Deliverable

D5.1 System Architecture & Implementation Plan

Project Acronym:	DUET
Project title:	Digital Urban European Twins
Grant Agreement No.	870697
Website:	www.digitalurbantwins.eu
Version:	1.0
Date:	30/11/2020
Responsible Partner:	ATC
Contributing Partners:	IMEC, AIV, VCS, TNO, P4A, KUL, AEGIS
Reviewers:	**Internal** Philippe Michiels (IMEC) Walter Lohman (TNO) Gert De Tant (OASC) **External** Pieter Morlion Andrew Stott Yannis Charalabidis
Dissemination Level:	Public
	Confidential – only consortium members and European Commission: X

This project has received financial support from the European Union's Horizon 2020 Programme under grant agreement no. 870697

Revision History

Revision	Date	Author	Organization	Description
0.1	05.09.2020	Athanasios Dalianis Marina Klitsi	ATC	Initial structure / TOC
0.2	30.10.2020	Athanasios Dalianis, Philippe Michiels, Marina Klitsi	ATC, IMEC	Provision of first draft
0.3	17.11.2020	Gert Vervaet, Philippe Michiels	AIV, IMEC	Provision of initial comments/suggestions
0.4	19.11.2020	Athanasios Dalianis, Marina Klitsi	ATC	Updated version sent for Internal Revision
0.5	24.11.2020 - 27.11.2020	Philippe Michiels Gert Vervaet Pieter Morlion Andrew Stott Yannis Charalabidis	IMEC AIV MORE LION AIV Un.Aegean	Internal review
0.6	28.11.2020	Athanasios Dalianis Marina Klitsi	ATC	Comments addressed
1.0	30.11.2020	Athanasios Dalianis	ATC	Final version

Table of Contents

List of Figures

List of Tables

Executive Summary

DUET is a diverse project with several work packages working in parallel in order to achieve common objectives. In order to start this work with a mutual sense of direction, we needed a high-level blueprint of the overall technology development within the project. The role of this document is to be a common point of reference throughout the project.

This deliverable is the outcome of the first task in WP5 and it presents the overall architecture of the DUET platform in terms of the supported functionalities, the respective processes and the components that realise them. The document presents a coarse-grained collection of technical components that provide the necessary functionality and defines the principles for their integration in a manner that ensures completeness, safety and efficiency. To this end, we propose a Service-oriented approach, where each functional unit is implemented as a stand-alone service, communicating in a standardized manner with other components.

The document sets the boundaries for the integration of the DUET modules by presenting the context and use for the DUET platform. As such, it builds on top of the technical project requirements and presents the way that the DUET capabilities will be able to integrate into the operational environments of the three pilots which are the early adopters of the proposed solution.

We consider the architecture to be a live document that will evolve and be aligned with the ongoing work in the project. An updated version of this deliverable will be delivered in M18 together with the first version of the Integrated Prototype.

Editorial notes

As stated in the Executive Summary, the architecture is a live document that will evolve and be aligned with the ongoing work in the project. An updated version of this report will be delivered in M18 (May 2021) together with the first version of the Integrated Prototype. The DUET technologies and requirements are expected to continue being investigated to ensure that the objectives and innovations of the project are valid, and the architecture and development work is aligned with those goals and requirements.

It has to be noted that the functional requirements as well as the Technical specifications presented in this document (Section 2) has been prepared based on the initial epics and the pilot scenarios identified by the Pilot partners. The architecture presented here will be used as a well-established basis during the elicitation process of epics in the future. The epics will be combined with an acceptance procedure of the presented architecture.

1. Introduction

This document is the confidential report for the specification of the DUET system, which analyses the requirements and the activities to be developed in DUET in order to deliver the relevant system for managing decision making in a virtual environment.

1.1 Purpose and Scope

The objective of this document is to present the overall architecture of the DUET platform in terms of the supported functionalities, the respective processes and the components that realise them. This document will serve as a reference point for the development work that will take place both in WP3, WP4 and WP5.
In addition, it presents the plan for the delivery of the integrated DUET prototype. This plan provides the roadmap for the development activities that the project should follow in order to produce a fully functional system. The decisions presented in this deliverable are subject to refinements and modifications, based on the progress of the technical work packages, as well as the validation and evaluation phases.

1.2 Relation to other Work Packages and structure of the document

This deliverable mainly affects the activities performed in WP5 aiming to provide the architectural and implementation aspects for the delivery of the DUET system taking into account the full range of requirements for such service. This document provides guidance on how the intended scenarios of use for the DUET system can be implemented in terms of the interconnection of the individual modules. On top of that, the pilot scenarios, as they have been introduced in WP2, provide the baseline for the integration requirements. The need for platform customization to meet the expectations of different pilots reflects the importance of the work in the definition of the pilot scenarios (in WP2) in the specification of the architecture and integration plan (done in this document).
To some extent, the work in this deliverable influences the exploitation activities in WP7, since the identification of integration requirements and the interoperability scenarios between the modules drive the licensing approaches for the DUET modules and exploitation potentials that need to be developed by each module owner.

This deliverable is organised as follows:

- Section 2 presents the functional and non-functional requirements of the system
- Section 3 provides a description of the DUET technical specifications derived from the functional requirements.
- Section 4 provides the description of the high level architecture of the DUET platform as well as a description of the components comprising the DUET platform.
- Section 5 provides the methodology that will be used in the context of the project.
- Section 6 provides an overview of the methodology that will be used for the system validation of the platform.
- Section 7 provides a description of the alpha version prototype.
- Section 8 provides the plan for the delivery of the DUET platform.
- Sections 9 and 10, provides the conclusion of the deliverable and the references.
- ANNEX I, the technical questionnaire used to collect partners’ input is presented

2. System Requirements

Below we present an overview of the epics, the main actors of the system and the relevant functionalities from the system point of view.

2.1 Overview of "epics"

As explained in the D2.3 [1] and presented in Section 5 of this report, DUET follows the Agile Software Development Practices [3] and as such the gathering requirements have been done through user stories using user interviewing, user observation, questionnaire and story writing workshops. To this end, the user requirements have been gathered and translated into user stories as part of epics, bigger user stories.
A first selection of epics is being presented in the following table, based on the D2.3 [1]. However, this list of epics will be further analysed by the Product Owner in cooperation with the DUET team.

Table 1: first selection of epics

Theme	Epic
Basic Infrastructure	- As a user of the digital twin, I can browse the 3D model of the area of interest so I can get a detailed look at the surrounding of areas I want to inspect. - As a user, I can browse the 2D road network on top of the 3D model so I can see where the roads are and can get extra attributes by clicking on a road - As a user, I can see the real time or historical data information being sent out by sensors on top of the 3D model, so I get an accurate indication of the current local status.
Traffic Model	- As a user, I see the prediction of traffic flows (e.g. the KUL, P4All model) as calculated in the traffic model of the area, so I can correlate predicted traffic flows with the current traffic flows. - As a user, I see the measurements done by the sensors interpolated (by a model that fuses information coming from different sensors from different sensor types) so I can get an approximation of the density of people also in places there are no sensors.
Air Quality Model	- As a user, I see the real time air quality model as calculated (by a model that fuses information coming from different sensors from different sensor types) so I can get an approximation of the air quality in places there are no sensors.

2.2 The DUET "Actors"

In order to identify the main 'Actors' relevant to the DUET system different user types are represented in this section as actors. An actor is anyone who exchanges data with the system. The basic actors (user categories) of the system are presented as follows:

Table 2: DUET System Actors

Actors	Role in the system	Description
Anonymous user	Anonymous User	The "Anonymous" user is using part of DUET functionality. Can browse the 3D model of the area of interest, browse the 2D road network on top of the 3D model and other info, following admin settings.
Digital Twin User	Registered User	The "Digital Twin User"can make use of the features of the DUET system i.e.: browse the 3D model of the area of interest, browse the 2D road network on top of the 3D model, can see the real time information being sent out by sensors, see the prediction of traffic flows, see the measurements done by the sensors interpolated, see the real time data model.
Data Consumer	Registered User	The "Data Consumer User" could make use of the features of the DUET system i.e. enters the Data catalog, view the list of the registered data sources and, request for data access and provide the necessary information.
Data Producer	Registered User	The "Data Producer User" could make use of the features of the DUET system i.e. enters the data catalog UI, selects to add a new data source, accepts / denies the relevant request.
DUET System Administrator	Administrator	The "System Administrator" is any authorized person who monitors the system from the technical point of view, ensures its proper operation and handles users and roles and statistics. The platform as a system that needs to be maintained and monitored shall have a person or organization which will be responsible for the overall administration of the system, user management (rights, permissions, accounts) and solving problems. The system administrator is also responsible for the overall operation of the platform and especially for the ways the content is provided to the users and can be made available through different communication channels in an easy and effective way.

The functionalities available for the actors of Table 2 are described in the following section.

2.3 The DUET functionalities

A description of the functionalities of the DUET is provided in this section based on the D2.3 [1]. It describes the set of functionalities that have a substantial architectural coverage or that stress or illustrate a specific, delicate point of the architecture.

2.3.1 Overview of the functionalities for the Anonymous User

F1: Browse available datasets

- The user will be able to access the DUET Data catalog in order to search for available datasets.

F2: Browse public cases

- The user will be able to view available cases, that is, predefined combinations of data layers, models and dashboards.

2.3.2 Overview of the functionalities for the Digital Twin User

F3: Register on DUET

- The user should create an account in order to register to the DUET platform. For this purpose, the user initiates a request for registration to the system by providing his/her personal data. For registration the user provides the following data:
    - first name,
    - last name,
    - email address,
    - preferred username and password

  If the users are already registered, then they have to login to the system by providing their username and password in order to access the appropriate system's web pages and functionalities.

F4: Browse available datasets

- The user will be able to access the DUET Data catalog in order to search for available public and private datasets.

F5: Access the 2D / 3D map visualization components

- view 2D/3D map city representations
- view data on map like traffic, air pollution, noise pollution etc

F6: Access public cases

- view available cases
- create new cases

F7: Access to city dashboards

- access the available dashboards
- view statistics related to city data

2.3.3 Overview of the functionalities for the Data Consumer

F8: Register on DUET

- The Data Consumer should create an account in order to register to the DUET platform. For this purpose, the user initiates a request for registration to the system by providing his/her personal data. For registration the user provides the following data: first name, last name, email address, preferred username and password. If the users are already registered, then they have to login to the system by providing their username and password in order to access the appropriate system's web pages and functionalities.

F9: View Notifications

- enter the notification UI
- view a list of notifications

F10: Register to data sources for data consumption

- enters the Data catalog UI
- view a list of the registered data sources
- selects a data source and views details
- request for data access
- provide necessary information e.g. purpose of data usage, preferred communication protocol etc
- receives connectivity details through the UI

2.3.4 Overview of the activities for the Data Producer

F11: Register on DUET

- The Data Producer should create an account in order to register to the DUET platform. For this purpose, the user initiates a request for registration to the system by providing his/her personal data. For registration the user provides the following data: first name, last name, email address, preferred username and password. If the users are already registered, then they have to login to the system by providing their username and password in order to access the appropriate system's web pages and functionalities.

F12: View Notifications

- enter the notification UI
- view a list of notifications

F13: Register data sources / model

- enters the Data catalog UI
- selects to add a new data source
- enters information about the data source like title, description, quality and service level
- maps registered data source with the DUET ontologies

F14: Allow data access to consumers

- enters the Data access UI
- review the access requests
- accept/deny the relevant request
- view statistics about the provided data sources

2.3.5 Overview of the functionalities for the System Administrator

F15: Configure DUET Metadata

- enter the configurations UI
- sets or updates the DUET metadata like terms and conditions, geographical boundaries, contact information etc

F16: Add a DUET ontology

- enters the knowledge graph UI
- uploads a new ontology

F17: Manage Users

- enters the users' management UI
- view a list of users
- add a new user
- delete a user
- search for users based on criteria
- update user's data
- assign roles to a user
- view selected user's data

F18: Manage Roles

- enters the roles management UI
- view a list of roles
- add a new role
- delete a role
- search for roles based on criteria
- update roles
- view selected role's details

F19: View Notifications

- enters the notification UI
- view a list of notifications

F20: Monitor System

- enters the monitoring UI
- view statistics about the system

2.3.6 DUET Functional Requirements

This section presents a set of functional requirements, based on the functionalities identified in the previous sections and the elicited user needs as derived from D2.3 [1]. As the DUET system design and implementation goes on, all these requirements will be refined and specified in more detail.

Table 3: DUET Functional Requirements

No	Description	Associated functionalities
FR1	DUET shall provide an easy registration process for different types of users as well as personalized access to the system.	3,8, 11
FR2	DUET shall provide the means for a system admin to manage system resources and to authorize (or deauthorize) users to specific roles and functionalities.	17,18,20
FR3	DUET shall provide the mechanism for the system admin to configure the metadata.	15
FR4	DUET shall provide the means for searching, downloading and querying data related to the traffic, air pollution, noise pollution.	1,4

FR5	DUET shall provide to the user a visual presentation of the city.	5
FR6	DUET shall provide to the user a mechanism to upload an ontology.	16
FR7	DUET shall provide to the user a mechanism for receiving notifications.	9,12,19
FR8	DUET shall provide the means for registering data related to the traffic, air pollution, noise pollution.	13
FR9	DUET shall provide the mechanism for the user to have access to data layers, models and dashboards.	2, 6,7,14
FR10	DUET shall provide the means for registering to data sources for consumption.	10

2.4 DUET Non - Functional Requirements

This section analyses the non-functional requirements that should apply for the DUET platform components. The DUET platform should satisfy a set of non-functional requirements, which will ensure the normal operation of the system and the provision of a proper environment for the desired system level functionalities. The non-functional requirements are depicted in the following table.

Table 4: DUET Non - Functional Requirements

ID	Requirement	Description
NF1	Storage	Relational databases like MySQL or no-SQL databases like MongoDB, will be used for storing various types of data e.g. the registered data sources and their metadata, user preferences, event data or temporary information like caches of query results, etc. An RDF-based repository is going to be used as well for storing the ontologies of the system.
NF2	Performance	The system should respond in a timely manner using the predefined resources when an increase in users/datasets occurs.
NF3	Security	The system should be secured against sabotages arising from all types of hacking attacks. Security techniques that discourage data loss and misuse of data for fraudulent acts should be utilised.
NF4	Privacy	A single authentication process, or secure access keys for APIs, will be required for accessing the personalised functionalities of the system. Personal data and user created content will never be published without user consent.
NF5	Data Licensing	The system must account for different data licensing.
NF6	Scalability/Expandability	The system should be able to handle the increasing size of datasets, as well as a potentially increased number of users.

NF7	Availability	The system should ensure that users have always access to data and associated assets 24/7 with 99.9% reliability. This requirement entails stability in the presence of localized failure.
NF8	Usability	The DUET system should have an attractive and intuitive User Interface. The interface needs to address various user groups and therefore should be easy to use and give access to all system functionalities providing easy navigation through all features.
NF9	Interoperability	The DUET system should use communication protocols that allow its use by different systems and devices.

3. DUET Technical Specifications

Based on the above analysis the following table summarises the technical specifications of the DUET system derived from the functional requirements. Every technical specification has at least one relevant user activity and provides a description of what the system has to support.

Table 5: DUET Technical Specifications

No	Description	Associated FR
TS1	The platform allows users to register and get assigned with various security roles.	FR1
TS2	Users can set their profile details as well as choose a set of personalisation parameters.	FR1
TS3	Admin functionality which allows the administration of users and their security roles.	FR2
TS4	Admin functionality which allows the administration of the DUET metadata.	FR3
TS5	A free-text search mechanism to find datasets in the DUET catalog, based on their available metadata.	FR4
TS6	Every dataset must offer a download link which fetches the actual data, when possible, e.g. historical data.	FR4
TS7	An API which will allow querying of data.	FR4
TS8	The platform will provide a 2D / 3D representation of the city.	FR5
TS9	An API for uploading RDF data.	FR6
TS10	A notifications mechanism that sends messages of importance to the users.	FR7
TS11	A data catalog that allows the users to register their data sources, models and visualisations and also to apply for data consumption.	FR8, FR10
TS12	A data access mechanism for data sources, models and UIs.	FR9

4. Platform Architecture

This section presents the initial blueprint of the DUET platform architecture. A high level architecture of the platform is described in order to set the stage for the development of the first prototype. It must be noted that the decisions presented in this section are subject to refinements and modifications, based on the progress of the technical work packages, as well as the validation and evaluation phases.

4.1 High Level Architecture

In this section we provide a high level overview of the platform architecture.

In order to tackle the non-functional requirements and technical specifications described in the previous sections, an event driven, microservices approach is proposed, where the core DUET components are connected with external systems like IoT data sources, models or apps, through a set of specialised Gateways. These core components constitute the "DUET cell", (Figure 1).

In summary the core components that comprise the DUET cell are:

- A set of Gateway components that lay on the edge of the DUET cell and control the data flow in and out of the cell
- A message streaming platform that allows components to exchange data and events in an asynchronous manner
- A data catalog for registering different kind of data sources
- A management module that controls the users, access rights and various configurations of the DUET platform

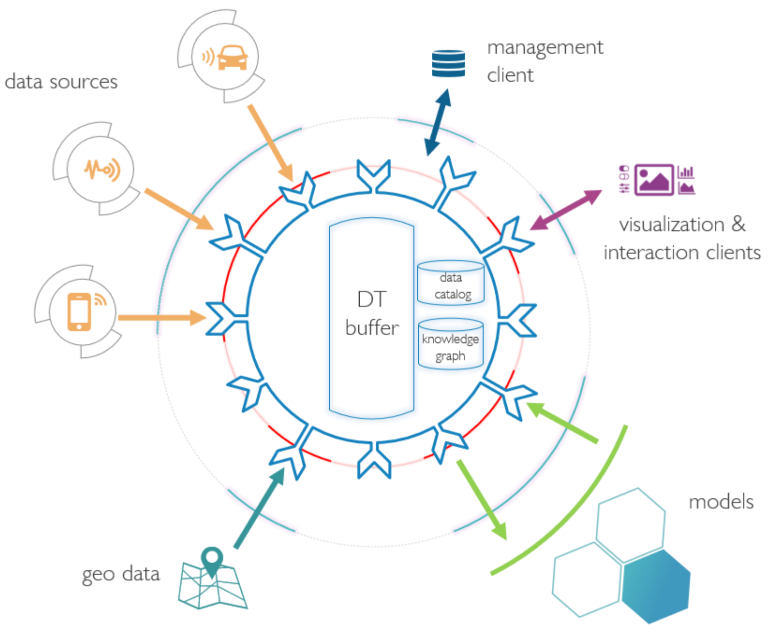


Figure 1:The DUET Cell

On top of the DUET cell, additional user interfaces and components will be implemented in order to address the user needs, validate our approach and compose along with the cell, the DUET platform that will be provided to the pilot cities at the end of the project.

On top of the DUET cell, additional user interfaces and components will be implemented in order to address the user needs, validate our approach and compose along with the cell, the DUET platform that will be provided to the pilot cities at the end of the project.

The DUET platform follows a layered architecture pattern, thus the system's components are divided into multiple distinct layers, each responsible for a specific set of functionalities. By convention, the layers interact with each other in a top-down manner, with each layer being able to access all layers below. Figure 2 presents a high-level overview on the DUET platform architecture.

The layers are described as follows:

- **Presentation Layer:** Contains all the User Interfaces and Visualization Modules that help the user to get all the information that the DUET platform offers in an easy (self-explanatory), efficient, and enjoyable (user-friendly) way.
- **Access Control Layer:** Contains all the necessary Gateway components, for controlling the data flow in and out of the platform
- **Service Layer**: The layer that exposes the core components's functionalities as a set of APIs
- **Business Layer:** It encapsulates all the business logic of the system that tackles the user needs.
- **Data Layer**: This layer consists of the logical set where all the heterogeneous data sources are archived. It is comprised of repositories holding information about the events of the system, the data sources, the ontologies, the users etc.
- **Infrastructure Layer:** It consists of the physical or virtual servers of the platform and the relevant supporting tools.
- **Security Layer**: A vertical layer that implements the necessary security measures for all the other layers

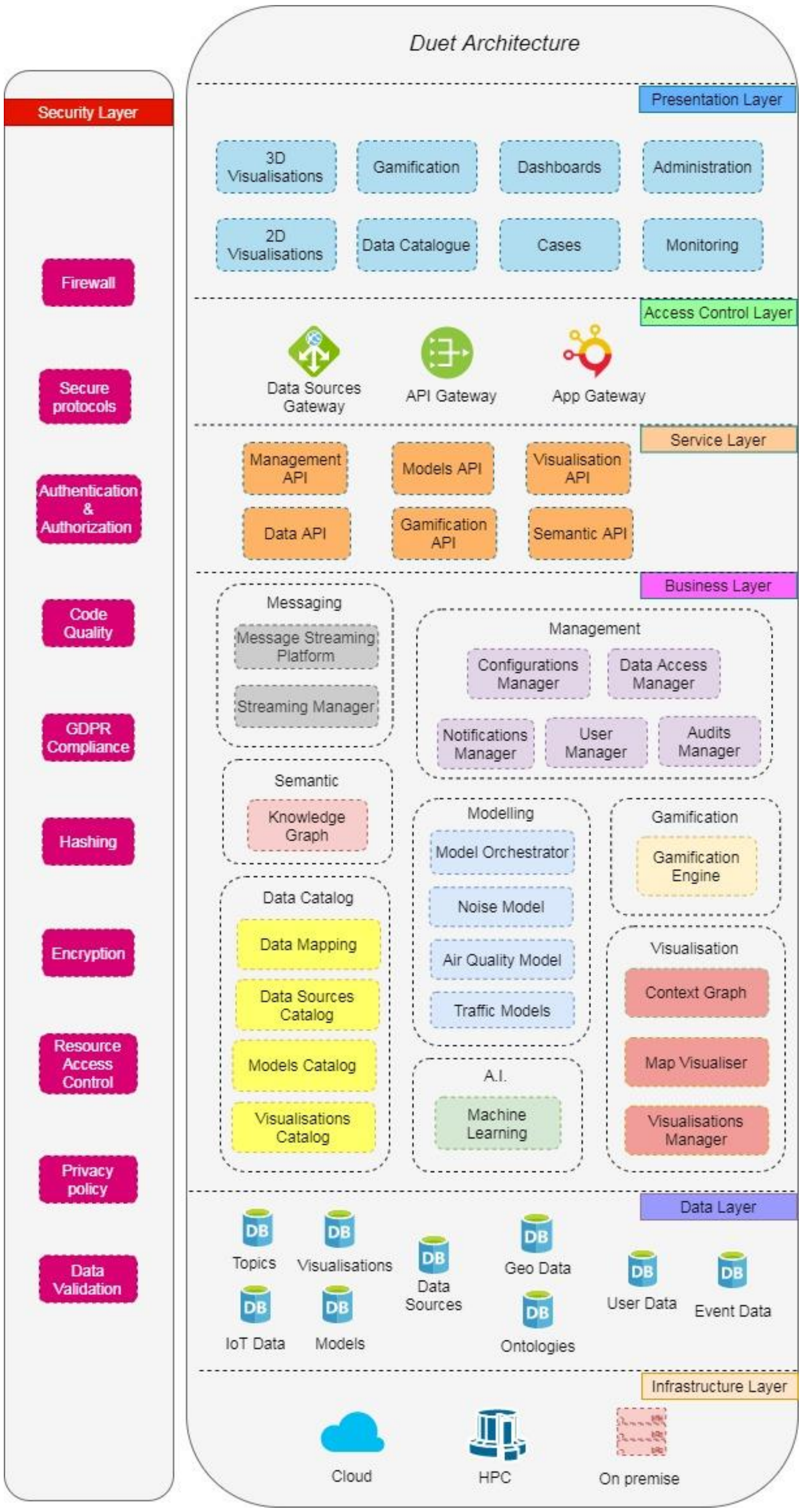


Figure 2: High level architecture of the platform

4.1.1 Security

DUET will institutionalize security-and privacy-by-design principles to prevent all kinds of security risks, from hackers trying to steal intellectual property or a distributed denial of service attack. The system will guard against security risks that may touch upon the various layers of the DUET architecture and enforce different complementary security measures. Some of these measures are presented but not limited in the list below:

- The internal components of the platform are guarded by the DUET Gateways. All communication between the Gateways of the platform and the external applications is done through secure protocols like https, wss etc.
- Additional to the secure communication protocols, the different Gateways of the platform will perform user and role authentication and authorization.
- The Gateways will also perform data validation on the messages sent, to avoid any attempt to corrupt the internal data.
- Users and roles are also defined on a database level, where different roles have different privileges on the various entities of the platform.
- Any sensitive information like user passwords will be sufficiently protected by using the necessary hashing / encryption algorithms and GDPR compliant rules will be imposed.
- At application level, all components that will be deployed on the platform, will be scanned for security vulnerabilities.
- At the network level, a firewall employs rules that allow the traffic flow only through specific ports and domains to the services of the system.
- Remote or physical access to the servers is provided only to authorised personnel.

More details about the security aspects of the platform can be found at D3.10 [5].

4.2 DUET Components

This section analyses the components comprising the DUET platform. A technical specification questionnaire (see ANNEX I) was circulated among the technical partners of the project in order to gather information about existing components and tools that will provide the necessary backend and frontend facilities to support the desired functional requirements.

4.2.1 DUET UI

DUET will offer a set of UIs that are needed in order to tackle the user and technical requirements of the project. These in summary include:

- the 2D/3D map visualisations and relevant data integrations e.g. traffic and air pollution data presentation
- the data catalog UIs, that allows authorised users to view and register various types of data sources
- management UIs that allows an administrator to manage various entities of the system
- monitoring visualisations that provide statistics to an administrator about the health of the system
- gamification UIs present to the user gamification elements like badges, points based on his/her actions in the system etc.
- user dashboards depicting useful aggregated information

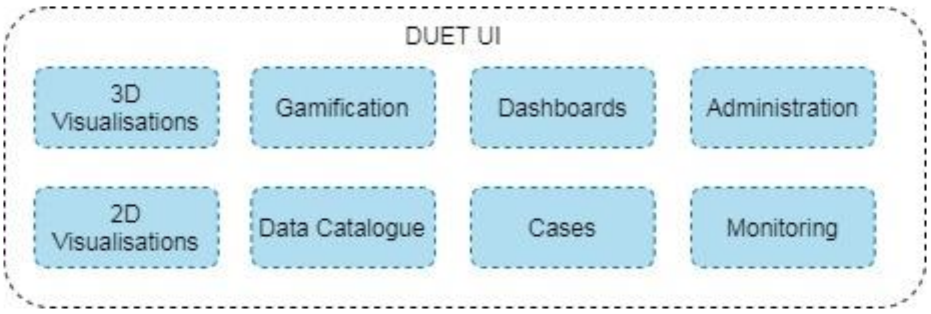


Figure 3:The DUET UI

More information about the user interfaces of the system can be found at D4.1 [7].

4.2.2 App Gateway

The App Gateway, intervenes between the registered models and other external systems like visualisations, except IoT sources, controlling the data flow from/to them and the Message Streaming Platform. Its two main subcomponents are the Message Receiver and the Message Sender (Figure 4).

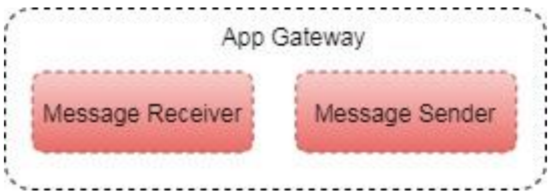


Figure 4:The App Gateway

Message Receiver

The Message Receiver exposes a REST API through which the models can send their data to the DUET cell. The communication is done through HTTPS and the Message Receiver performs three main actions:

1. Model authentication, where the Message Receiver checks the provided security keys in the message against the ones obtained during the model's registration (in coordination with the Management component)
2. Data validation, where the data sent are validated against the structure defined upon the model's registration (in coordination with the Data catalog)
3. Message transformation, where the message is sent to the Message Streaming Platform in the appropriate format.

Message Sender

The Message Sender is responsible for sending data outside of the DUET cell to the registered models or other external systems like visualisations.
The communication is done through secured protocols like HTTPS, WSS or MQTTS depending on the relevant provided information upon the model's or visualisation's registration. The Message Sender performs the following actions:

1. Listens to specific topics of the Message Streaming Platform for messages that need to be send outside of the cell
2. Maps the data received in a topic to the structure (in coordination with the Data Mapping Registry) requested from the model(s) / visualisations "listening" to the same topic
3. Converts the message and sends it through the channel requested by a model "listening" to the same topic e.g it sends it through WSS or HTTPS

4.2.3 Data Sources Gateway

Similarly, to the App Gateway, the Data Sources Gateway is mainly responsible to control the data flow from the IoT sources to the DUET cell. The Data Sources Gateway consists of a set of specialised connectors based on the type of data exchanged (Figure 5), that receive data from IoT sources and forward the data to the Message Streaming Platform of the system. The connectors of the Data Sources Gateway are described in more detail in D3.8 [6].

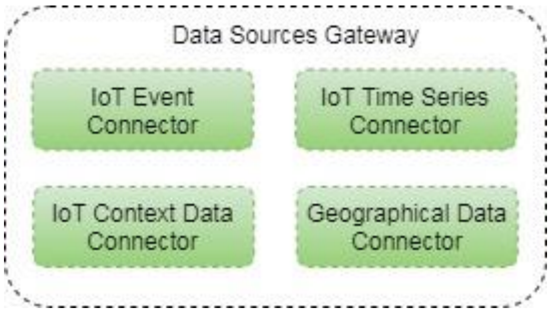

Data Sources Gateway
IoT Event Connector
IoT Time Series Connector
IoT Context Data Connector
Geographical Data Connector

Figure 5:The Data Sources Gateway

IoT Event Connector

The IoT event connector provides an interface for IoT event sources to connect to the DUET platform and send in their events. Data coming from actual sensors deployed in the field needs to enter the twin with a reasonable latency to reflect the actual state of the urban area covered. There are two main strategies to achieve this, subscribing and polling both of which will be implemented in the context of DUET.

IoT Time Series Connector

The time series connector provides an interface for time series data sources to connect to the DUET platform. By implementing this interface, DUET will be able to send user queries to the time series data source, capture the response, map it and make it available to the user

IoT Context Data Connector

Many IoT stacks keep track of the IoT context in a context broker. These brokers typically expose their data via an API that permits to do some querying on them. The IoT context data connector is responsible for retrieving data from these brokers and forwards them to the DUET cell, through the message streaming platform.

Geographical Data Connector

Like the time series connector, the geographical data connector provides a way for geographical data sources such as WFSes to connect to the DUET cell. The strategy for wiring up geographical data will be very similar to that of time series data. Connectors supporting specific standard services could be provided out of the box. Examples of such standards are: CityGML, GeoSPARQL, GeoAPI/GeoJSON, WFS.

4.2.4 API Gateway

In microservices architectures, where many services are deployed in a number of virtual or physical nodes and multiple instances of the same service can exist, an API Gateway is a necessity.

An API Gateway is a component that intervenes between a web client and the backend APIs, acting as a reverse proxy that forwards the request to the appropriate microservice, usually after proper authorization.
The API Gateway thus provides a single point of access to UIs and in general external applications, decreases the complexity of implementation and allows security measures, and functionalities such as load balancing and service discovery to be applied more easily.

4.2.5 Management component

The Management component is responsible for the initialisation of the DUET cell and the management of key entities of the system like users, roles and access rights (Figure 6). The Management component consists of the sub-components described below.

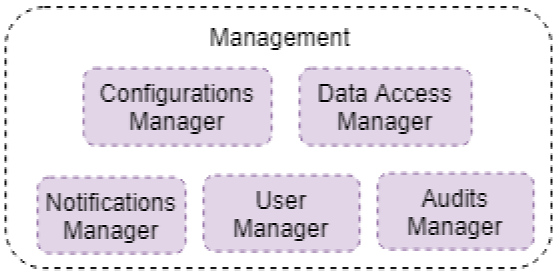


Figure 6:The Management component

User Manager

The role of the User Manager can be summarized as follows:

- It provides authentication and authorization mechanisms based on Json Web Tokens
- It allows for user registration and user profile edit
- It allows for role definition and user assignment
- It allows for user group definition and user assignment
- It allows for permissions definition and user groups / users assignment

The User Manager provides both APIs and a UI for managing the related entities.

Data Access Manager

The Data Access Manager is responsible for handling access rights related to data sources registered in the system.

It allows a Data Consumer to request for data source access permissions and a Data Provider to allow access to his / her data sources either to individual users or a group of users following the principle of least privilege (PoLP).

Audits Manager

The Audits Manager is responsible for monitoring the user's actions and store information about them for future use e.g. for UX improvements or stress testing, always in respect to GDPR and the privacy policy of the project.

The Audit Manager doesn't expose an API but receives events in a specific format from all the components involved, regarding the user activities.

Configurations Manager

The Configurations Manager is responsible for setting up specific metadata about the DUET instance like contact information, terms and conditions etc, as well as automate the initialisation of the DUET cell.

It exposes its operations through a set of secured JSON / REST APIs (Configurations Microservice) that could be consumed by a Configurations page in the Administrator's Dashboard.

Notifications Manager

The notifications manager listens to events coming from various DUET components and notifies the appropriate users in case a message of importance arrives, for example it notifies a Data Provider that there is a request for accessing one of its data sources or an Administrator if a service is offline.

4.2.6 Messaging Component

The Messaging component, provides the necessary tools and operations that permit the seamless communication between the DUET components and is composed of the Message Streaming Platform and the Streaming Manager.

Message Streaming Platform

The Message Streaming Platform is the heart of the DUET cell. It allows the different DUET components to communicate and exchange data in an asynchronous way.

Although the DUET components will be implemented in such a way that any relevant Streaming Platform can be used with the appropriate extensions, for the purposes of the project, we are going to use Apache Kafka [12].

Apache Kafka is an open source Massage Streaming Platform, with characteristics like stream processing, highly scalable architecture, high availability and throughput, as well as a large ecosystem of open source tools around it. All these characteristics make Apache Kafka a perfect option for the DUET project.

Apache Kafka is used in a master - worker mode, thus it is possible to create a cluster of Kafka Brokers, each one being able to support thousands of message queues that are called event topics. Every component that needs to exchange data through Kafka, is required to send its data to specific topics in a certain format and listen to specific topics to retrieve them.

Streaming Manager

The Streaming Manager provides operations related to the Message Streaming Platform like retrieval of topics, topic creation, etc that can be triggered by an Administrator manually or by the system automatically upon the reception of relevant events. For example the Streaming Manager listens to data registration events produced by the Data catalog and creates the appropriate topics.

4.2.7 Data Catalog

The Data catalog is the main metadata repository and a key component of the project. It provides to the users all the necessary interfaces that allows them to register their data sources, models and visualisations. It also offers to the system, mapping capabilities between different interpretations of data.

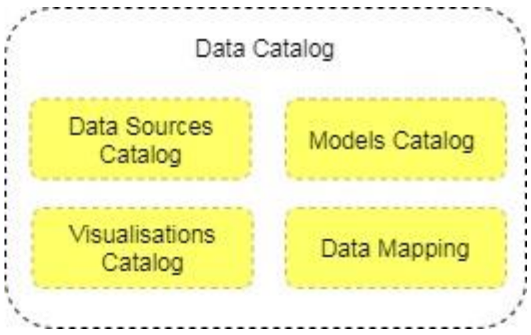


Figure 7:The Data catalog

Data Sources catalog

The Data Sources catalog handles the registration process of the various data sources and stores its metadata. As data sources we consider:

- IoT data sources providing data like IoT context data, geographical data etc
- REST endpoint providing data like static files available on web or relevant web services

In both cases a Data Provider registers these sources by providing the necessary metadata like description of source, data definitions, license etc.

Models catalog

The Models catalog is used by the Data Providers to register their models. As models we consider actual running components that react to specific conditions and after a series of calculations, provide to the system geographical data that can be visualised on a map. In the context of the DUET project, we are going to implement two traffic models, a noise pollution model and an air quality model.

For the registration process, a Data Provider enters information about the model such as, description, license, inputs and outputs, initialisation parameters, data sources needed and preferred way of communication.

Visualisations catalog

The Visualisations catalog allows authorized DUET users to provide information about their visualisations, that is, web applications that they have implemented and consume data registered in DUET. These applications potentially can be deployed inside the platform, after a validation process and be used by the DUET Digital Twin users.

Data Mapping

The Data Mapping component, implements the necessary functionalities, in order to map the data flowing into a DUET topic from a source, to the internal DUET data structure and vice versa from the internal data structure to the one needed by a model or visualisation listening to a specific topic.
More information about the Data Catalog can be found at D3.8 [6].

4.2.8 Knowledge graph

The knowledge graph will contain a formal semantic definition of data offered through the registered data sources and as such may assist in providing validation components to ensure that published data conforms to these definitions. It makes the information offered through the data APIs searchable for both systems and humans and allows the creation of a navigable structure on top of registered data sources. More details about the Knowledge graph can be found at D3.8 [6].

4.2.9 Model Orchestrator

When a model is registered, the Data Provider defines the initialisation parameters, as well as the start or stop conditions of the model. The Model Orchestrator is responsible to initialise properly the models, if needed, and monitor them so that it stops or restart the models when the specified conditions are met.

4.2.10 Visualisation component

The visualisation component is responsible for handling different aspects of the core UIs of the DUET platform. It consists of the following:

Context Graph

The context graph serves the following purposes:

1. Tracking context: as IoT events come in, the context graph can be updated to keep track of the last state (e.g., the last measurement of a sensor),
2. Interlink objects in the city (in a graph) such as streets, buildings, sensors, etc. This allows to let information bubble up or down the graph and aggregate data,

3. Providing a handle to manipulate objects is the visualization so that what-if scenarios can be tested. Good examples are the manipulation of buildings, streets, etc.

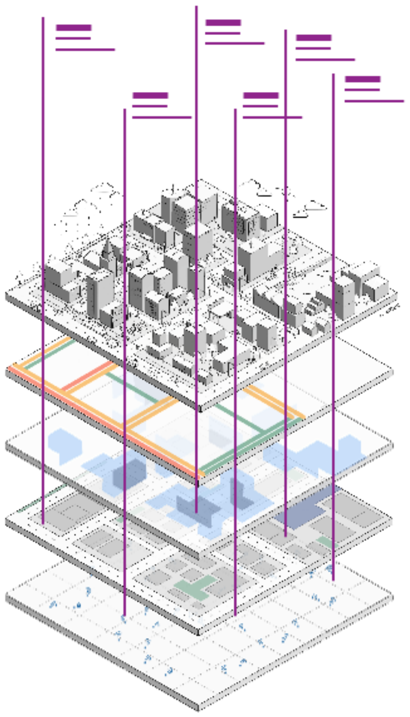


Figure 8:Conceptual drawing of the context graph, a property graph linking smart city artifacts

Map Visualiser

The Map Visualiser, encapsulates the business and presentation logic of the 2D/3D visualisations. It integrates various data sources of the DUET system like geographical data, air pollution data, noise quality or traffic data etc and presents them to the user in the form of 2D and 3D maps.

Visualisations Manager

The Visualisations Manager coordinates the different core UIs of the system offering an integrated view to the users, responds to user interactions realizing part of the use cases of the project and enables or disables registered visualisations based on the user settings.

4.2.11 Machine Learning

The Machine Learning component applies artificial intelligence techniques to the data flowing into the platform in order to analyze them and extract useful information like data correlations or predictions. The exact usage of this component will be further investigated during the course of the project and the evolving epics and user stories.

4.2.12 Gamification Engine

Gamification is the application of techniques found in games to a non game context. The gamification techniques take advantage of the intrinsic and extrinsic human motivations in order to increase the user's participation in certain aspects of a software application.

The DUET Gamification Engine will monitor the user actions in the DUET UIs and the events produced in the system, and implement the necessary game mechanics that will increase the user's satisfaction of the system and engage him/her more on its functionalities.

The exact usage of this component will be further investigated during the course of the project and the evolving epics and user stories.

4.3 DUET Components Interaction

In this section, the architecture is described through block/sequence diagrams stressing the interactions between the different components that compose the DUET platform.

The following diagram presents the components' main interactions between them and with the external systems in order to realise the user requirements. For clarity, the diagram does not depict all the possible communications between the components but provides a good overview. The main interactions can be summarised in the following list:

- The App Gateway exchanges data with models and external applications over secure protocols. Similarly, the Data Sources Gateway interacts with the IoT sources and the API Gateway with other external applications like visualisations.
- The DUET UI interacts with the users over HTTPS. In order to communicate with the internal APis of the cell, the DUET UI communicates only with the API Gateway that provides a single point of access.
- The App Gateway, with every message that arrives, performs user authentication and authorization with the help of the Management component and data validation with the help of the Data catalog. The same applies for the Data Sources Gateway. The API Gateway only performs user authentication and authorisation.
- The App Gateway performs data mapping on the messages to the applications out of the cell with the help of the Data Catalog.
- The Data catalog communicates with the Knowledge Graph during the registration of a new data source, for retrieval of the available ontologies and storage of the semantic representation of the datasource.
- All components besides the API Gateway send events to the Message Streaming Platform during their operations.

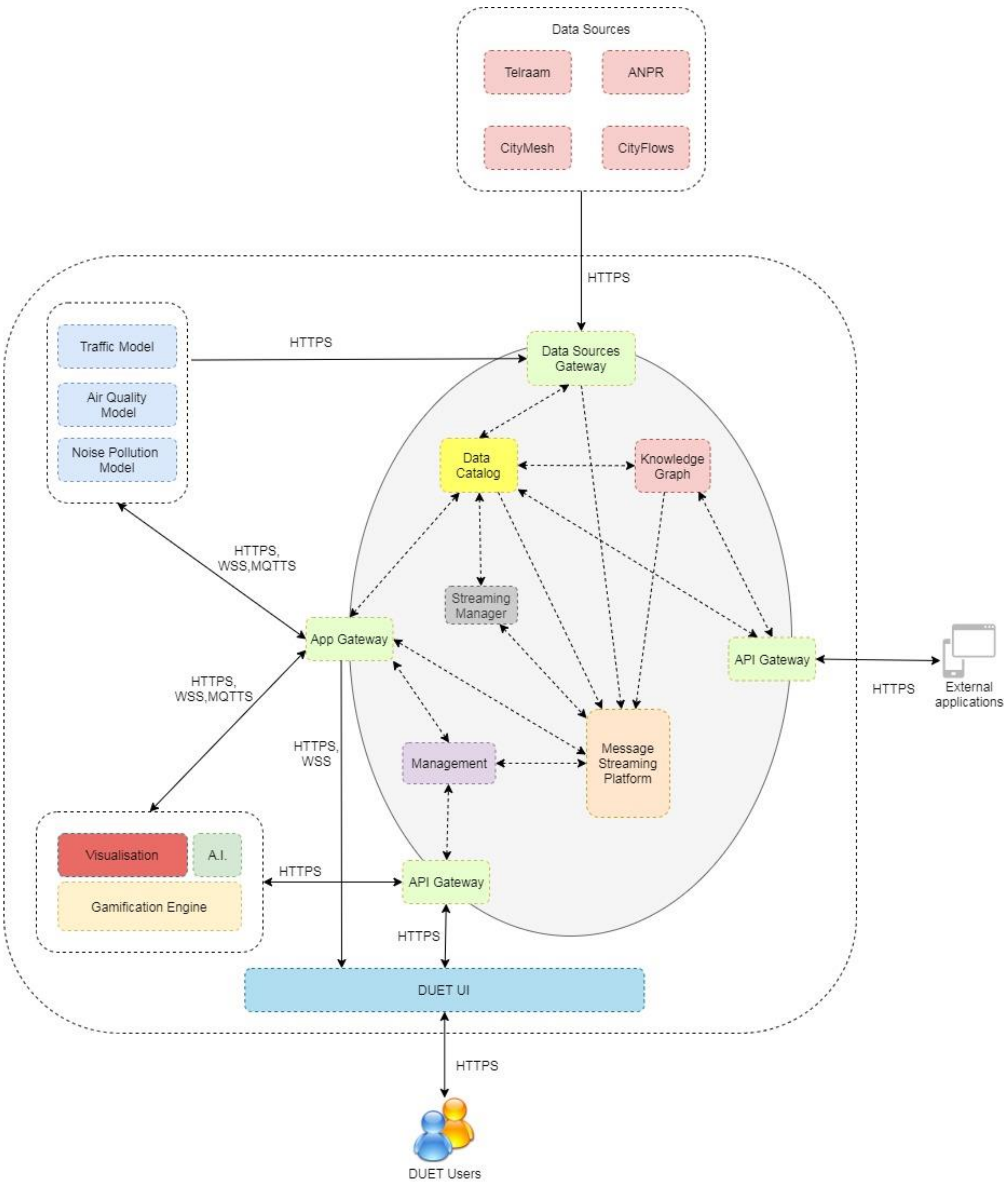


Figure 9:Components interactions

5. Integration Methodology

For the integration purposes of the DUET project we are going to follow the Agile Software Development Practices with frequent integration cycles, rapid prototyping and close collaboration between self-organizing, cross-functional teams. Based on agile principles, we are also going to apply Continuous Integration techniques for performing automated building, testing and deployment of the provided modules. For adopting Continuous Integration & Continuous Deployment practices we are going to set up a development environment containing a set of continuous integration and deployment tools.

5.1 Agile Methodology

A research among the most dominant development methodologies [13] indicates that the most appropriate way of implementing integration mechanisms for the DUET platform would be 'Rapid Application Development'. This implies that a system prototype is implemented, tested and evaluated in an iterative manner, using short cycles to add functionality to the prototype. This is more suitable for an Innovation action project aiming to deliver a system prototype, since it enables end users to continuously participate in the development of the integration mechanisms and guide the development towards their needs. In this manner, the processes of implementation and definition of the integration mechanisms will proceed in parallel until the end of the project by means of close collaboration between all the teams. One of the most popular types of Rapid Application Development is the 'Agile Methodology', which is associated with a list of terms and rules that have to be followed during development as described in the 'Agile Manifesto'[14].

Agile methodology implies and enforces collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change. Some of the principles of the Agile Manifesto are:

- Welcome changing requirements, even late in development
- Working software is delivered frequently (weeks rather than months)
- Working software is the principal measure of progress
- Customer satisfaction by rapid delivery of useful software
- Close, daily cooperation between business people and developers
- Projects are built around motivated individuals, who should be trusted
- Continuous attention to technical excellence and good design
- Simplicity
- Self-organizing teams
- Regular adaptation to changing circumstances

The methodology workflow could be reflected in the following diagram:

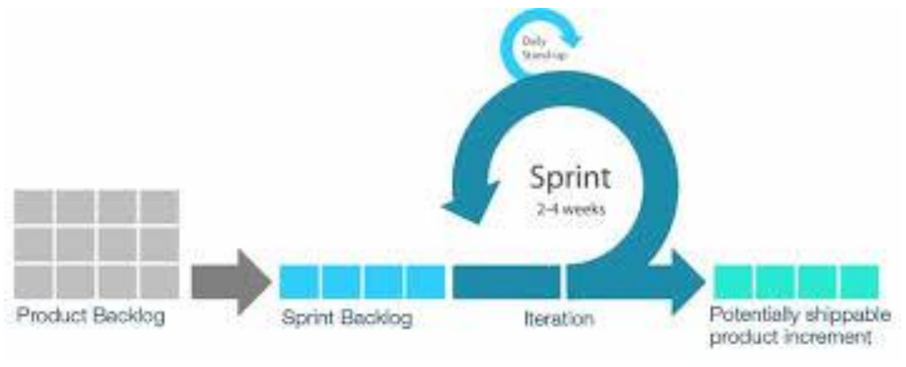


Figure 10:Agile methodology workflow

5.1.1 API Guidelines

Most of the components developed in DUET will have APIs exposed. It is important that these APIs follow best practices for better understanding and communication. The minimum API guidelines that need to be followed are summarised in the following list:

- All APIs will use the JSON / REST protocol, over HTTP for internal APIs or HTTPS for publicly available ones
- APIs must return the appropriate HTTP status codes based on status code definitions [9]
- APIs must use the correct service methods [10] for their operations
- APIs must be cacheable when possible
- APIs must support versioning
- APIs must support security measures such as authorization headers
- APIs must support pagination

5.1.2 Tools and CI / CD process

In this section we present the tools that will be used for the integration of the components of the DUET system. An overview of these tools is presented in the table below.

Category	Tool
Task management	Jira
Component packaging	Docker
Component orchestration	Kubernetes
Cloud services provision	Microsoft Azure
Code repository	Gitlab
Component images repository	DockerHub
Code Testing	JUnit, Mockito, Mocka, Jest, Nose
Monitoring	Prometheus, Grafana
Code Quality	SonarQube

Task management

In order to coordinate the technical efforts between the technical partners and break down the user requirements into technical tasks, in compliance to the agile principles, we have set up a DUET space on Jira Cloud [11].

Jira is a tool that provides an easy way to create epics, user stories and tasks, to plan agile sprints and assign work to the agile teams, and also to keep track of the progress done.

For the purposes of the project:

- Initially the epics, user stories and tasks are being set (for the Alpha version this is completed already). These are being refined frequently based on the users' feedback
- bi-weekly sprint planning sessions take place, where each sprint contains the prioritised tasks that can be implemented in the given time, after discussion with all the technical teams.
- bi-weekly sprint reviews take place where the results of each sprint are presented to the pilots

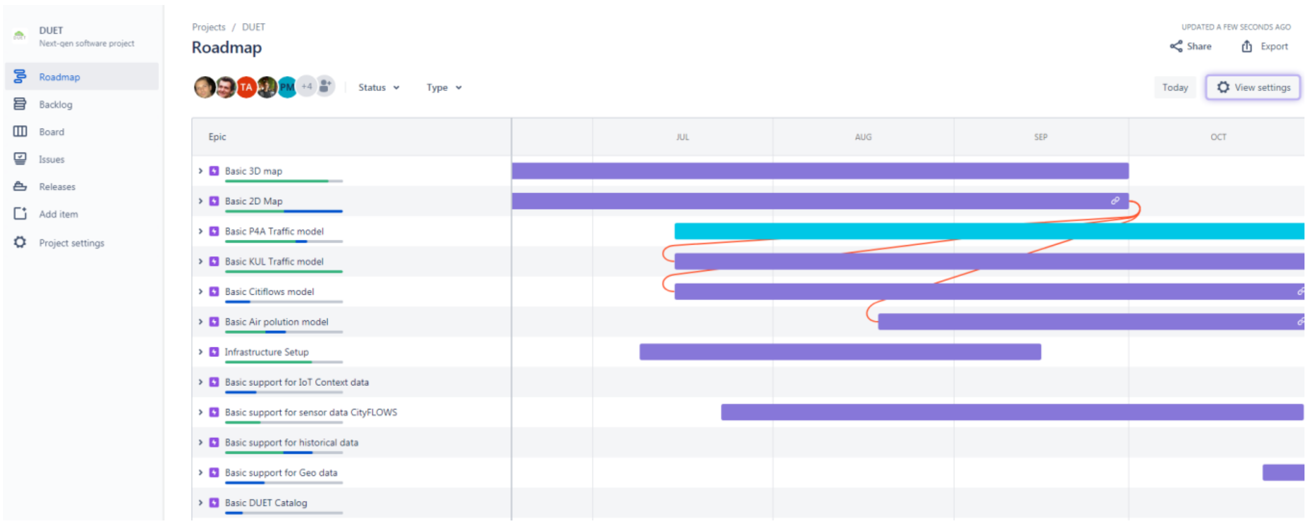
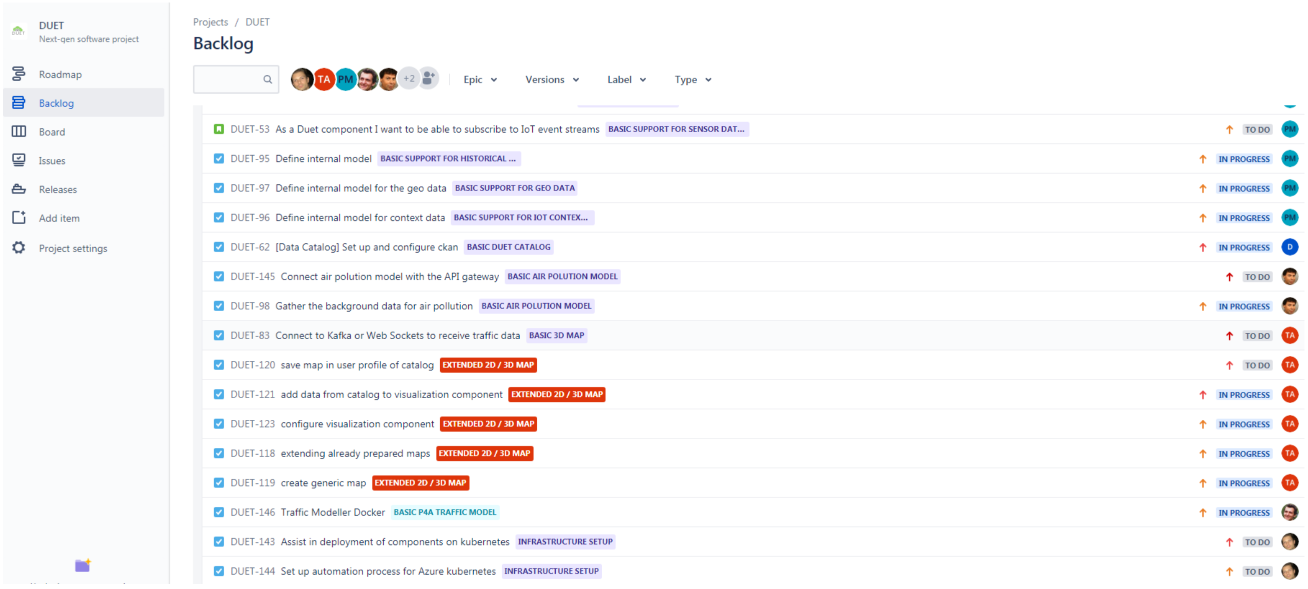


Figure 11:Jira roadmap view



DUET-53	As a Duet component I want to be able to subscribe to IoT event streams	BASIC SUPPORT FOR SENSOR DAT...	TO DO
DUET-95	Define internal model	BASIC SUPPORT FOR HISTORICAL ...	IN PROGRESS
DUET-97	Define internal model for the geo data	BASIC SUPPORT FOR GEO DATA	IN PROGRESS
DUET-96	Define internal model for context data	BASIC SUPPORT FOR IOT CONTEX...	IN PROGRESS
DUET-62	[Data Catalog] Set up and configure ckan	BASIC DUET CATALOG	IN PROGRESS
DUET-145	Connect air polution model with the API gateway	BASIC AIR POLUTION MODEL	TO DO
DUET-98	Gather the background data for air pollution	BASIC AIR POLUTION MODEL	IN PROGRESS
DUET-83	Connect to Kafka or Web Sockets to receive traffic data	BASIC 3D MAP	TO DO
DUET-120	save map in user profile of catalog	EXTENDED 2D / 3D MAP	TO DO
DUET-121	add data from catalog to visualization component	EXTENDED 2D / 3D MAP	IN PROGRESS
DUET-123	configure visualization component	EXTENDED 2D / 3D MAP	IN PROGRESS
DUET-118	extending already prepared maps	EXTENDED 2D / 3D MAP	IN PROGRESS
DUET-119	create generic map	EXTENDED 2D / 3D MAP	IN PROGRESS
DUET-146	Traffic Modeller Docker	BASIC P4A TRAFFIC MODEL	IN PROGRESS
DUET-143	Assist in deployment of components on kubernetes	INFRASTRUCTURE SETUP	TO DO
DUET-144	Set up automation process for Azure kubernetes	INFRASTRUCTURE SETUP	TO DO

Figure 12:Jira backlog view

Deployment

For component isolation and easy deployment, we are going to use Docker [15], that is, all the components developed for the DUET cell, will have to be dockerized.

Docker packages and runs applications in Docker images.
A Docker image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings. Docker images are stored in a Docker registry and in order to run them in virtual or physical machines, the machines need to have installed the Docker Engine software. The running instance of a Docker image is called Docker container. A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another.

For orchestrating and monitoring the Docker containers of the projects we will use Kubernetes [16].
Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications, that groups containers into logical units for easy management and discovery. Some of its features include service topology, discovery and load balancing, secret configuration, self healing, horizontal scaling etc.

The DUET solution will be cloud agnostic, that is, a city at the end of the project will be able to deploy the DUET components, to any cloud provider or cluster of virtual or physical servers, by following the DUET instructions.

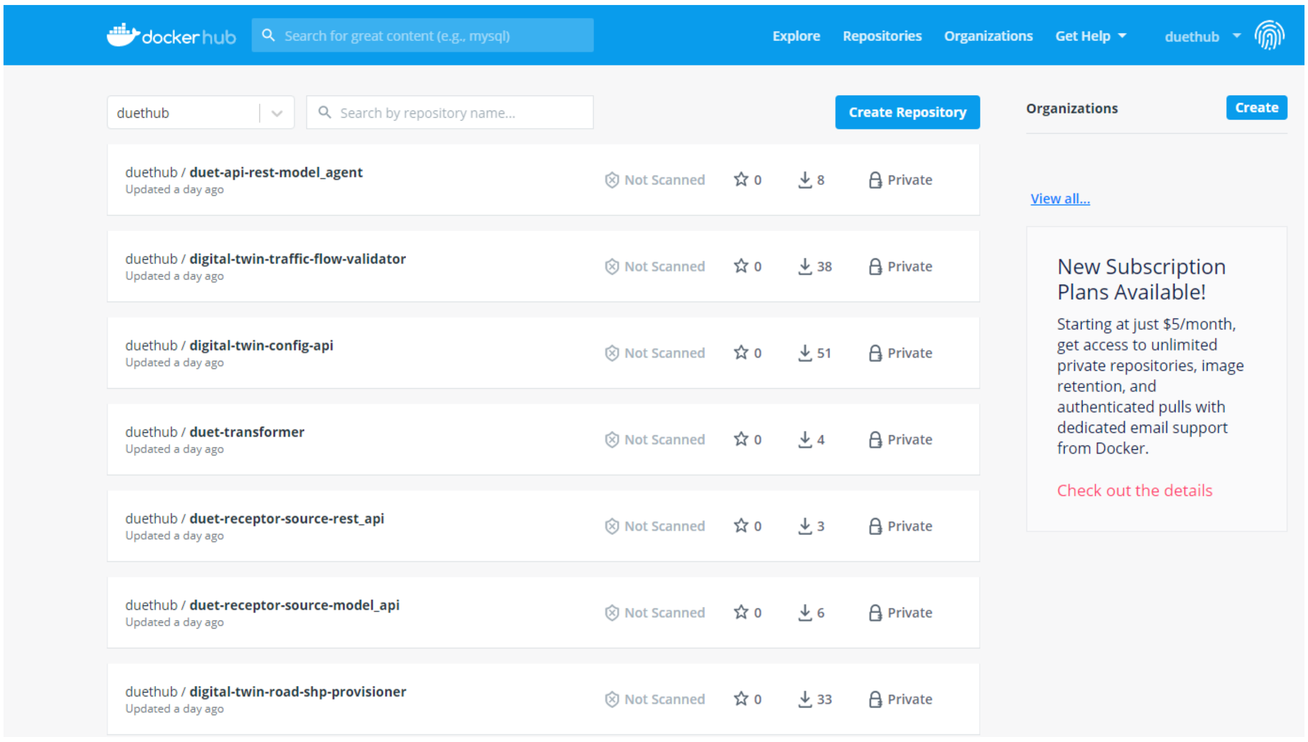
For the purposes of the Alpha and Prototype versions however, we have selected to deploy the DUET components to Microsoft Azure [17]. Azure is a cloud computing service used to easily build, manage and deploy applications. It provides an intuitive user interface and command line tools, to assist in the creation of VMs, Kubernetes clusters, automation processes etc, that will be used in the context of the DUET project.

Repositories

As a repository for the project's open source code, a new group at GitLab [18] was created.
GitLab provides a free private space on cloud, with unlimited repositories and users and also a set of tools like wiki based documentation and issue boards. More importantly Gitlab offers functionality to automate the entire DevOps life cycle, through its GitLab CI/CD pipelines features.

Technical partners that intend to provide their code under commercial licence, are not obliged to use the DUET gitlab repository, however the integration guidelines and quality standards set for all components applies to their own as well.

All components that will be deployed in the DUET platform, have to be dockerized. This means that a private Docker registry for the project is required, so that the deployment of the Docker images can be done in a uniform and easy manner.
To this end, a DockerHub account for the project was created and all the docker images produced will be stored there.



duethub / **duet-api-rest-model_agent** Updated a day ago	Not Scanned	0	8	Private
duethub / **digital-twin-traffic-flow-validator** Updated a day ago	Not Scanned	0	38	Private
duethub / **digital-twin-config-api** Updated a day ago	Not Scanned	0	51	Private
duethub / **duet-transformer** Updated a day ago	Not Scanned	0	4	Private
duethub / **duet-receptor-source-rest_api** Updated a day ago	Not Scanned	0	3	Private
duethub / **duet-receptor-source-model_api** Updated a day ago	Not Scanned	0	6	Private
duethub / **digital-twin-road-shp-provisioner** Updated a day ago	Not Scanned	0	33	Private

New Subscription Plans Available!

Starting at just $5/month, get access to unlimited private repositories, image retention, and authenticated pulls with dedicated email support from Docker.

Check out the details

Figure 13:Docker Hub repositories

Testing and Code Quality

The system validation methodology is described in more detail in Section 6 of the current document. In this subsection we present some of the tools that can be used in order to retrieve the measurements that will be defined based on the methodology metrics.

All components developed for DUET, will include unit and/or integration tests, in order to guarantee a higher level of code quality.

Unit tests are automated tests that check if a small part of the application, known as unit, behaves as it is intended to. In unit testing, any dependencies the unit may have is replaced by "mock" units, that is, units that just return a defined response without implementing any actual functionality.

Integration tests check the behaviour of not only one unit, but a group of units that work together for the completion of a specific functionality. All the dependencies, in this case, including external ones like databases, are real.

Various tools exist for implementing unit and integration tests, in all popular programming languages like:

- JUnit[1], Mockito[2] for Java
- Karma[3], Mocha[4], Chai[5] for Node JS

[1] https://junit.org/junit5/
[2] https://site.mockito.org/
[3] https://karma-runner.github.io/latest/index.html
[4] https://mochajs.org/
[5] https://www.chaijs.com/

- unittest[6], nose[7] for Python
- Jest[8], Mocha for React JS etc

Besides the unit and integration tests, the DUET components will undergo stress tests in order to measure their performance under load. A tool that can be used for this purpose is JMeter[9].
JMeter is designed to load test functional behaviour and measure performance of web applications and web services by defining a set of Web Services Test Plan, which include information like the parameters of the service, the number of concurrent users, the time frame etc.

in order to have a more reliable and globally accepted measure of code quality, for the various quality metrics defined in the validation methodology, the popular SonarQube[10] a quality gateway will be used.

SonarQube is an open-source platform developed for continuous inspection of code quality to perform automatic reviews with static analysis of code to detect bugs, code smells, and security vulnerabilities on more than 20 programming languages. SonarQube offers reports on duplicated code, coding standards, unit tests, code coverage, code complexity, comments, bugs, and security vulnerabilities.

Monitoring

in systems like DUET, it is important to ensure that the different system element services are running smoothly. To his end, the overall performance of the system needs to be constantly monitored and actions to be taken by a system administrator in case of performance degradation.

A first level of monitoring is performed by Azure and Kubernetes, however since DUET needs to be cloud agnostic and in the course of the project more performance metrics may be defined, additional monitoring tools will be deployed in the platform. Some of these tools can be Prometheus[11] and Grafana[12].

Prometheus is an open source tool under Apache License, used for event monitoring and alerting. It records real time metrics and stores them in a time series database. It features functionalities like distributed storage, multiple nodes of graphing and dashboarding support and can collaborate with a wide range of tools like Docker, Kubernetes and Grafana

Grafana is an open source and extendable analytics and interactive visualisation web application, that allows a user to query and visualize data, through a set of charts, graphs and alerts, no matter where this data is stored.

[6] https://docs.python.org/3/library/unittest.html
[7] https://pypi.org/project/nose/
[8] https://jestjs.io/
[9] https://jmeter.apache.org/
[10] https://www.sonarqube.org/
[11] https://prometheus.io/
[12] https://grafana.com/

CI/CD Process

As mentioned earlier, we are going to use the GitLab pipelines feature for Continuous Integration and Deployment.

Pipelines are the top-level component of continuous integration, delivery, and deployment and are composed of Jobs that define what needs to be done and Stages that define when the jobs must run.
The Jobs of each Stage can be executed in parallel while the Stages can only be completed sequentially. If all the Jobs of a Stage complete successfully, then the pipeline proceeds to the next Stage. If a Job fails then the whole pipeline fails.

The pipelines are defined in specific files (gitlab-ci.yml), that are stored in the root folder of the code repository and involve the creation of integration parameters on the administration pages of GitLab.

The DUET code projects will have to run the pipeline depicted in Figure 14 and has the following Stages:

1. Build the code. This can be considered for example the equivalent of mvn build or npm build in Java and Node JS respectively
2. Run the unit and integration tests defined in the code project. If one of the test fails, the pipeline fails
3. Produce the quality metrics and push them to the project's SonarQube for further evaluation
4. Create the Docker image of the component and push it to the relevant DockerHub repository
5. Deploy the component to the project's Kubernetes cluster

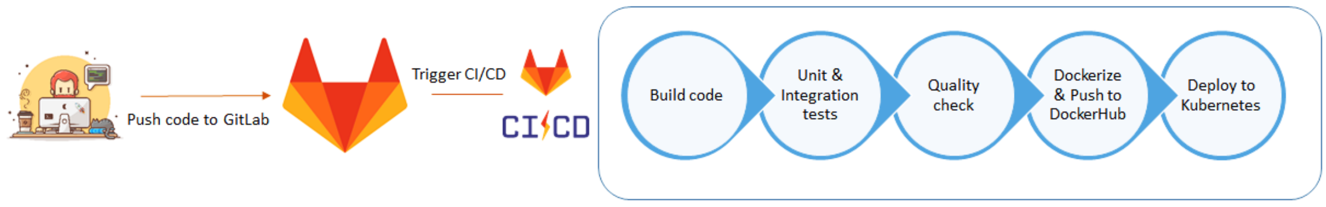


Figure 14:CI/CD pipeline

6. System Validation

In this section we provide an overview of the methodology that will be used for the system validation of the platform. More specifically we present the ISO/IEC 25010:2011 and explain its quality characteristics. Out of these characteristics we will select the most appropriate ones, in order to form the most suitable quality model for the DUET project and perform our validation tests to the final version of the DUET platform.

6.1 Introduction

Software validation is the "confirmation by examination and provision of objective evidence that software specifications conform to user needs and intended uses, and that the particular requirements implemented through software can be consistently fulfilled"[1]. Since software is usually part of a larger system, the validation of software typically includes evidence that all software requirements have been implemented correctly and completely.

In general, software validation is the process of developing a "level of confidence" that the system meets all requirements, functionalities, and user expectations as set out during the design process. It is a critical tool used to assure the quality of its component and the overall system. It allows for improving/refining the end product.

Software validation is realized through quality models. In the past, different quality models have been proposed, each one of which addresses different quality attributes that allow evaluating the developed software. Some of the most well-known are:

McCall's model of software quality (GE Model, 1977), which incorporates 11 criteria encompassing product operation, product revision and product transition.

Boehm's spiral model (1978) based on a wider range of characteristics, which incorporates 19 criteria. The criteria in both this and the GE model, are not independent as they interact with each other and often cause conflicts.

ISO 9126-1 incorporates six quality goals, each goal having a large number of attributes. These six goals are then further split into sub-characteristics, which represent measurable attributes (custom defined for each software product).

6.2 ISO/IEC 25010:2011

Recently, the BS ISO/IEC 25010:2011 standard about system and software quality models has replaced ISO 9126-1. Applying any of the above models is not a straightforward process. There are no automated means for testing software against each of the characteristics defined by each model. For each model, the final attributes must be matched against measurable metrics and thresholds for evaluating the results must be set. It is then possible to measure the results of the tests performed (either quantitative or qualitative/observed).

The ISO/IEC 25010:2011 standard is the most widespread reference model and includes the common software quality characteristics that are supported by the other models. This standard defines two quality models providing a consistent terminology for specifying, measuring and evaluating system and software product quality, as described below.

Quality in use model

The *Quality in use model* is composed of five characteristics that relate to the outcome of interaction with the system and characterizes the impact that the product can have on the stakeholders. It pertains to the notion of *external quality*, i.e. the quality of a (software) product as perceived by its users. External quality assesses the characteristics of the product quality model by *black-box* measurement.

Product quality model

The *Product quality model* is composed of eight characteristics that relate to static properties of software and dynamic properties of the computer system. It is intended to measure the *internal quality*, i.e. the quality of the software (and, particularly, its internal components) that eventually **delivers external quality**. Internal quality assesses the characteristics of the product quality model by *glass-box* measurement, i.e. measuring system properties based on knowledge about the internal structure of the software. For our case, the product quality model is adopted. The eight quality characteristics, are further divided into sub-characteristics, as shown in the following figure:

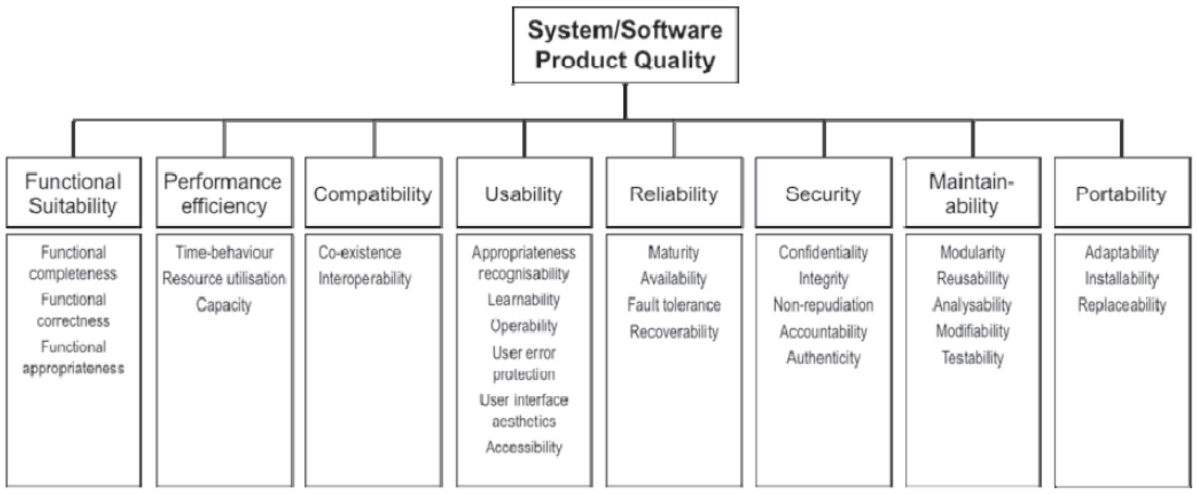


Figure 15:The ISO/IEC 25010:2011 system/software quality model characteristics

Although rather generic, not all of the listed quality characteristics might be applicable for our purpose, so a tailor-made subset could be better suited. For each of the sub-characteristics, a metric/measurable attribute will be defined, along with thresholds. These metrics and thresholds are customized for each software product, which in our case is the DUET platform (consisting of individual components). By evaluating these metrics, we will be able to assess the overall quality of our platform and the percent to which we were able to meet the user and technical requirements (reflected to system specifications and functionalities), defined during the design phase of the project.

6.3 Designing a quality model

As we have seen, a quality model is the cornerstone of a product quality evaluation system. It determines which quality characteristics will be considered when evaluating the properties of a software product.

Understanding the product quality model

The quality of a system is the degree to which the system satisfies the stated and implied needs of its various stakeholders, and thus provides value. Those stakeholders' needs are precisely what is represented in the quality model, which categorizes the product quality into characteristics and sub-characteristics, as defined below.

Functional suitability

This characteristic represents the degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions. This characteristic is composed of the following sub characteristics:

- **Functional completeness.** Degree to which the set of functions covers all the specified tasks and user objectives.
- **Functional correctness.** Degree to which a product or system provides the correct results with the needed degree of precision.
- **Functional appropriateness.** Degree to which the functions facilitate the accomplishment of specified tasks and objectives.

Performance efficiency

This characteristic represents the performance relative to the amount of resources used under stated conditions. This characteristic is composed of the following sub characteristics:

- **Time behavior.** Degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements.
- **Resource utilization.** Degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements.
- **Capacity.** Degree to which the maximum limits of a product or system parameter meet requirements.

Compatibility

This is the degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment. This characteristic is composed of the following sub characteristics:

- **Co-existence.** Degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product.

- **Interoperability.** Degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.

Usability

This characteristic represents the degree to which a product or system can be used by specified users to achieve specific goals with effectiveness, efficiency and satisfaction in a specified context of use. This characteristic is composed of the following sub characteristics:

- **Appropriateness recognizability.** Degree to which users can recognize whether a product or system is appropriate for their needs.
- **Learnability.** degree to which a product or system can be used by specified users to achieve specific goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use.
- **Operability.** Degree to which a product or system has attributes that make it easy to operate and control.
- **User error protection.** Degree to which a system protects users against making errors.
- **User interface aesthetics.** Degree to which a user interface enables pleasing and satisfying interaction for the user.
- **Accessibility.** Degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.

Security

This is the degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization. This characteristic is composed of the following sub characteristics:

- **Confidentiality.** Degree to which a product or system ensures that data are accessible only to those authorized to have access.
- **Integrity.** Degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data.
- **Non-repudiation.** Degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later.
- **Accountability.** Degree to which the actions of an entity can be traced uniquely to the entity.
- **Authenticity.** Degree to which the identity of a subject or resource can be proved to be the one claimed.

Maintainability

This characteristic represents the degree of effectiveness and efficiency with which a product or system can be modified to improve it, correct it or adapt it to changes in environment, and in requirements. This characteristic is composed of the following sub characteristics:

- **Modularity.** Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components.
- **Reusability.** Degree to which an asset can be used in more than one system, or in building other assets.
- **Analyzability.** Degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified.
- **Modifiability.** Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality.
- **Testability.** Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.

Reliability

This is the degree to which a system, product or component performs specific functions under specified conditions for a certain period. This characteristic is composed of the following sub characteristics:

- **Maturity.** Degree to which a system, product or component meets needs for reliability under normal operation.
- **Availability.** Degree to which a system, product or component is operational and accessible when required for use.
- **Fault tolerance.** Degree to which a system, product or component operates as intended despite the presence of hardware or software faults.
- **Recoverability.** Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.

Portability

Portability is the degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another. This characteristic is composed of the following sub characteristics:

- **Adaptability.** Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments.
- **Installability.** Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment.
- **Replaceability.** Degree to which a product can replace another specified software product for the same purpose in the same environment.

7. Alpha version prototype

The purpose of the alpha version prototype is to provide a first version of key components that collaborate in order to realize the initial set of epics as these are described in the current document. These components validate the DUET architecture as they demonstrate main data flows and integrations in the system:

- The flow of data from the IoT sources to the Message Streaming Platform and from the M.S.P to the UIs through web sockets
- The flow of data from the models through the App Gateway and then to the UIs again through web sockets
- The usage of the Data Catalog from the UIs
- The data integration of traffic, noise pollution, air quality, 2D/3D etc information on the UIs.

The components implemented and the communication between them is depicted in Figure 16.

More specifically:

- Regarding the infrastructure, the Gitlab and DockerHUb repositories are set up, the Kubernetes cluster is set up on Azure and all components, except the UIs, are deployed in the cluster.
- A first version of CityFlows is implemented that sends IoT data to the platform.
- The first version of the Data Sources Gateway is implemented that involves the Iot Time Series and Event Connectors. The connectors receive data from CityFlows and Telraam and send them to the Message Streaming Platform (Kafka).
- The first versions of the traffic models provided by KUL and P4A, as well as the air quality model from TNO are implemented and provide data to the App Gateway.
- The App Gateway (Receiver), accepts messages from the models and sends them to Kafka. The Sender part, listens to the relevant model topics and pushes the data to the equivalent web socket servers.
- Regarding the user interfaces, there is one UI per pilot and the functionalities implemented are summarised as follows:
    - The user can view a 2D or 3D map of the three pilot cities
    - The user can view Points of Interest (POIs) on the maps
    - The user can select to view traffic or noise data on the map (Flanders and Pilsen pilots)
    - The user can access the Data catalog and view the available data sources
    - The user can apply specific UI settings

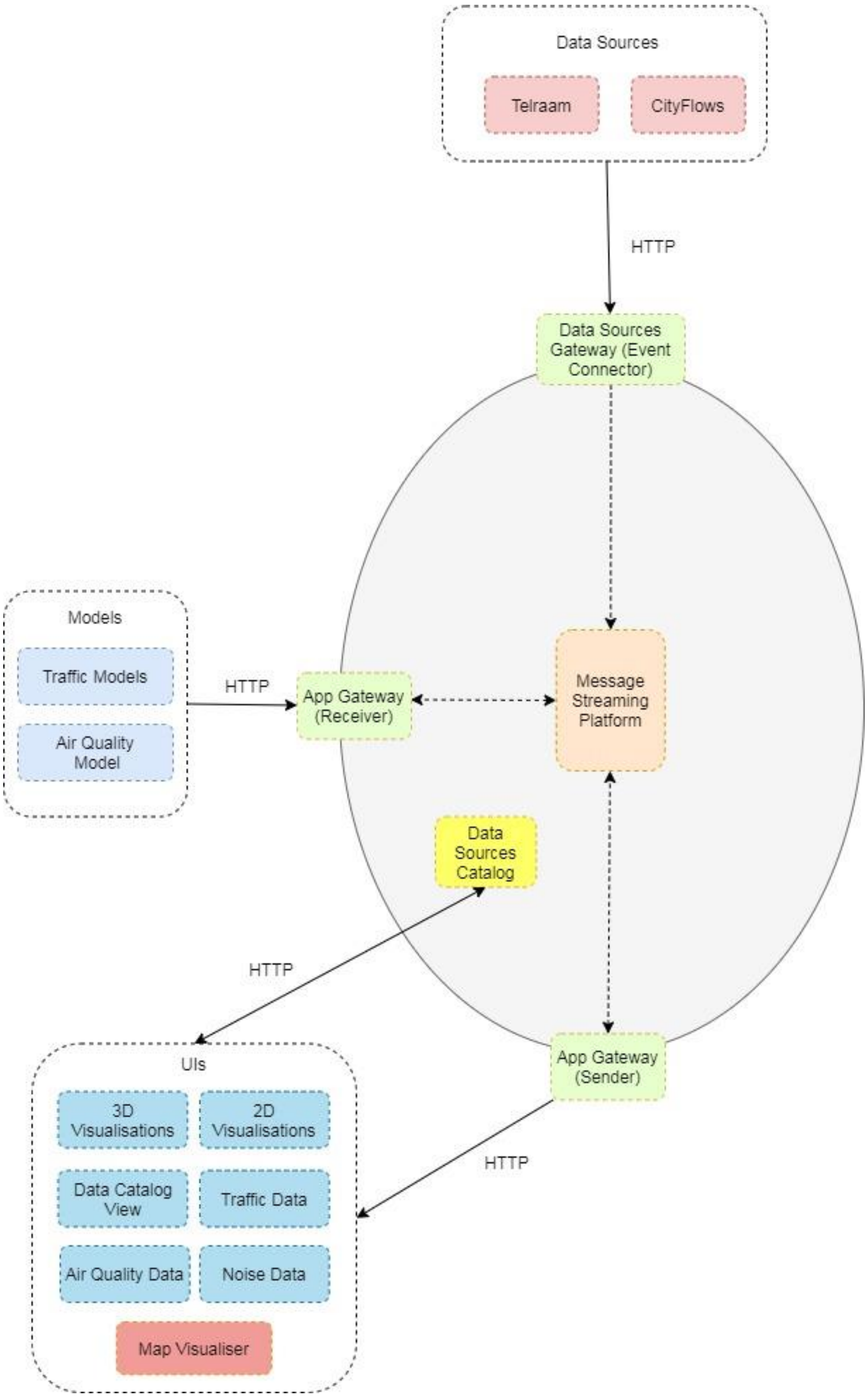


Figure 16:Alpha version components communication

The user interfaces developed for the alpha version can be found at the following web links.

Flanders pilot	https://duet.virtualcitymap.de/flanders/#/
Athens pilot	https://duet.virtualcitymap.de/athens/#/
Plzen pilot	https://duet.virtualcitymap.de/plzen/#/
Data catalog and consolidated view	https://duet.virtualcitymap.de/catalog/#/
Use the following credentials: **Username**: duet **Password** : flanders2020	

In the following paragraphs we provide some screenshots from the UI applications along with a quick guide on how to use their features. Most of the functionalities are the same, so we describe them using the Plzen pilot, providing the necessary screenshots when there are differences.

After signing in, the first page presented to the user is the 3D map representation of the city, with some default data applied.

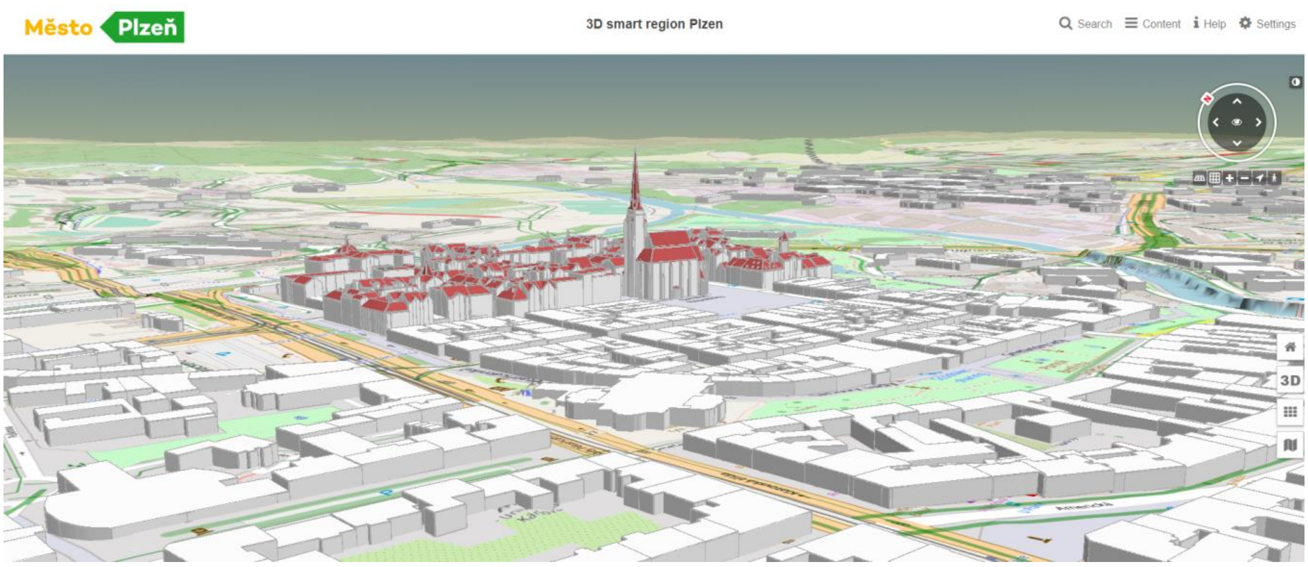


Figure 17:3D Plzen map

By clicking on the "Content" page, the user is able to view a list of available data that can be added to the map.

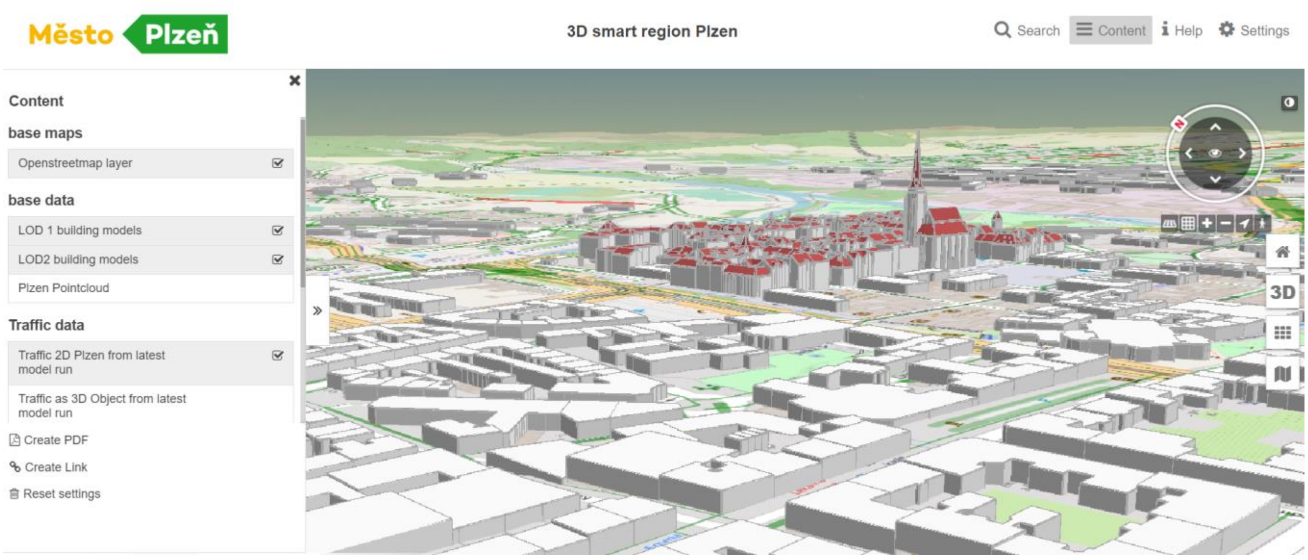


Figure 18:Plzen content menu

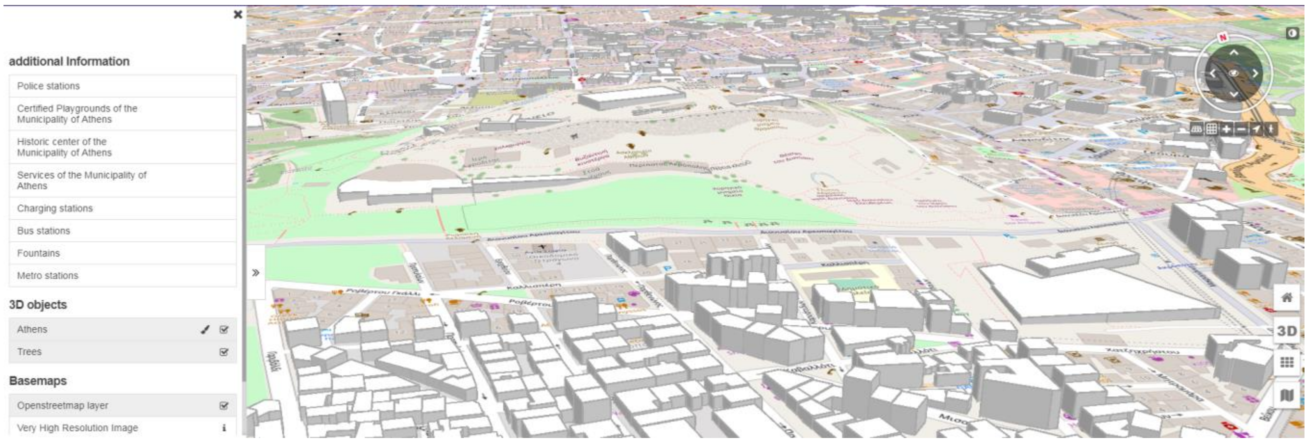

additional Information
Police stations
Certified Playgrounds of the Municipality of Athens
Historic center of the Municipality of Athens
Services of the Municipality of Athens
Charging stations
Bus stations
Fountains
Metro stations
3D objects
Athens
Trees
Basemaps
Openstreetmap layer
Very High Resolution Image
3D

Figure 19:Athens content menu

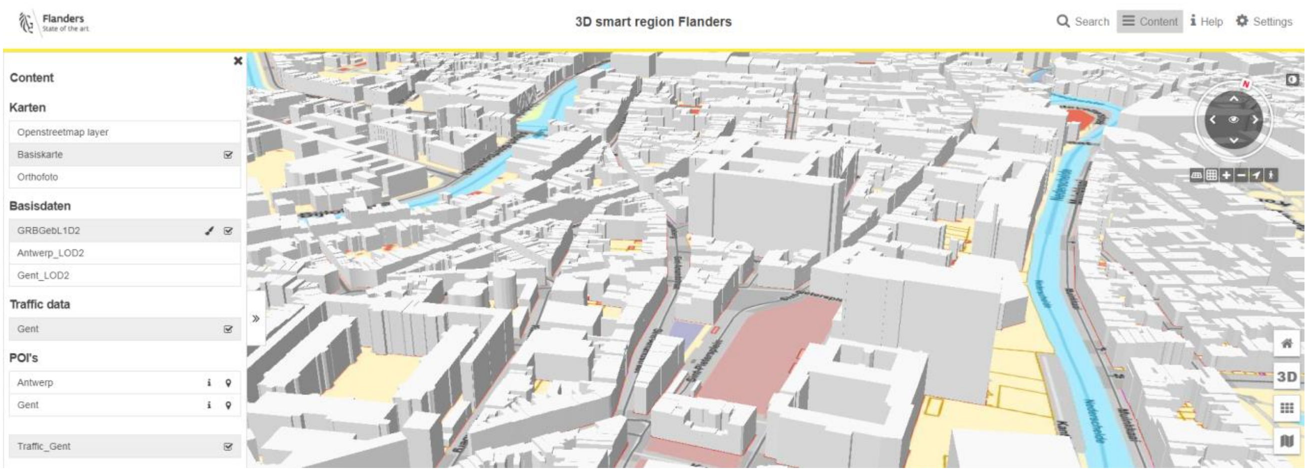


Figure 20:Flanders content menu

For Plzen, the available data are base data that are related to the building in the area, traffic data and noise data, for Athens there are available POIs and buildings and for Flanders buildings, POIs and traffic data.

By selecting / deselecting the data from the menu, this is added or removed from the map. For example if the user clicks on the “traffic sensors” and the “Traffic as 3D Object from latest model run” the data will be added to the map as depicted in Figure 21, will select the “Noise as PointCloud” as depicted in Figure 22**.**

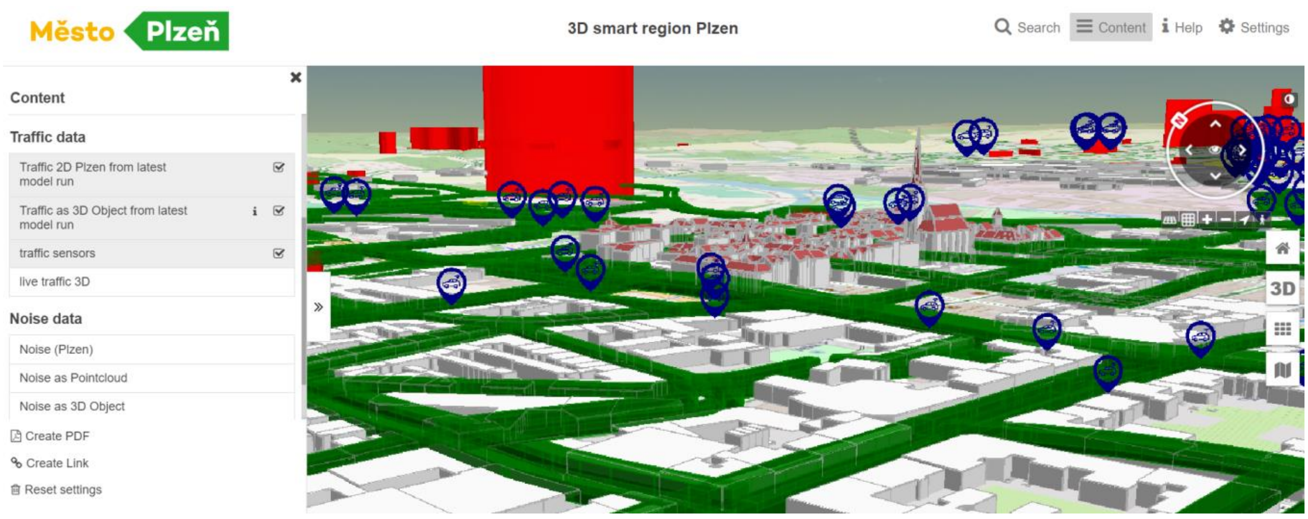


Figure 21:Traffic data and traffic sensors on 3D Plzen map

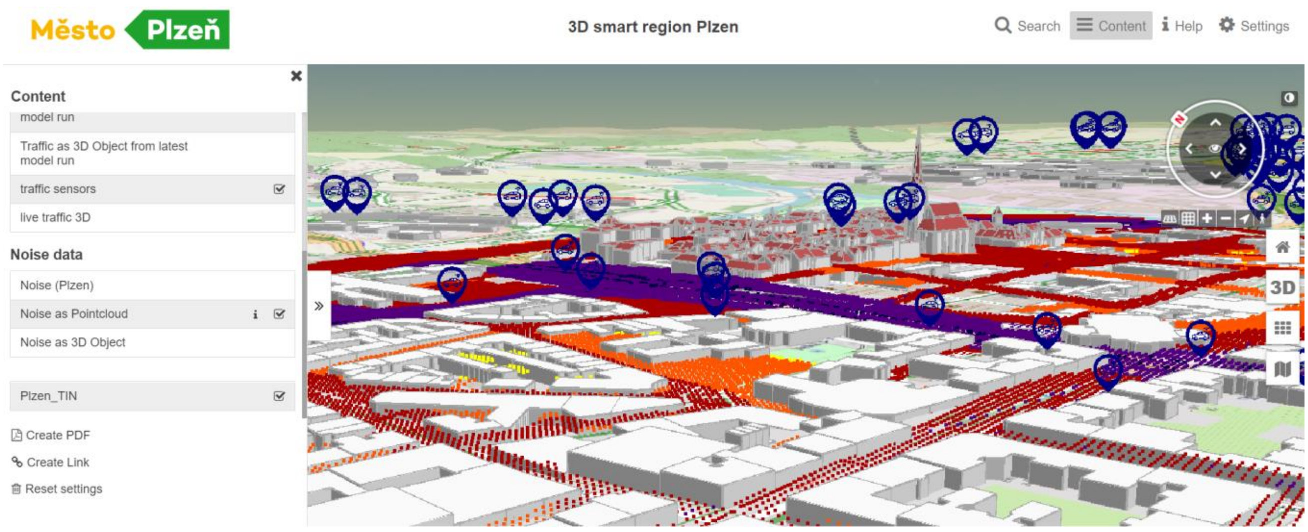
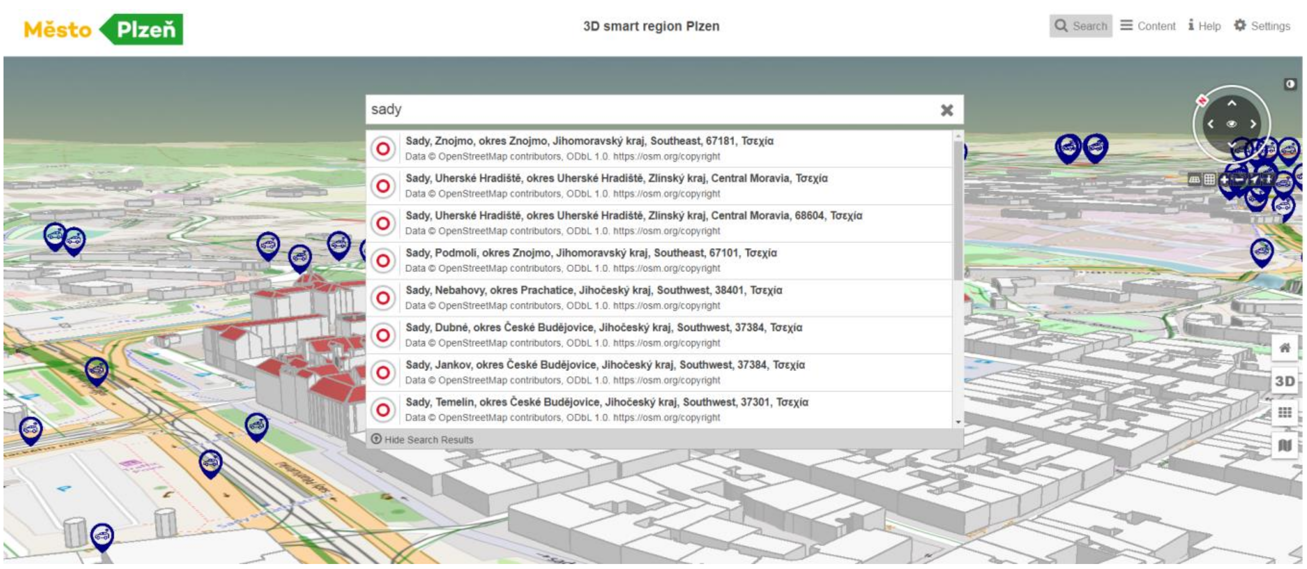


Figure 22:Noise data and traffic sensors on 3D Plzen map

The user can search for a specific POI or street on the map, by clicking “Search” and entering the name of the desired location. Clicking on one of the results, redirects the user to the specific location.



sady

Sady, Znojmo, okres Znojmo, Jihomoravský kraj, Southeast, 67181, Τσεχία
Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright

Sady, Uherské Hradiště, okres Uherské Hradiště, Zlínský kraj, Central Moravia, Τσεχία
Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright

Sady, Uherské Hradiště, okres Uherské Hradiště, Zlínský kraj, Central Moravia, 68604, Τσεχία
Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright

Sady, Podmolí, okres Znojmo, Jihomoravský kraj, Southeast, 67101, Τσεχία
Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright

Sady, Nebahovy, okres Prachatice, Jihočeský kraj, Southwest, 38401, Τσεχία
Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright

Sady, Dubné, okres České Budějovice, Jihočeský kraj, Southwest, 37384, Τσεχία
Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright

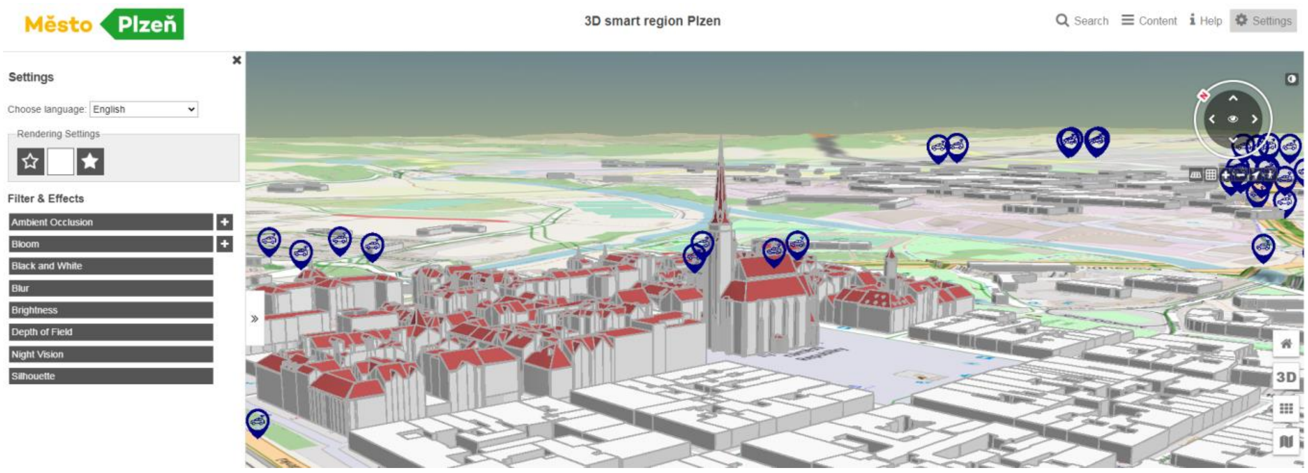
Sady, Jankov, okres České Budějovice, Jihočeský kraj, Southwest, 37384, Τσεχία
Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright

Sady, Temelín, okres České Budějovice, Jihočeský kraj, Southwest, 37301, Τσεχία
Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright

Hide Search Results

Figure 23:Search on the map

The user can change the settings of the maps, through the "Settings" page. More specifically, the user can change the language of the map by selecting an available language from the list, the quality of the map by clicking on one of the rendering settings buttons and apply certain filters and effects like brightness, blur etc



Choose language: English

Figure 24:Settings page

On the bottom right of the map, there is a menu where the user can:

- go to the original location when the map loaded, by clicking the "House" button
- select 2D or 3D representation of the city, by clicking the 2D or 3D button
- view a snippet of the map for the current location by clicking on the "Wave" button
- and perform additional actions on the map like, change the viewpoint, take a measurement or export to a pdf file

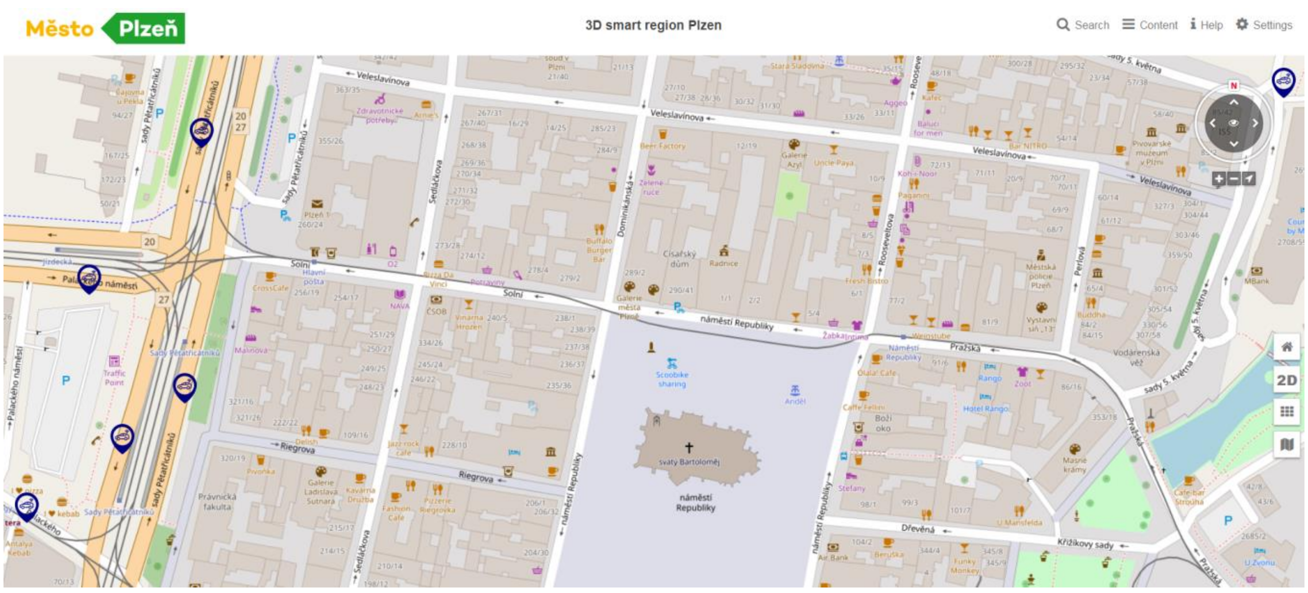


Figure 25:Plzen 2D map

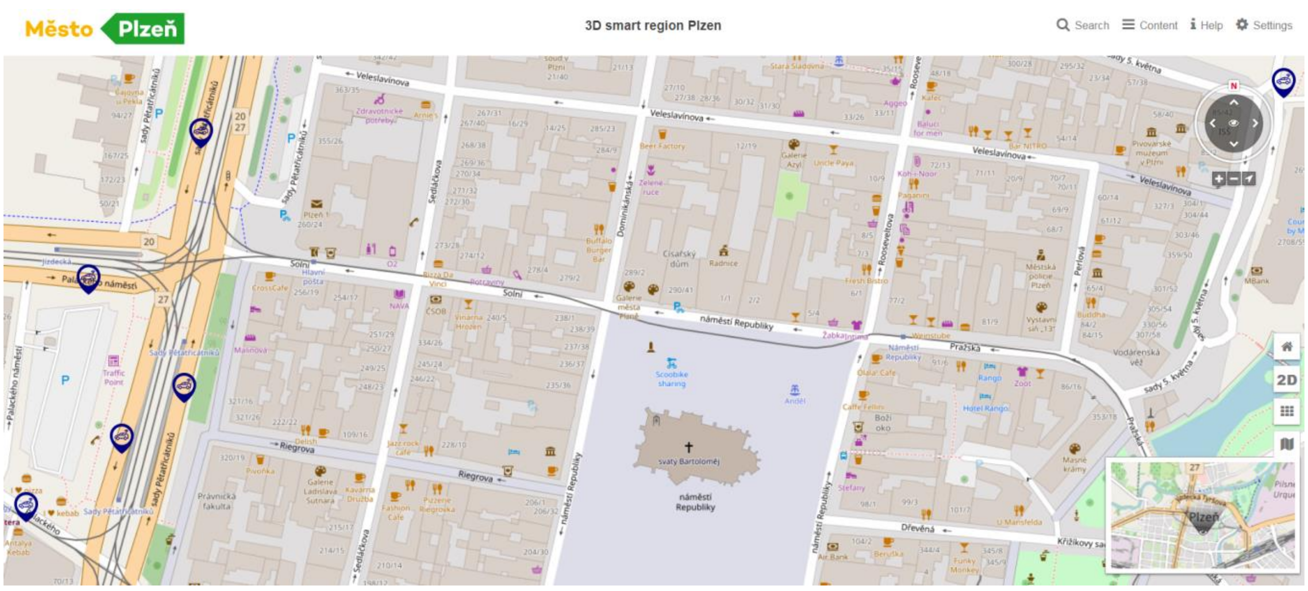


Figure 26: Plzen map snippet

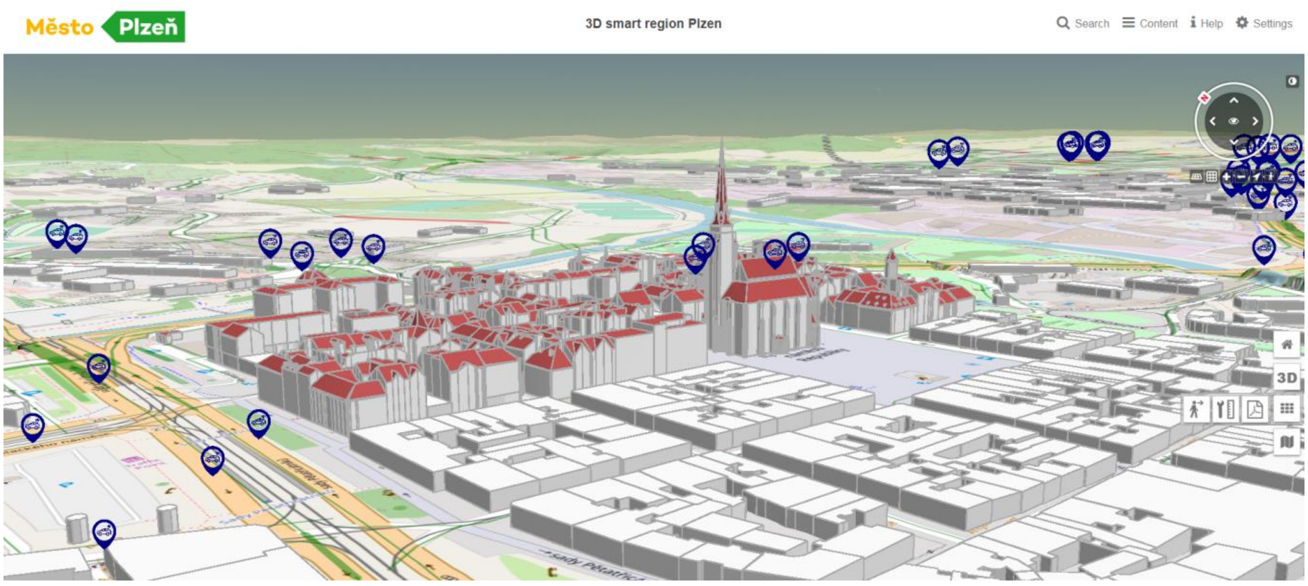


Figure 27:: Plzen context menu

For changing the view of the map, the user clicks on the pedestrian button, then clicks once on the map to set a starting point and again to set the direction of the view.

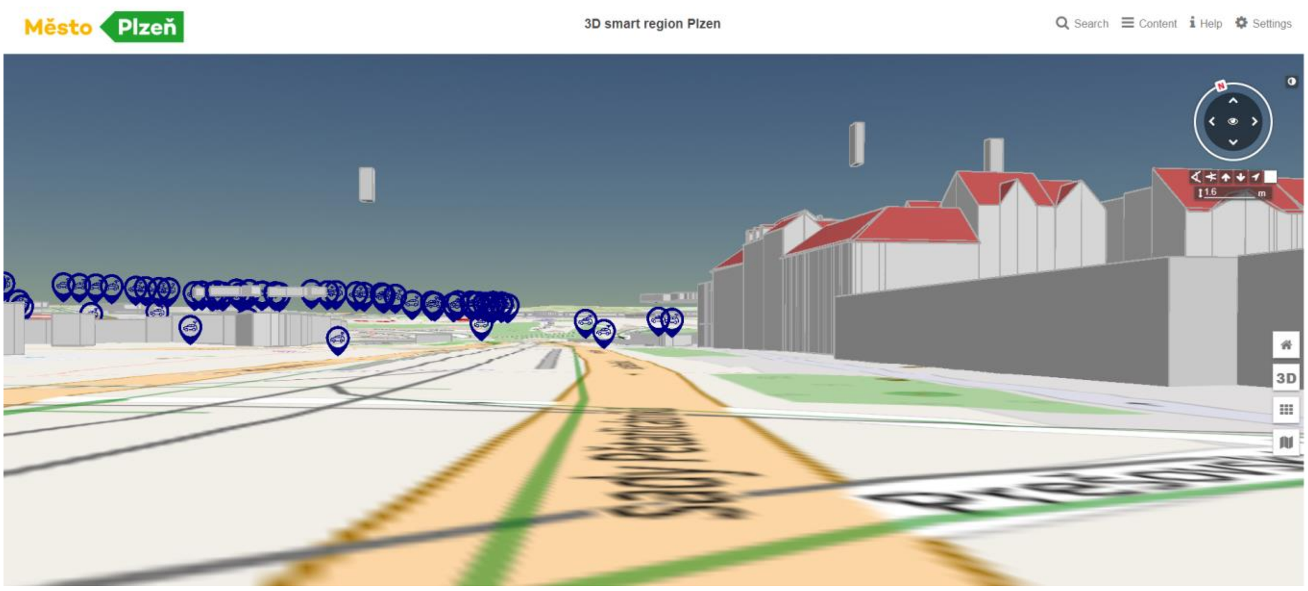
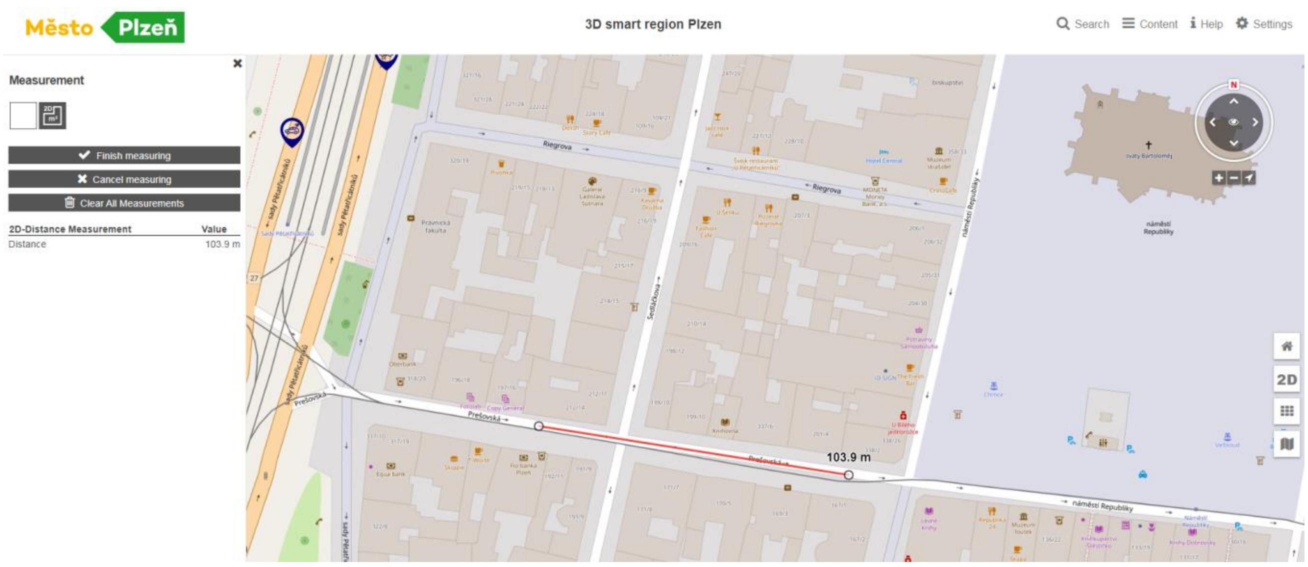


Figure 28: Pedestrian view

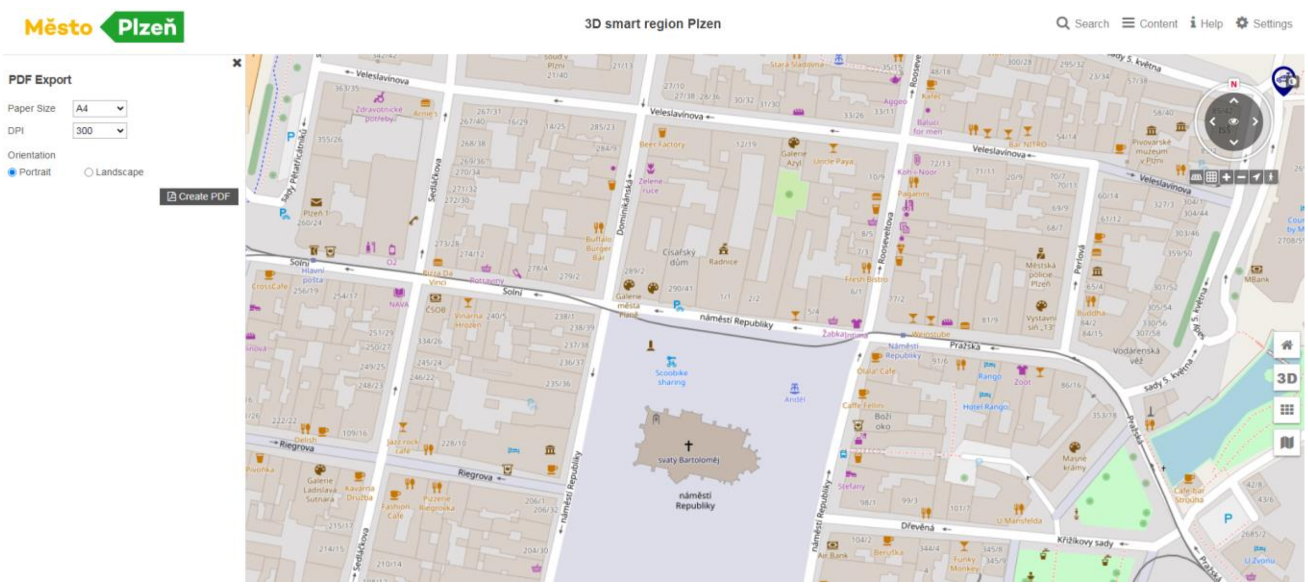
For getting a measurement on the map, the user selects the “wrench” button, then once on the map to set the starting point and then once more to set the ending point. To finalize or cancel the measurement, the user clicks on the relevant button on the left of the page.



2D-Distance Measurement	Value
Distance	103.9 m

Figure 29:Measurement on 2D map

For exporting the image to a pdf file, the user clicks on the "pdf" button of the context menu, selects paper size, analysis and orientation on the menu on the left of the map and clicks "Create pdf".



Paper Size: A4
DPI: 300
Orientation: [✓ Portrait] Landscape

Figure 30:Pdf export

8. General plan for the implementation of the DUET system

Following the general plan for the execution of the DUET project, which has been listed on the Description of Action of the DUET Grant Agreement, in this section we present the plan for the delivery of the DUET system environment.

More specifically, the integration of the DUET system lasts until the end of the project (M36), exhibiting an interim release of the integrated environment by M18. Apart from these two release dates for the integrated platform, we consider at least two intermediate internal releases by M24 and M30. This is a critical turning point from the development and testing phase to the operational phase of the project, since, by M24, almost all the DUET components will have been delivered in their final version and the system should have been customised and released for intense piloting from the pilot partners (Flanders region, City of Athens, City of Pilsen).
Thus, the DUET implementation plan distinguishes between four major milestones for the go-live scenarios:

- **Milestone 1: Alpha version of the DUET platform available for internal testing (M12)**

Scope: To provide an early version of the DUET platform that contains the features as detailed in the first selection of epics. It works as a proof of concept of what we want to implement within DUET.

Outcome: In this milestone, we emphasize on a set of DUET capabilities that enable the end users to see a 2D and 3D map of the area of interest and also load data from the data catalog. The main functions that are addressed in this milestone are those, which are described in Section 7.

Components	Date
Messaging Streaming platform (Kafka) with initial topics	M12 (Nov 2020) / completed
Data sources gateway	
App gateway	
Data sources catalog (CKAN)	
Initial traffic models	
Initial air quality model	
Map visualiser	
3D / 2D visualisations	
UI initial integration of traffic, air pollution and noise data	
UI initial integration of Data catalog	

- **Milestone 2: Solution release Closed Beta available for Closed User Group (M18)**

Scope: To provide a working prototype of the DUET capabilities that will be used in the piloting phase of the project. Each pilot will test their individual Twins with a small number of colleagues to ensure the Digital Twins work as planned.

Outcome: In this milestone, we emphasise on a set of the DUET capabilities that enable the end users to use the available functionalities.

Components	Date
Messaging Streaming platform (Kafka) with updated topics	M18 (May 2021)
Updated Data sources gateway	
Updated App gateway	
Initial API Gateway	
Updated Data catalog (Data sources, Data mapping)	
Initial Knowledge Graph with initial ontologies	
initial Management (User management, Data Access, Audit Manager)	
Initial platform UIs	
Initial Visualisation Manager	
Traffic models updated	
Air quality model updated	
Noise model updated	
Monitoring tools set up	
Initial security measures applied	

- **Milestone 3: Solution Release Open Beta available for Open User Group (M24)**

Scope: To fine tune the DUET platform, following the user validation in three pilot cases. This version will be tested by external users.

Outcome: In this milestone, we emphasise on the calibration of the DUET functions, according to the feedback resulting from the three pilot scenarios.

Components	Date
Messaging Streaming platform (Kafka) with updated topics	M24 (Nov 2021)
Updated Data sources gateway	

Updated App gateway	
Updated API Gateway	
Updated Data catalog (Data sources, Data mapping, Models catalog)	
Updated Knowledge Graph with refined ontologies	
Updated Management component (User management, Data Access, Audits Manager, Notifications Manager, Configurations Manager)	
Updated Platform UIs	
Updated Visualisation Manager	
Initial Streaming Manager	
Initial Models Orchestrator	
Traffic models updated	
Air quality model updated	
Noise model updated	

- **Milestone 4: Solution Release Candidate for sister user group (M30)**

Scope: Data models and APIs will be exchanged between the pilot cities to explore the concept of Policy Ready Data as a Service and test scalability and transferability of the data models between the Twins.

Outcome: In this milestone, we emphasise on the refinements of the DUET functions according to the feedback resulting from the evaluation that will be made by the representatives of the pilot cities.

Components	Date
Messaging Streaming platform (Kafka) with final topics	M30 (May 2022)
Updated Data sources gateway	
Updated App gateway	
Updated API Gateway	
Updated Data catalog	
Updated Knowledge Graph	
Updated Management component	
Updated Visualisation Manager	

Updated Streaming Manager	
Initial Machine Learning	
Gamification Engine	
Updated Platform UIs	
Traffic models updated	
Air quality model updated	
Noise model updated	

The final version of the DUET integrated prototype will be ready by November 2022. Technical partners will make sure that the Platform is functional and resolve any bugs identified during the pilots.
The system technological modules will evolve throughout the project and we need to make sure that it is consistent with design and implementation work in the other technical work packages and the pilot activities of the project.

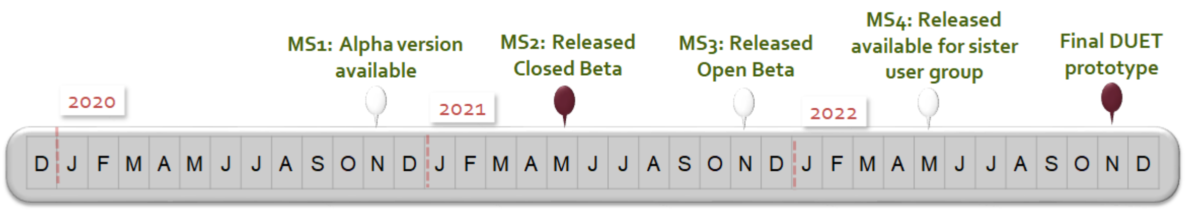

MS1: Alpha version available
MS2: Released Closed Beta
MS3: Released Open Beta
MS4: Released available for sister user group
Final DUET prototype
2020
2021
2022
D J F M A M J J A S O N D J F M A M J J A S O N D J F M A M J J A S O N D

Figure 31:Milestones

9. Conclusion

The current document includes the architecture specifications and design of the integrated DUET platform and serves as the basis for the development tasks of the project. Information about the functionalities from the system point of view the characteristics of the components of the system and the data flow between them is presented in detail.

This architecture description document will be very useful to define and communicate the initial blueprint of the DUET platform. The architecture will continue to evolve throughout the project and the most important point is to make sure that it is consistent and in line with the design and implementation work being described in the other technical work packages, as well as with the early pilot activities of the project. This deliverable acts as the reference point for the actual development of this platform and offers a shared and common background for the Consortium participants on the envisaged technologies that are necessary to build such a platform.

10. References

[1] DUET project, Deliverable D2.3: “Final list of user requirements for the DUET solution”, July 2020

[2] DUET project, Deliverable D2.2: “Scenario specifications of the DUET solution, June 2020

[3] DUET project, Deliverable D8.1: “Project Vision”, March 2020

[4] Agile Project Management: Creating Innovative Products. Jim Highsmith. 2009. Addison-Wesley Professional

[5] DUET project, Deliverable D3.10: “Multi Layered security model specification (by design)”, Nov 2020

[6] DUET project, Deliverable D3.8: “Digital Twin data broker specifications and tools v2”, Nov 2020

[7] DUET project, Deliverable D4.1: “D4.1 Frontend mockups”, Sep 2020

[8] General Principles of Software Validation; Final Guidance for Industry and FDA Staff, January 11, 2002

[9] https://restfulapi.net/http-status-codes/

[10] https://restfulapi.net/http-methods/

[11] https://www.atlassian.com/software/jira

[12] https://kafka.apache.org/

[13] http://en.wikipedia.org/wiki/Software_development_methodology

[14] http://agilemanifesto.org/

[15] https://www.docker.com/

[16] https://kubernetes.io/

[17] https://azure.microsoft.com/en-us /

[18] https://about.gitlab.com/

11. Annex 1: Technical Questionnaire

The following questionnaire was distributed to the technical partners of the platform in order to gather information about the components and tools that will be used to compose the DUET platform.

DUET Technical Questionnaire

INTRODUCTION

The following questions are requested to be filled by DUET's technology partners who are going to develop the components in RTD WPs and that will be integrated in the DUET platform. Please provide your answers and you are kindly requested not to respond with a simple yes or no. Try to elaborate on your answers by giving an example of use, to the extent that this is possible.

QUESTIONS

1. Questions about your component
   1.1. Please name your component and provide a short description of functionality you plan to deliver in the project and any foreseen interrelation with another DUET software component.

   Component Name:

   Component Description:

   1.2. Do you have an existing background tool that will be adopted in the project to support your component? If yes, please provide the following:
   i. Any past project that was used:
   ii. The component architecture and technologies used in the development:

   1.3. Did/Will you implement your component from scratch or is it an extension of a third party platform? If you have extended a third party platform please provide at least the following:
   i. Name:
   ii. Version:

   1.4. Does your component support / require user interaction? If yes, please provide a short description of the expected user feedback.

   1.5. Have you ever performed a stress test on your component under heavy load? If yes, please provide the test results.

   1.6. Which person in your team is the designated component "owner" (for communication purposes)?

   1.7. If known, please state the names of people actively involved in the development of the component (for communication purposes).

   1.8. Will the component be provided as a service, binary or source code?

   1.9. What will be the license of the delivered component?
   i. Are there any restrictions in its use?
   ii. Will it be made available under an open source license?

2. Development Needs
    - 2.1. In which programming languages does your component depend (i.e. java, C++, etc.)? Please provide at least the following:
        - i. Name:
        - ii. Version:
    - 2.2. Does your component depend on existing design tools (i.e. eclipse, process modeling tool, etc.)? Please provide at least the following:
        - i. Name:
        - ii. Version:
    - 2.3. Which are the hardware requirements for your component to run (i.e. any restrictions in memory use, required CPU, etc.)?
    - 2.4. Which are the software requirements for your component to run (i.e. operating system, tomcat, apache, etc.)
    - 2.5. Does your component have any third party dependencies or use third party libraries? If yes, please provide at least the following:
        - i. Name:
        - ii. Version:
    - 2.6. Can you support automatic builds? Can you support Ant or Maven?
    - 2.7. Does your component have any database requirements? If yes please provide the following:
        - i. required design (i.e. SQL-like, noSQL):
        - ii. database schema (i.e. ER diagram, JSON format):
    - 2.8. Is there a requirement on a specific Version Control System?
    - 2.9. Are you going to use a specific tool for the technical verification and evaluation of your component? If yes, which is the tool?

3. Interoperability Requirements
    - 3.1. Does your component offer an API to work as a service?
        - i. **If yes**, please provide the full API Documentation for both input and output streams:
        - ii. **If no**, please describe the communication schema supported by your component:
    - 3.2. Can your component support asynchronous messaging[13]?
    - 3.3. Does your component need any specific data transformation schema[14]?
    - 3.4. Which are the foreseen dependencies with other DUET software components, if any? Please provide at least the following:
        - i. List of components:
        - ii. Input and / or output requirements:
        - iii. Type and aim of the information that need to be exchanged:

4. Security principles
    - 4.1. Which are the security principles (ie. Principle of Least privilege, Establish secure defaults, etc.) that should be taken into consideration to protect your component's data?

[13] asynchronous messaging= when your component places a message in a message queue and does not need to wait for a reply to continue processing

[14] data transformation schema= the process of creating a correspondence between records and fields of a source schema to records and fields in a destination schema