freight volume within the last mile delivery (urban area). A core aspect of the service is that it scales the optimisation within a big number of delivery nodes, in the order of thousands per day. It prioritises the use of hybrid and electric vehicles to comply with carbon emission norms within the Madrid urban area.
Data Input	Historical data from MongoDB (CityLogin dataset) or real-time data (longitude, latitude, distance, time, traffic intensity and density, weather along route, holiday data) from the Event Streaming Service, namely Kafka.
Data Output	In MongoDB: [Faster/better route]

Table 23: Routing Optimisation EGTN Service

4.2.11 Predictive Transport Models

Predictive Transport Models

Task Mapping	T2.3
Description/Added Value	This service can make use of simulation data and machine learning-based predictive models to predict changes in freight volume across corridors. Based on the data provided, the service can generate correlation graphs between the predicted changes in volume and historical information on relevant variables such as transportation cost and lead time.
Data Input	Historical data from MongoDB, ITAINNOVA dataset from simulation tools. Data variables include the freight volume transported across different corridors and also, the lead time, carbon emissions and costs (in euro) per corridor.
Data Output	Forecasted freight volume arriving to the city of Madrid (or other delivery city destination depending on data input). Data vectors build correlation plots between predicted freight volume and historical information on cost, lead time and carbon emissions.

Table 24: Predictive Transport Models

4.2.12 PI Port/Hub Choice

PI Port/Hub Choice

Task Mapping	T2.4
Description/Added Value	The PI Port/Hub Choice Service optimises terminals to be visited along a route, considering hinterland transport options and terminal congestion. Given a set of containers (and their final destinations), some candidate ports (and delays that are input by the user) and an Origin Destination (OD) matrix for one or more modes, it determines which subset of ports should the vessel visit.
Data Input	Historical data from MongoDB (Cosco dataset)
Data Output	In MongoDB: discharge port/hub per container & vessel ports to be called

Table 25: PI Port/Hub Choice

4.2.13 Last Mile Parcel Reshuffling

Last Mile Parcel Reshuffling

Task Mapping	T2.4
Description/Added Value	The Last Mile Parcel Reshuffling Service optimises parcel reshuffling for last mile delivery, involving collaborative opportunities between vans to expedite deliveries. It determines if a van operating in proximity can be of meaningful assistance to a van running late. If yes, it redistributes the parcels, sets a meeting point, and redesigns the vehicle route.

Data Input	Current location of vans and pending deliveries
Data Output	In MongoDB: updated routes for two vehicles & parcels to be moved from the late van to the second vehicle

Table 26: Last Mile Parcel Reshuffling

4.2.14 MAMCA

MAMCA

Task Mapping	T2.4
Description/Added Value	The MAMCA Service introduces a stakeholder perspective considering individual stakeholder criteria and weights for a comprehensive characterisation of each last mile operator. Each operator can pre-define acceptable performance criteria for collaboration.
Data Input	Stakeholders analysis through workshops
Data Output	In MongoDB: Table of weights in json format

Table 27: MAMCA EGTN Service

4.2.15 Booking Capacity DSS

Booking Capacity DSS

Task Mapping	T2.4
Description/Added Value	The Booking Caparcity DSS Service determines the number of trucks that a warehouse needs to book in a time range of ten or three days ahead. It is based on time series predictions (from the Demand Forecast Service) and on a dynamic pricing strategy i.e. booking and cancellation fees. The service determines the trucking capacity that needs to be booked by the warehouse towards minimising costs.
Data Input	It receives predictions and confidence intervals from the Demand Forecast Service (4.2.9)
Data Output	In MongoDB: [quantity of trucks to be booked for 10 and 3 days ahead], which is consumed by the AI-enabled Smart Contracts service (4.2.17).

Table 28: Booking Capacity DSS EGTN Service

4.2.16 Logistics Events

Logistics Events

Task Mapping	T2.5
Description/Added Value	It increases the visibility of the supply chain by provisioning blockchain events from heterogeneous logistic stakeholders. The service implements a universal front-end to

	existing backend blockchain systems transparently and at the same time securely exposing logistics events i.e., container arrival at port, container pickup, container unload etc.
Data Input	It listens for blockchain events from backend blockchain networks through a Kafka topic. The service also consumes IoT data from sensors tracking smart pallets/trucks through the Track & Trace Service (4.2.1).
Data Output	In Kafka: trusted meta-events (e.g. enhanced-trust departure/delivery) and SLA monitoring events (e.g. contract fulfilled/violated) In MongoDB: metadata for EGTN Smart Contracts managing SLAs

Table 29: Logistics Events EGTN Service

4.2.17 AI-enabled Smart Contracts

AI-enabled Smart Contracts

Task Mapping	T2.5
Description/Added Value	It enables the automatic generation of smart contracts for warehouse management, embedding special conditions and violation measures based on risk assessment and a dynamic pricing policy. The smart contracts are generated between different blockchain communities e.g. between FF and Carrier communities and they are based on predefined contracts that increase response time significantly.
Data Input	It receives the quantity of trucks to be booked for 10 and 3 days ahead, from the Booking Capacity DSS service (4.2.15).
Data Output	In MongoDB: contract Id + contract metadata

Table 30: AI-enabled Smart Contracts EGTN Service

5 EGTN Data Lake

5.1.1 Event Streaming Service

An introduction to the Apache Kafka event streaming platform used by the EGTN has already been discussed in D2.1 but a brief introduction and overview is going to be presented in the following.

The Apache Kafka storage model is based on topics. Topics are the way Kafka organizes its data and, in the context of the PLANET project, contain information of a specific context. Topics can have multiple producers and consumers although more often there is a single producer and multiple consumers. Furthermore, the data retention policy of the topics can be adjusted; this has been set to a large value of 2 years according to the living labs requirements. All topics are partitioned between the three brokers of the EGTN platform. Finally, topics offer the necessary access control as discussed later in section 6.1.

The ingestion of a new dataset was based on a few criteria. Firstly, the nature of the data; real time data are more suited for the publish-subscribe model that the event streaming platform offers, while more static data are sent to a database. The decision for the type of storage to be used is not rigid; it is taken primarily based on the needs of and in agreement with the partners involved in the production and processing of the data.

In Table 31 through Table 37, the topics used for the EGTN Platform are presented with information regarding the producer, consumers, and data structure.

CityLogin Deliveries Data Description

Topic Name	citylogin-deliveries
Description	Dataset of deliveries performed by CityLogin. Shared as an Excel file. Ingested using the Data Aggregator.
Data Input	XLSX file with header.
Topic Event	`{ originWarehouse: <str>, activity: <str>, serviceType: <str>, taskID: <int>, date: <str>, route: <str>, sequence: <str>, start: <str>, end: <str>, clientID: <int>, country: <str>, latitude: <float>, longitude: <float>, serviceTime: <int>, ETA: <str>, acticles: <int>, weight: <float>, volume: <float> }`

Table 31: CityLogin Deliveries Data Description

DHL Metrics Data Description

Topic Name	dhl-metrics
Description	Dataset of DHL orders. Shared as an Excel file. Ingested using the Data Aggregator.
Data Input	XLSX file with header.
Topic Event	`{ DATE: <datetime>, CUSTOMER: <str>, PROCESS: <str>,`

	SUBPROCESS: <str>, QUANTITY: <int> }

Table 32: DHL Metrics Data Description

IoT EPCIS v2 Data Description

Topic Name	ll1-iot
Description	Data coming from sensors deployed in LL1.
Data Input	EPCIS v2 documents published by the sensors.
Topic Event	The retrieved objects are structured as the JSON-based EPCISDocument schema version 2.0. https://github.com/gs1/EPCIS/blob/master/EPCIS-JSON-Schema.json

Table 33: IoT EPCIS v2 Data Description

Events EPCIS v1 Data Description

Topic Name	ll3-events
Description	EPCIS v1 events retrieved from ILIM EPCIS server with the Data Aggregator.
Data Input	EPCIS v1 database.
Topic Event	The retrieved data are structured as the XML-based EPCISDocument schema version 1.0.

Table 34: Events EPCIS v1 Data Description

Blockchain Events Data Description

Topic Name	bc-events
Description	Blockchain events coming from the EGTN Interledger service and backend Blockchain systems
Data Input	Blockchain events generated either within the EGTN Interledger service or in backend Blockchain systems and ingested through the connectors of the Interledger service.
Topic Event	{ "Document_ID": <str>, "Document_type": <str>, "Document_hash": <str>, "Transport_Order_reference_number": <str>, "City_of_loading": <str>, "Date_of_loading": <str>, "Time_of_loading": <str>, "City_of_discharge": <str>, "Date_of_discharge": <str>, "Time_of_discharge": <str>, "Type_of_package": <str>, "Number_of_packages": <str>, "Weight": <str>, "Volume": <str>, "Issue_date": <str>, "Ledger_name": <str>, "TSN_ID": <str>, "Event": <str> }

Table 35: Blockchain Events Data Description

Weather Data Description

Topic Name	weather-ingestion

Description	Weather data retrieved from open sources.
Data Input	Weather ingestion SIRMA Cron Job.

Topic Event	`{`
	`"station": <str>,`
	`"type": <str>,`
	`"observations": [`
	`{`
	`"date": <ISODate>,`
	`"measurements":[`
	`{`
	`"property": <str>,`
	`"component": <str>,`
	`"value": <str>`
	`},`
	`...`
	`}`
	`...`
	`}`

Table 36: Weather Data Description

SIRMA Timeseries Data Description

Topic Name	timeseries-ingest
Description	Timeseries data ingested by SIRMA platform.
Data Input	Timeseries ingest SIRMA service.

Topic Event	`InfluxDB Line Protocol Data. Sample schema:`
	`Measurement=iot,`
	`TagsSet=[platformID],`
	`FieldSet=[`
	`https://gs1.org/voc/Temperature,`
	`https://gs1.org/voc/RelativeHumidity,`
	`https://gs1.org/voc/Luminance,`
	`https://ngs-sensors.it/Battery,`
	`https://gs1.org/voc/Acceleration/Comp-x,`
	`https://gs1.org/voc/Acceleration/Comp-y,`
	`https://gs1.org/voc/Acceleration/Comp-z,`
	`https://gs1.org/voc/Latitude/Comp-latitude,`
	`https://gs1.org/voc/Longitude/Comp-longitude,`
	`],`

Table 37: SIRMA Timeseries Data Description

5.1.2 Database

The container-based cloud-native approach of the EGTN platform enables the configuration of the provided storage services to be defined dynamically based on the requirements of the partners. From the initial phases of the PLANET project a document-based database has been deployed to store any data required by the partners' services and APIs in a flexible manner. Furthermore, a timeseries database is employed to store time series data.

The InfluxDB timeseries database has been directly used in combination with a Telegraf operator to fulfil the monitoring requirements of the platform. Different buckets could then be created based on the desired access rights. Despite the existence of the timeseries database, no use case was defined among the partners for the exploitation of the resource.

The document-based database has been much more popular among partners. Most of the partners' services use in some way one or more collections from the MongoDB. The Data Aggregator has again been utilized to ingest data into the database. Separate databases have been created to separate the data sources and manage the access control. The convention used in the context of the project was that each data source was ingested separately in a collection in a database. Any partner that could read the initial raw data would then be able to create further collections in the same database to store metadata and any processed data. The convention was suggested and encouraged but not enforced, since dynamicity and flexibility were required and appreciated.

In the following an overview of the data stored in the MongoDB is presented.

Database: info

Stores general, non-private information. Accessible by everyone.

Collection	depots
Description	Documents with information on the depots.
Sample	`{ "depotId": "BCN17", "name": "CIMAT S.A.", "location": "ESBCN", "address": "Calle 100 s/n Barcelona 08820", "isDryPort": false }`
Collection	portCosts
Description	Documents with information on the costs of each port.
Sample	`{ "name": "Barcelona", "callCost": 100.0 }`
Collection	distances
Description	Documents describing the distances from one location to several others.
Sample	`{ "name" : "Barcelona", "destinations" : [ { "name" : "Zaragoza", "distance": 310.0 }, { "name" : "Madrid", "distance" : 630.0 }, … }`
Collection	terminals

Description	Documents with information on the terminals.

Sample	{
	"depotId" : "BCN03",
	"name": "TERMINAL PORT NOU S.A.",
	"location" : "ESBCN",
	"address": "Costa Puerto Barcelona 08039"
	}

Table 38: Database: info

Database: ingestion

Stores raw ingested data. Accessible by IBM, COSCO, Itainnova, eBOS and VLTN.

Collection	cityloginRealrounds
Description	Documents with information on CityLogin deliveries.

Sample	{
	"Activity": "classic",
	"Hub": "COMPANY HUB",
	"Sender": "COMPANY",
	"Name": "Aname",
	"Status": "COMPLETED",
	"Driver": "1",
	"Vehicle": "1",
	"Capacity Weight": 1200.0,
	"Capacity Volume": 3.0,
	"Orders": NumberInt(38),
	"Distance": 22.52,
	"Distance Unit": "km",
	"Weight": 76.65,
	"Volume": 0.76,
	"DateStarted" : ISODate("<>"),
	"DateFinished" : ISODate("<>"),
	"DurationSeconds" : NumberInt(26101),
	"Duration": "7h15m1s"
	}

Collection	dhlMetrics
Description	Documents with information on DHL metrics. A copy from the corresponding Kafka topic so that some services can use the data.

Sample	{
	"date": ISODate("2017-09-04T00:00:00.000+0000"),
	"customer": "COMPANY CUSTOMER",
	"process": "OUTBOUND",
	"subprocess": "PICKING LINES B2B",
	"quantity": NumberInt(2)
	}

Collection	edis

Description	Documents describing EDI documents.

Sample	{
	"syntax": "UNOA:2",
	"sender": "T073",
	"receiver": "VALENCIAPORT",
	"preparation_datetime": "210225:1551",
	"control_reference": "T1234567890123",
	"message_reference": "T1234567890123",
	"message_type": "COPRAR:D:00B:UN:SMDG20",
	"transport_instruction": "118:::COPLIS",
	"doc_id" : "T1234567890123",
	"msg_datetime" : "202102251551",
	"payload": BinData(0, "...bytes")
	}

Collection	inbound
Description	Documents with information on the Cosco inbound deliveries.

Sample	{
	"Container Number": "ABCD1234567",
	"Container Size Type": "20GP",
	"Cargo Nature Code" : "GC",
	"Estimated Berth Arrival": ISODate("<>"),
	"Container Last Discharge Voyage Code": "ABC1-ABC-123 W",
	"BL POL Port English Name": "Shanghai",
	"Container Last POD Port Code": "VLC",
	"Container Last Hub Code": "VLC01",
	"FND City Local Name": "Valencia, Valencia, Comunidad Valenciana, Spain",
	"Container Empty Return Location" : "ABC01",
	"Vessel Name": "CMA CGM BALI",
	"POL Country": "China"
	}

Collection	itainnovaContainers
Description	Data to support the simulations components.

Sample	{
	"scenario": "baseline",
	"cont_id": NumberInt(0),
	"cont_origin": "Shanghai",
	"cont_dischargeport": "Valencia",
	"cont_destination": "Madrid",
	"cont_transport": "rail",
	"cont_distance_km (hinterland)": 393.578044,
	"cont_leadtime_hours (hinterland)": 60.8537419729874,
	"cont_emissions_kgCO2 (hinterland)": 77.1412966239999,
	"cont_cost_eur (hinterland)": 588.52295012,
	"cont_timestamp": ISODate("<>")
	}

Collection	vlcToMad

Description	Cosco data with a similar format to the inbound collection but for inland routes.

Sample	{
	"Container Number": "ABCD1234567",
	"Container Size Type": "20GP",
	"Cargo Nature Code" : "GC",
	"Estimated Berth Arrival": ISODate("<>"),
	"Container Last Discharge Voyage Code": "ABC1-ABC-123 W",
	"BL POL Port English Name": "Shanghai",
	"Container Last POD Port Code": "VLC",
	"Container Last Hub Code": "VLC01",
	"FND City Local Name": "Madrid,Madrid, Spain ",
	"Container Empty Return Location": "ABC01"
	}

Table 39: Database: ingestion

Database: container

Stores processed data from the itainnovaContainers collection. Accessible by Itainnova and IBM.

Collection	container_train
Description	Documents with information from a model execution run for the container's dataset.

Sample	{
	"_id": "containers_MODEL1_test",
	"start_date": "2019-05-01",
	"end_date": "2020-09-01",
	"forecasting_horizon": NumberInt(100),
	"rep": NumberInt(10)
	}

Collection	YYYY-MM-DD--YYYY-MM-DD
Description	Documents generated by the processing of metadata described by the collection container_train. The name of the collection is defined from the start and end date.

Sample	{
	"forecasted_value_mean": 0.18773014843463898,
	"lower_bound": 0.10055843655049548,
	"upper_bound": 0.2749018603187825,
	"std": 0.11560431867837906,
	"actual_value": NumberInt(3),
	"date": ISODate("2020-09-02T00:00:00.000+0000")
	}

Table 40: Database: container

Database: pallet

Stores processed data from the dhlMetrics collection. Data concern pallet predictions from the Demand Forecast Service. Accessible by Itainnova and IBM.

Collection	pallet_train

Description	Documents with information of model execution runs for the dhlMetrics dataset.

Sample	{
	"_id": "pallet_MODEL1_test",
	"start_date": "2017-09-04",
	"end_date": "2019-06-06",
	"forecasting_horizon": NumberInt(10),
	"rep": NumberInt(10)
	}

Collection	YYYY-MM-DD--YYYY-MM-DD
Description	Documents generated by the processing of metadata described by the collection pallet_train. The name of the collection is defined from the start and end date.

Sample	{
	"forecasted_value_mean": 1538.3659888923169,
	"lower_bound": 1475.5519057365077,
	"upper_bound": 1601.180072048126,
	"std": 83.30201480744918,
	"actual_value": 1701.0,
	"time": ISODate("2019-06-07T00:00:00.000+0000")
	}

Table 41: Database: pallet

Database: last_mile_delivery_routing_jobs

Stores data of incoming jobs for routing. Accessible by CityLogin and IBM.

Collection	citylogin
Description	Documents with information of model execution runs for the cityloginRealrounds dataset.

Sample	{
	"_id": "City001382309282543001000033",
	"Unnamed: 0": NumberInt(0),
	"Activity": "classic",
	"Service_Type": "delivery",
	"Hub_origin": "lat, lon",
	"Task_ID": "City001382309282543001000033",
	"Date": ISODate("<>"),
	"End": "21:00",
	"Street": "Street Name",
	"Postcode": 28011.0,
	"City": "Madrid",
	"Country": "Spain",
	"Latitude": float-lat,
	"Longitude": float-lon,
	"weight_kg": NumberInt(1),
	"Volume_cm": 0.02
	}

Table 42: Database: last_mile_delivery_routing_jobs

Database: last_mile_delivery_routing_routes

Stores the results of routing jobs. Processes data from the cityloginRealrounds collection. Accessible by CityLogin and IBM.

Collection	citylogin
Description	Documents with information of model execution runs for the cityloginRealrounds dataset.

Sample

```
{
  "date": ISODate("2022-02-06T00:00:00.000+0000"),
  "remaining_driver_hours" : null,
  "optimization_target" : "total_time",
  "success" : true,
  "routes": [
    {
      "index": NumberInt(0),
      "_id": "City00135700275724600100034",
      "Unnamed: 0": 3533.0,
      "Activity": "classic",
      "Service_Type": "delivery",
      "Hub_origin": "lat, lon",
      "Task_ID": "City00135700275724600100034",
      "Date": ISODate("<>"),
      "End": "21:00",
      "Street": "Stree Name",
      "Postcode": 28722.0,
      "City": "El Vellón",
      "Country": "Spain",
      "Latitude": float-lat,
      "Longitude: float-lon,
      "weight_kg" : NumberInt(1),
      "Volume_cm": 0.02,
      "dest_string": "lat,lon",
      "quantity": NumberInt(1),
      "van_index": 0.0,
      "delivery_order": 0.0,
      "step_distance": 48591.3,
      "step_duration": 2519.6,
      "van_type": 0.0,
      "cluster": NumberInt(2089),
      "delivery_time": ISODate(""),
      "steps": "semicolon;separated;list;of;lat,lon"
    }...
  ],
  "message": "Route Found",
  "van_stats": [
    {
      "van_index" : NumberInt(0),
      "van_type" : NumberInt(0),
      "total_time_seconds" : NumberInt(9480),
      "total_distand_km" : 157602.8,
      "total_volume_cm" : 0.12,
      "total_quantity" : NumberInt(6),
      "total_weight" : 6.0
    },...
  ],
```

	"route_stats": {
	"total_distance": 6794628.199999999,
	"total_ev_distance_km": 2780277.9000000004,
	"total_time_seconds": NumberInt(1182600),
	"vans_used": NumberInt(54),
	"total_quantity": NumberInt(2618),
	"total_volume_cm": 52.36,
	"total_weight_kg": 2618.0
	},
	"updated": ISODate("2022-10-20T16:24:44.354+0000")
	}

Table 43: Database: last_mile_delivery_routing_routes

Database: PredictiveTP

Stores data and metadata from the Predictive Transport Models service. Accessible by IBM.

Collection	PredictiveTP_train
Description	Documents used by the Predictive Transport Models service.

Sample	{
	"_id": "PredictiveTP_MODEL1_test",
	"destination": "Madrid",
	"train_partition": NumberInt(80)
	}

Collection	container_prediction
Description	Documents used by the Predictive Transport Models service.

Sample	{
	"volume_no_containers": NumberInt(26),
	"predicted": 25.595989227294922,
	"time": ISODate("<>")
	}

Collection	correlation_vector_predicted_var
Description	Documents used by the Predictive Transport Models service.

Sample	{
	"volume_no_containers": NumberInt(185)
	}

Collection	correlation_vector_predictors_vars
Description	Documents used by the Predictive Transport Models service.

Sample	{
	"cont_cost_eur (hinterland)": 173406.2488971998,
	"cont_emissions_kgCO2 (hinterland)": 359237.684526369,
	"cont_leadtime_hours (hinterland)": 3751.3843080833212,
	"cont_distance_km (hinterland)": 114083.05848499999,
	"cont_transport_numeric": NumberInt(185)
	}

Table 44: Database: PredictiveTP

Database: Contracts
Stores data of . Accessible by .

Collection	agreements
Description	Documents with information on agreements between T&L communities, used by the EGTN Interledger service.

Sample	{
	"_id": "<>",
	"name": "Boots",
	"gsin": "<>",
	"destination": "Athens",
	"minHumidity": "10.0",
	"maxHumidity": "70.0",
	"minTemperature": "10.0",
	"maxTemperature": "20.0",
	"latestDeliveryDate": "12/2/2022",
	"status": "active"
	}

Table 45: Database: Contracts

5.2 Data Use Cases

The PI Logistic Services listed in section 4.2, serve a specific purpose and are driven by LLs requirements. On top of that, combinations of services following a specific data pipeline offer meta-services with additional value for the T&L community. The following subsections describe the end-to-end flow of indicative EGTN data starting from the data source, going through the EGTN infrastructure for processing, the EGTN Services and up to the EGTN Dashboard for visualisation.

5.2.1 Warehouse Management based on AI and Smart Contracts

The AI-enabled smart contracts meta-service, Figure 6, employs the Demand Forecast service for predicting incoming pallets in a warehouse as well as the Booking Capacity DSS for establishing a clear strategy for booking trucks and setting up a pricing policy for the agreements between different communities.

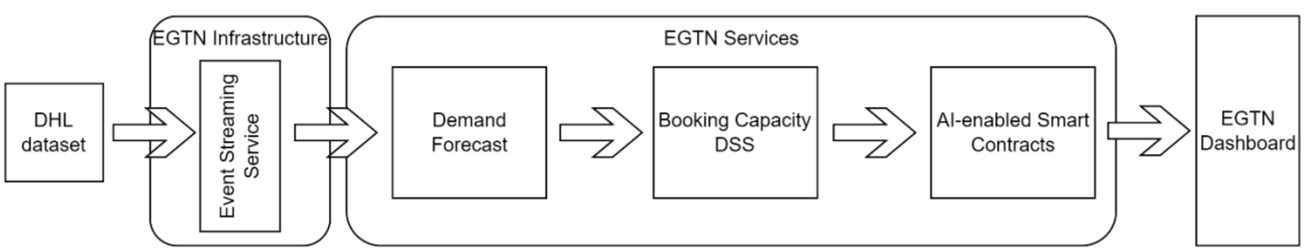

Figure 6: Warehouse Management based on AI and Smart Contracts

5.2.2 Logistics Events

The Logistics Events meta-service, Figure 7, employs the Interledger service for connecting to multiple backend blockchain systems and validates the integrity of incoming blockchain events based on IoT data and/or other trust anchors.

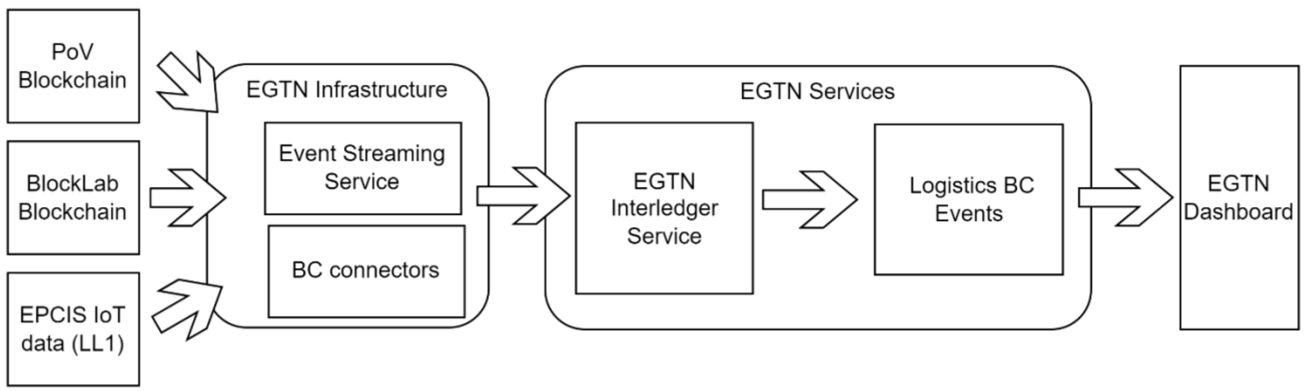

Figure 7: Sharing Logistic events through interoperable blockchains

5.2.3 Routing Optimisation

The Route Optimisation service, Figure 8, consumes either real time or historical data, processes them and calculates better routes in terms of carbon emissions and/or other factors.

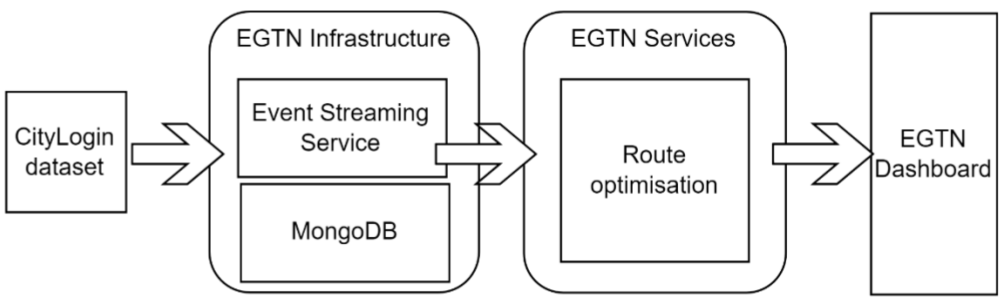

Figure 8: Route Optimisation in last mile delivery

5.2.4 Predictive Analytics based on Knowledge Graphs

The Predictive Analytics meta-service, Figure 9, uses Knowledge Graphs and through a GraphQL endpoint consumes IoT readings in an efficient and scalable way to process them and predict incoming pallets in a warehouse or containers in a port, which are then visualised in the Dashboard.

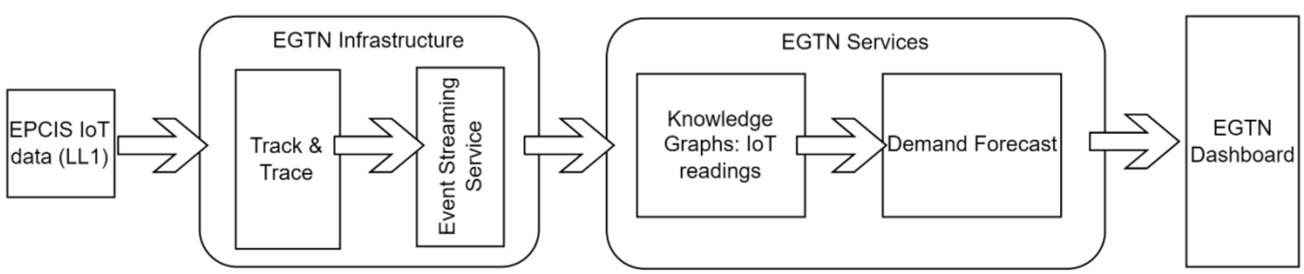

Figure 9: Sharing IoT measurements through Knowledge Graphs

6 Privacy and Security Considerations

The security and privacy of the architecture are of paramount importance for engaging T&L stakeholders in a trustful manner. To this end, the design of the architecture imposes certain principles that are followed by all the internal components to minimise the attack surface. The interfaces exposed publicly are limited, while accounts are created by invitation only and using single sign-on mechanisms to avoid users' confusion due to multiple passwords.

Towards complying to the most possible extent with GDPR, control of data is on the data owners/controllers' side by clearly stating to administrators the data governance model and data retention policies, providing even the possibility for data not to be stored in the platform but instead populating the topic of the Event Streaming service at run time. In addition, only anonymised, non-confidential data are shared through the Platform. Only pointers to the data residing in individual backend systems are stored in the EGTN infrastructure and consumed by the EGTN Services.

Regarding the ingested data, ACL protection is used in the Kafka topics, as explained in section 6.1, while users and services have permissioned access to the rest of the EGTN Data Lake components.

In terms of network security and encryption, it is of critical importance to secure the connection and transfer of sensitive information between the user and the platform, therefore all domains related to the EGTN Platform use X.509 certificates for the Transport Layer Security encryption and are accessible through the HTTPS protocol. Furthermore, the Apache Zeppelin notebooks interface is secured using Apache Shiro[8], which handles authentication, authorisation, session management and cryptography among others. Apache Shiro is an open-source software security framework developed by the Apache Software Foundation.

Finally, the server infrastructure provider, namely Hetzner, provides by default encryption in all its servers meaning that the infrastructure is deployed on a secure layer and all data stored are encrypted using the AES-256 algorithm.

6.1 Kafka Topics ACL protection

As previously discussed, the event streaming service ingests real time data into Kafka topics. Each topic concerns a specific data type provided by a certain actor that needs a certain degree of data privacy.

To meet such requirements, the EGTN platform's Apache Kafka is using Access Control Lists (ACL) that uses the Simple Authentication and Security Layer (SASL) [4]. The user credentials are stored to the Kafka platform through a Java Authentication and Authorization Service (JAAS) configuration file. As a result, access levels can be adjusted for each user. Access can be completely refused to a topic just like any unauthorised user, or be provided as read-only access (consumer), write-only access (producer) or read-write access (consumer-producer). Both the users and their ACL configuration can be updated at any time.

Users/Topics	**bc-events**	**ll1-iot**	**ll3-events**	**dhl-metrics**	**citylogin-deliveries**	**weather-ingestion**	**timeseries-ingest**
interledger	**(R)W**	**R**					
sirma		**R**	**RW**		**R**	**RW**	**RW**
ilim			**R**			**R**	

[8] Apache Shiro, https://shiro.apache.org/

ngs		**RW**				**R**	
ebos	**R**	**R**	**R**	**R**	**R**	**R**	**R**
ibm				**R**	**R**	**R**	
dhl				**RW**		**R**	
citylogin					**RW**	**R**	
vltn					**R**		

Table 46: EGTN Kafka Topics ACLs

Specifically, each partner who produces or consumes data to or from one or more topics receives a set of credentials. A consensus has been reached between the partners regarding the access levels (0/R/W/RW) that each would have on every topic. Table 46 summarises the users, topics and the respective access levels that have been shared with the platform's administrators. A script that sets up the access levels of all the topics was then created and applied to the event streaming platform.

7 An Open-Source Blueprint for the EGTN Network

7.1 The EGTN Platform Functionalities and Innovation

The PLANET project aims at changing the way T&L actors interact, share information, and optimise operational performance under the principles of the PI paradigm. The EGTN Platform is a key-enabling element towards the realisation of this vision.

Modern T&L networks face several challenges, such as traceability of shipments, trust, and transparency between different stakeholders, and complex trade and customs processes that are difficult to manage. Lately, several solutions have been introduced aiming at the digitisation of the supply chain process - Tilkal, Transparency-One, IBM Food Trust [5], TradeLens [6], Trust Your Supplier. In a similar context, the PLANET project and the EGTN Platform aims to empower T&L stakeholders by offering them tools, services and guidelines for shipping, routing, and PI node optimisation as well as to collaborate with other actors in the supply chain, within and across borders in a self-determined and secure way. Through this lens, the PLANET project does not aspire to develop just another "platform". Instead, its ambition is to develop an original blueprint accompanied with best practices that support T&L actors in the definition and implementation of clear digital strategies and to offer support in their physical operations.

The open cloud-based infrastructure that is developed is the cornerstone of the PLANET project, as it offers the foundation on top of which the EGTN Platform and the EGTN services are developed, as shown in Figure 1. The unique combination of technologies and models includes among others: Blockchain services for interoperability of backend systems and intelligent forecasting algorithms for predictive analytics. More, specifically, the EGTN Platform Blockchain interoperability service aims to breakdown the silos of the different Blockchain systems/partners to support critical interorganisational trade workflow, while the use of smart contracts facilitates automated and paperless negotiations. The EGTN users can take advantage of these cutting-edge technologies along with the unique set of features offered by the platform, as it:

1. Improves customs control through the digitisation of the process
2. Increases trust but also confidentiality between different partners
3. Ensures the authenticity and the integrity of the data shared between partners

One of the key technological enablers for increasing visibility in logistics is the real-time data ingestion pipeline which can accumulate data from a plethora of data sources, ranging from IoT sensors to weather and traffic data. In this way, critical information such as waiting times, order status, or even delays in vessel journeys can be fed into the platform and used for offering T&L actors real-time automated decisions, such as dynamic contract activation. Moreover, the EGTN Platform provides T&L actors data-driven decision support services related to synchromodality, based on optimisation models and predictive analytics. More precisely, these services include corridor route optimisation, forecasting services for warehouses and ports and supplier collaboration analytics. Another key topic that PLANET project aims to address is the alignment of the EGTN Platform to the PI roadmap. It addresses key PI challenges by using intelligent forecasting such as predicting the use of resources in a PI node or rerouting cargo in case of congestion in one of the corridor ports.

Other key features of the platform include its modelling and simulation capabilities of analysing T&L and ICT innovations that position emerging technologies (e.g., Blockchain and IoT) as contributors to the Physical Internet, while its Human Machine Interfaces set a new standard for a more open and inclusive ecosystem where logistics partners share infrastructure and data and, in this way, overcome the silos of existing T&L systems and organisations.

7.2 Value to the T&L Industry

The EGTN Platform aims at empowering T&L stakeholders by offering them services and guidelines for shipping, routing and PI node optimization as well as for collaboration with other actors of the supply chain, within and across borders in a self-determined and secure way. Its unique selling point is that it is an accelerator service towards realizing the PI vision. It also aspires to be an inclusive and powerful platform, as it can be adopted by any size of T&L actor or company. As such, it is not limited to only large enterprises with expensive IT budgets, but smaller ones, hence taking a step closer to the realisation of the PI paradigm.

The architecture of the platform was designed in such way to enable its reusability and redeployment in different contexts by diverse T&L communities who need to collaborate and innovate in an open and transparent way. Except for providing a holistic approach for collaborating and optimizing processes along the entire supply chain, it stands out for its innovative interdisciplinary modeling approach. It utilizes simulations for incorporating the effects of geo-economic and technological changes and simulate the operational impact of aspects such as synchromodal planning toward the PI, and the corresponding business models that are supported by new generation optimization algorithms.

External benefits to the T&L industry include reduced management and operational costs, reduced environmental footprint in T&L, increased transparency and accountability of the actors/actions in T&L, provision of improved services to the final customers and finally, a framework for policymakers to monitor and act accordingly regarding fluctuations in infrastructural capacity of certain nodes.

The EGTN Platform is currently in TRL 4, since it is validated in the lab, and will reach TRL 5 at the end of the project, when it will have been validated in the LLs.

8 Conclusions

The purpose of this deliverable is to present the final version of the open cloud-based architecture of the EGTN Platform. The technical outcome of T2.1 is twofold, on the one hand to propose an architectural blueprint that enables any relevant organisation to build upon and to implement T&L design tools, collaborative logistics and new e-commerce models underpinned by data-driven supply chain insights and on the other hand to deliver an instantiation of the architecture on the cloud, aka the EGTN Platform to support the Living Labs activities and evaluate the architecture.

More specifically, the EGTN Platform brings together both real time and batch data from heterogeneous sources, which are then processed and used by the predictive and prescriptive models, part of the Decision Support Systems offered by the platform. A secure and scalable layer for data aggregation and data ingestion is proposed that enables the pre-processing and validation of data based on metadata. On top of this, the data-driven services developed in WP2 are deployed that offer optimisations in the selection of routes, ports and in general PI nodes towards developing an interconnected network and improve the efficient use of resources. The provision of a blockchain front-end for the interconnection of backend blockchain systems increases the visibility across the entire supply chain and enhances the collaboration between communities. Finally, critical data are presented to the end users through the dashboards developed in the frontend of the EGTN Platform. All these different components are brought together through the EGTN Platform and deliver an innovative solution that aims to achieve the overall goals of the PLANET project.

The innovative architectural blueprint enables different sizes of T&L/PI communities to deploy cloud-based platforms and improve their efficiency through collaboration, openness and secure access to advanced services. This is achieved following an explicit governance model for data, users and services integration as well as strict security and privacy rules to enable a trustworthy environment for sharing data.

9 References

[1] T. O. Group, "ArchiMate® 3.1 Specification," The Open Group Library, 4 November 2019. [Online]. Available: https://publications.opengroup.org/standards/archimate/c197. [Accessed December 2022].

[2] A. Tiwana, B. Konsynski and A. A. Bush, "Research Commentary—Platform Evolution: Coevolution of Platform Architecture, Governance, and Environmental Dynamics," *Information Systems Research,* vol. 21, no. 4, pp. 661-1010, 2010.

[3] "European Data Protection Supervisor," Europian Union, [Online]. Available: https://edps.europa.eu/data-protection/data-protection/glossary/d_en#:~:text=The%20principle%20of%20%E2%80%9Cdata%20minimisation,necessary%20to%20fulfil%20that%20purpose..

[4] "Simple Authentication and Security Layer specification," [Online]. Available: https://www.iana.org/assignments/sasl-mechanisms/sasl-mechanisms.xhtml.

[5] "IBM Food Trust," IBM, [Online]. Available: https://www.ibm.com/products/supply-chain-intelligence-suite/food-trust.

[6] "TradeLens," [Online]. Available: https://www.tradelens.com/.