Deliverable D6.5

D6.5 Cross-CPP Methodology

WP 6

Project Acronym & Number:	Cross-CPP – GA 780167
Project Title:	Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources
Funding Scheme:	Innovation Action
Start date of the project:	01.12.2017
Duration:	36

Status:	Final
Document lead:	ATB
Contributors:	All project partners
Document Identifier:	Cross-CPP_D6.5_v101
Date:	03.05.2021
Revision:	101
Project website address:	https://cross-cpp.eu/

Project Summary

The objective is to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research is to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project will develop: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to provide enhanced local weather forecast and predict and optimise energy consumptions in households. The project will build upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, will be used as a basis for further development aiming to extend it to integrated, cross-sectorial data streams analytics.

Project Consortium

- Institut für angewandte Systemtechnik Bremen GmbH (ATB), Germany
- Volkswagen AG (VW), Germany
- Siemens SRO (SIM), Czech Republic
- Meteologix AG (ML), Switzerland
- ATOS Spain SA (ATOS), Spain
- X/Open Company Limited (TOG), United Kingdom
- Universidad Politecnica de Madrid (UPM), Spain
- Vysoke Uceni Technicke V BRNE (BUT), Czech Republic

More Information

ATB Institut für angewandte Systemtechnik Bremen GmbH (Coordinator)
Represented for the purposes of signing the Agreement by: Christian Wolff
E-Mail: wolff@atb-bremen.de
Phone: +49-(0)421 / 22092 33

Institut für angewandte Systemtechnik Bremen GmbH, ATB
HRB 13969 HB
Wiener Straße 1
28359 Bremen
Germany
Web: www.atb-bremen.de

Dissemination Level

PU	Public	X
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Change History

Version	Notes	Date
001	Creation of the document	02.06.2020
002-009	Update structure and inputs	-/-
100	Final Version	04.12.2020
101	Revision of the Annex	03.05.2021

Document Summary

The deliverable D6.5 “Cross-CPP Methodology” provides an overview of developed set of support material, which serve the different stakeholders of the Cross CPP value chain as basic guidelines for participating in the overall Cross-CPP workflow.

One of the key objectives of the support material is to empower Cross-CPP ecosystem stakeholders to easily implement specified and developed procedures and tools, as well as to understand offered functions and tools. Thus, the developed support materials will address organisational, administrative and contractual measures concerning the interaction of the various stakeholders with the Cross-CPP Ecosystem.

The developed support material is of large scale (9 documents with overall 466 pages), anyway the material is included as appendix to this document (D6.5). Given the fact that all support materials are of public nature, D6.5 also includes the hyperlinks wherefrom to download all documents.

Abbreviations

CIDM	Common Industrial Data Model
CVIM	Common Vehicle Information Model
CPP	Cyber physical product
D	Deliverable
etc.	Et cetera
H2020	Horizon 2020 Research and Innovation programme
MP	Marketplace
OEM	Original Equipment Manufacturer
RTD	Research and Technological Development
SDK	Software Development Kit
SP	Service Provider
SW	Software
WP	Work Package

Table of Contents

List of Figures

List of Tables

1 Cross-CPP Methodology Framework

1.1 Introduction

Beside the specification and implementation of the Cross-CPP ecosystem (Marketplace with Data Analytic Toolbox, CIDM etc.) the project has developed and provided a set of support material, which serve the different stakeholders of the Cross CPP value chain (data providers, data consumers, CPP owners, etc.) as basic guidelines for participating in the overall Cross-CPP workflow.

In this context, one of the key objectives of the support material is to empower Cross-CPP ecosystem stakeholders to

- easily implement specified and developed procedures and tools and
- to understand offered functions and tools.

The developed support materials will also address organisational, administrative and contractual measures concerning the interaction of the various stakeholders with the Cross-CPP Ecosystem.

Taking the overall Cross-CPP workflow with its three main pillars as a baseline, the key stakeholders can be mapped to this workflow as presented in Figure 1.

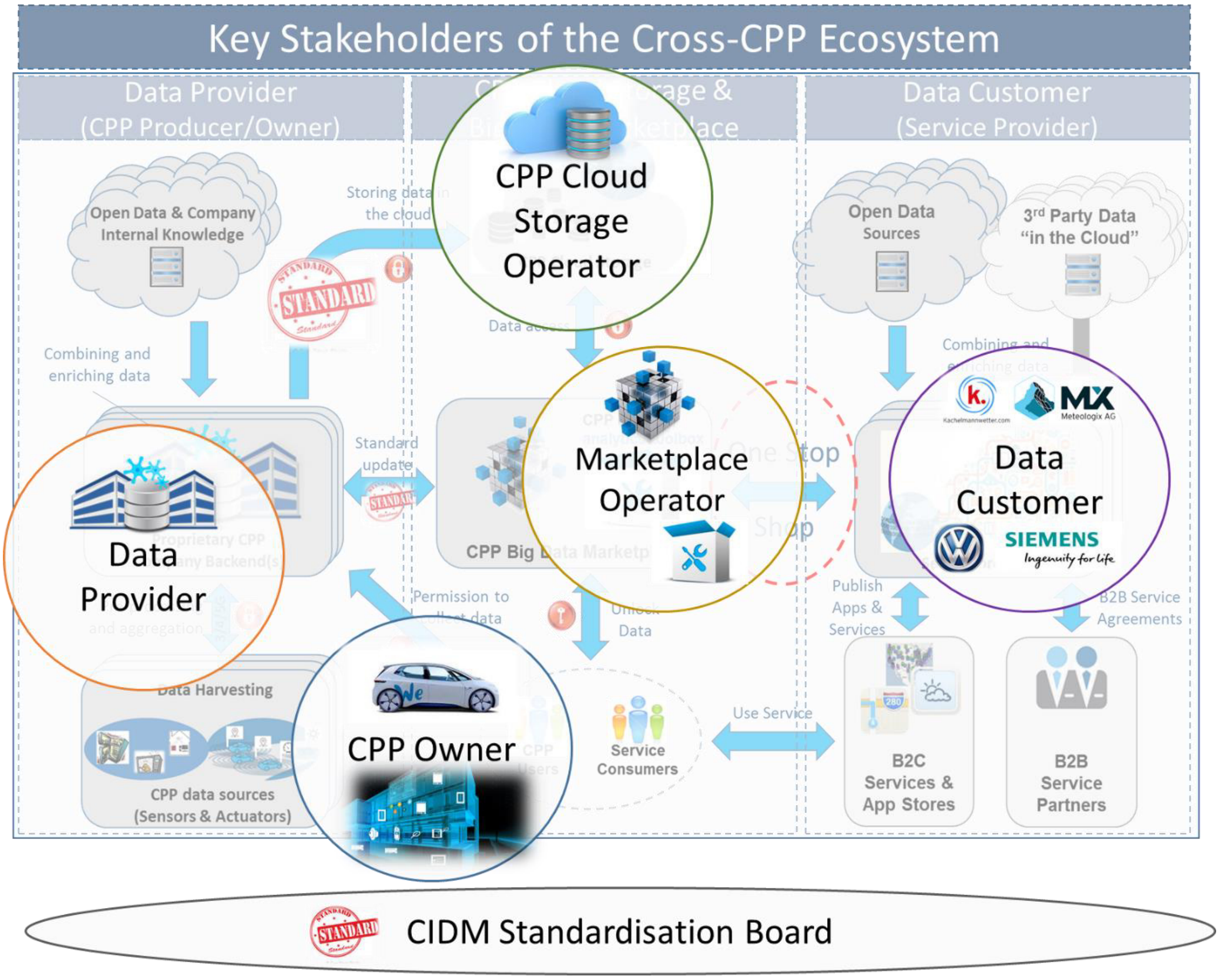


Figure 1: Support material for all key Stakeholders of the Cross CPP Ecosystem

To achieve such envisaged Methodology Concept and to identify required tailored support material to be developed, the specific view and needs for each of the Cross-CPP ecosystem stakeholders was taken into account, which results into the following four types of support materials:

Developer Guidelines: This type represent specific developer guidelines for industrial marketplace users (data providers & data customers), aiming to give these user groups all necessary knowledge on how to connect their systems with the heart of the Cross-CPP ecosystem (cloud storage and marketplace)

User Guidelines: This type represent specific user guidelines for all type of Cross-CPP marketplace users (Data Owner, Data Customer, Marketplace Administrators), aiming to help users on how to use the platform (marketplace, toolbox, context sensitive functionalities and context sensitive solution) and gives knowledge about the different functionalities available.

Concept Papers: This documents provide easy understandable general descriptions of the key innovative project outcome, like the overall Cross-CPP ecosystem, the central standardization concept (Common Industrial Data Model - CIDM), as well as the open specification of the CIDM.

Ecosystem Policy Guidelines: As a complementary support material, this document provide covers legal, privacy and consent regulations for key processes/activities in respect to the actions/roles of the various stakeholders and their interaction required for the operation of the Cross-CPP Ecosystem.

The developed support material is of large scale (9 documents with overall 466 pages), anyway the material is included as appendix to this document (D6.5). Given the fact that all support materials are of public nature, D6.5 also includes the hyperlinks wherefrom to download all documents (see Table 1).

The following Table 1 gives a more detailed overview on all developed methodology support material, including a short description of each document purpose and the targeted Cross-CPP stakeholder group.

1.2 Overview on Cross-CPP methodology support material

Table 1: Overview on Cross-CPP methodology support material

	Support Material Name	Short summary of support material	Addressed Key Stakeholder groups	Annex number and Hyperlink to material [1]
Developer Guide	Cross-CPP Marketplace Developer Guide Data Provider	This guide aims to help developers from an Original Equipment Manufacturer (OEM from now on) company about how to develop a Company Backend system capable of sharing data with the Cross-CPP solution. Data Provider Developer Guide offers data providers the information needed to connect a Company Backend system to the Cross-CPP Cloud Storage to share CPP data in Data Packages using Common Industrial Data Model (CIDM)	Data Provider	Annex 1: Developer Guide Service Provider LINK
Developer Guide	Cross-CPP Marketplace Developer Guide Data Consumer	This guide aims to help users from Service Provider companies on how to use the platform and give knowledge about the different functionalities available. Service Providers User Guide describes Service Providers functionalities within Cross-CPP Marketplace such as discovering available data, requesting data from CPP owners, managing AEON subscription channels, understanding the utilities and how to use the analytics toolbox and its requests and results, the data views filtering utility, and to understand the use of the context monitoring and extraction module.	Data Consumer	Annex 2: Developer Guide Data Provider LINK
User Guide	Cross-CPP Marketplace User Guide Data Owner	This guide aims to help users from a CPP owner role on how to use the platform and gives knowledge about the different functionalities available. CPP owner User Guide describes CPP owner functionalities within Cross-CPP Marketplace, such as discovering available data, browsing available data requests, managing accepted requests and browsing data sent to the marketplace and to the Service Providers	Data Owner	Annex 3: User Guide Data Owner LINK

[1] See also Figure 2 for representation of support material on our project webpage (www.cross-cpp.eu)

	Cross-CPP Marketplace User Guide Administrator	This guide aims to help users from an Administrator role on how to use the platform and give knowledge about the different functionalities available. Administrator User Guide describes management functionalities within Cross-CPP Marketplace, such as listing, reviewing and managing data signals, measurement channels and platform users.	Administrator	Annex 5: User Guide Administrator LINK
	Cross-CPP Marketplace User Guide Data Consumer	This guide aims to help developers from Data Consumers companies (from now on called Service Providers) on how to develop a backend system capable of working with Cross-CPP marketplace data and give knowledge about the different functionalities available. Service Provider Developer Guide offers data consumers the information needed to develop a backend system to start working with Cross-CPP Marketplace data.	Data Customer	Annex 4: User Guide Service Provider LINK
Concept Papers	Cross-CPP Ecosystem White Paper	In this white paper, the key innovations of the Cross-CPP ecosystem concept are presented. These key innovative features concern the Standard Data Model fundamental for the brand independent data exchange, the one-stop shop concept represented by the Cross-CPP Marketplace acting as the central mediator for the data exchange between data provider and data consumer and the basic data privacy and security concept granting full control to the data owner over his data	Data Customer Data Provider MP Operator Cloud Storage Operator Standardization Board	Annex 6: Ecosystem White Paper LINK
	CIDM Cross-Industrial Data Model White Paper	In this white paper, the Cross Industrial Data Model (CIDM) is presented, which - being the central standardization concept - is one of the most crucial parts of the Cross-CPP ecosystem. It features the essential link between Manufacturers of cyber-physical mass products (CPP) and Service Providers, making a huge step in simplifying the processes of data exchange and data provision, at the same time allowing data owners full control over their data at any time	Data Customer Data Provider MP Operator Cloud Storage Operator Standardization Board	Annex 7: CIDM White Paper LINK
	CIDM Cross-Industrial Data Model	In reference to the CIDM White Paper, this document comprises the specification of the CIDM and describes a key result of the Cross CPP project. A machine-readable CIDM definition is publicly available as part of the SDK.	Data Customer Data Provider	Annex 8: CIDM Open Specification

	Open Specification	The open specification document covers among other, the terminology and format of the CIDM Specification as well as the Model Architecture, the Signal Layer Specification, the Measurement Layer Specification, the Data Layer Specification and the OpenAPI Specification.	MP Operator Cloud Storage Operator Standardization Board	LINK
Ecosystem Policy Guidelines	CROSS-CPP Ecosystem Policy Guidelines	The Ecosystem Policy guidelines cover legal, privacy and consent regulations for key processes/activities in respect to the actions/roles of the various stakeholders and their interaction required for the operation of the Cross-CPP Ecosystem. These Methodology Guidelines describes regulations in respect to the interaction between the key Cross-CPP Ecosystem Stakeholders with the Ecosystem, such as contractual and privacy/security issues, as well as update of reference data model regulations.	Data Customer Data Provider MP Operator Cloud Storage Operator Standardization Board	Annex 9: Ecosystem Policy Guideline LINK

1.3 Methodology support material presentation

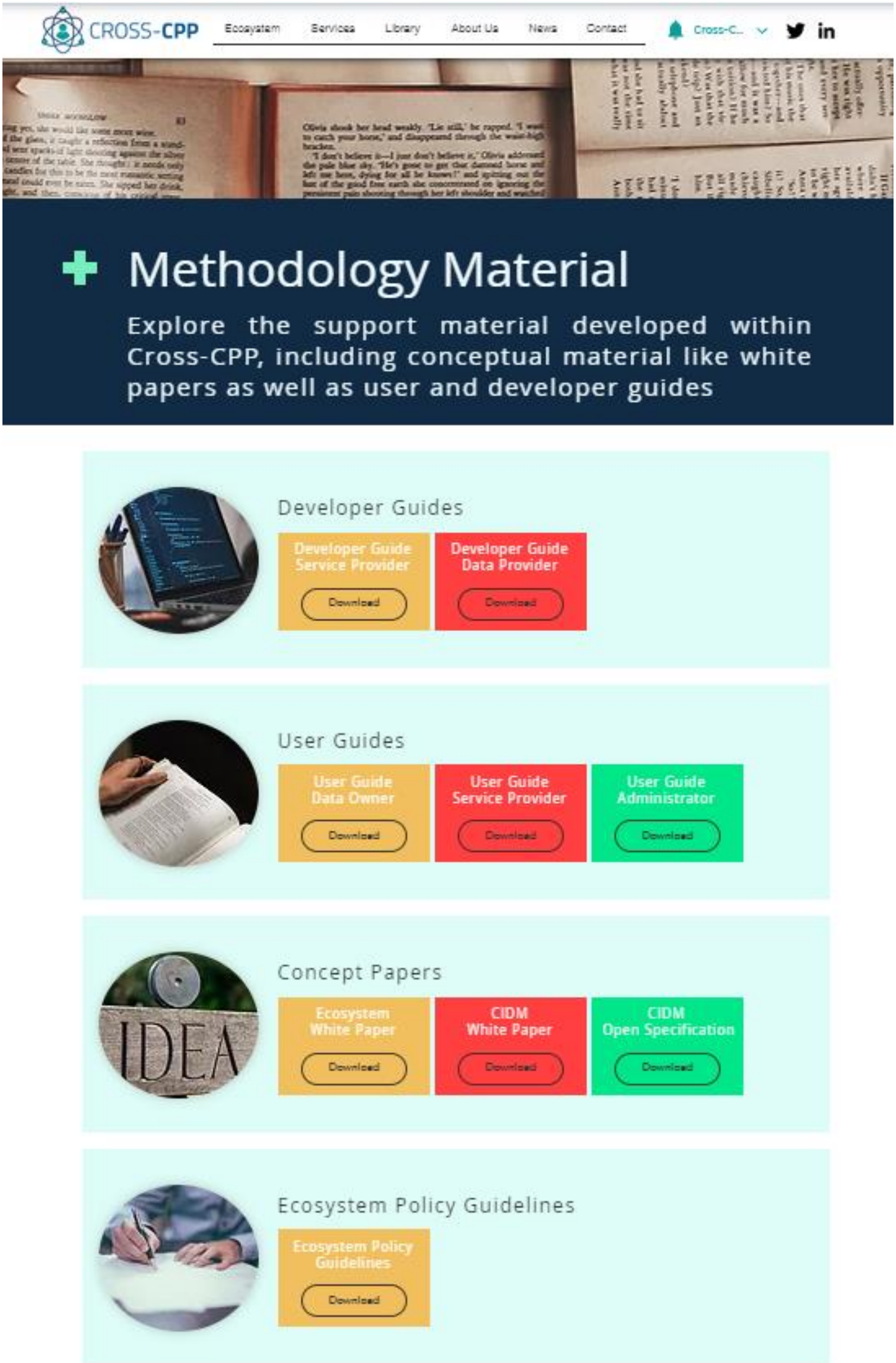


Figure 2: Cross-CPP Methodology support material representation on our project website (www.cross-cpp.eu)

2 Annex 1: Developer Guide Service Provider

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

CROSS-CPP

Cross-CPP MARKETPLACE DEVELOPER GUIDE

CPP DATA CONSUMER GUIDE

(CROSS-SECTORIAL SERVICE PROVIDER)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780167

Introduction

Service Provider Developer Guide offers data consumers the information needed to develop a backend system to start working with Cross-CPP Marketplace data.

Purpose

This guide aims to help developers from Data Consumers companies (from now on called Service Providers) on how to develop a backend system capable of working with Cross-CPP marketplace data and give knowledge about the different functionalities available.

Audience

This guide is meant for and solely for developers Service Provider companies that wants to work with Cross-CPP marketplace.

Scope

The contents of this guide are meant to be taken into consideration only when developing a backend system looking to work with Cross-CPP Marketplace and will only cover functionalities meant to be used by those developers.

Cross-CPP team does not take responsibility on bad use of the application or the data provided when not following the instructions given in this guide.

If you find there is no content in this guide for some functionality you can request it through: cross-cpp-support@lists.atosresearch.eu.

Troubleshooting

For any questions or inquiries about the use of the Cross-CPP Marketplace API or AEON, or the contents of it or this guide, or if you find there is no content in this guide for some functionality please forward it to, please forward it to: marketplace-support@cross-cpp.eu.

Contact

Cross-CPP Project website: https://cross-cpp.eu

Cross-CPP Marketplace: https://datagora.eu

Marketplace support: marketplace-support@cross-cpp.eu

Context Monitoring and Extraction Module (CME): context-support@cross-cpp.eu.

Contents

Guide

The Cross-CPP Marketplace offers five distinct sections for data consumers:

- Cross-CPP Marketplace workflow: this section describes the main steps a Service provider needs to perform in order to consume shared data.
- Cross-CPP Marketplace Analytics Toolbox guide: complete guide of all available analytics functions to analyse and work with collected data.
- Cross-CPP Marketplace API: is considered as the customer central point where they can query the available data and the company current requests.
- AEON: bus driven data communication application through which the Cross-CPP Marketplace data is sent for currently active data requests and analytics.
- Context Monitoring and Extraction (CME): for data consumers that want to customise the CME Reasoning Rules to provide needed filters.

1. General Workflow of Marketplace Usage

The main purpose of the Marketplace is to share smart buildings and connected vehicle data and to provide these shared data for creating new added value services. This section objective is to provide an insightful understanding, of how data can be retrieved by Service Providers.

The following figure (Figure 1) shows the Cross-CPP Marketplace generic usage flow[1]. First, a registration at the Marketplace has to be performed in order to identify the Service Provider user. Without registration no access to the functionalities is granted. After the registration, the first step is the discovery of shared data. When enough data is available or the search matches all needs, an offer can be created. Different types of services require different types of data. Therefore, multiple offers can be created. Cross-CPP empowers the data owner users to have control over their data. They need to agree that a Service Provider may have access to their data. In the next step, the Service Provider needs to wait for data owner user's agreements. After users commit to share the data, it can be retrieved by the service provider using the Marketplace API or the SDK for data subscription.

[1] This workflow does not include the use of the toolbox or CME (see sections **Fehler! Verweisquelle konnte nicht gefunden werden.** and **Fehler! Verweisquelle konnte nicht gefunden werden.** for more information)

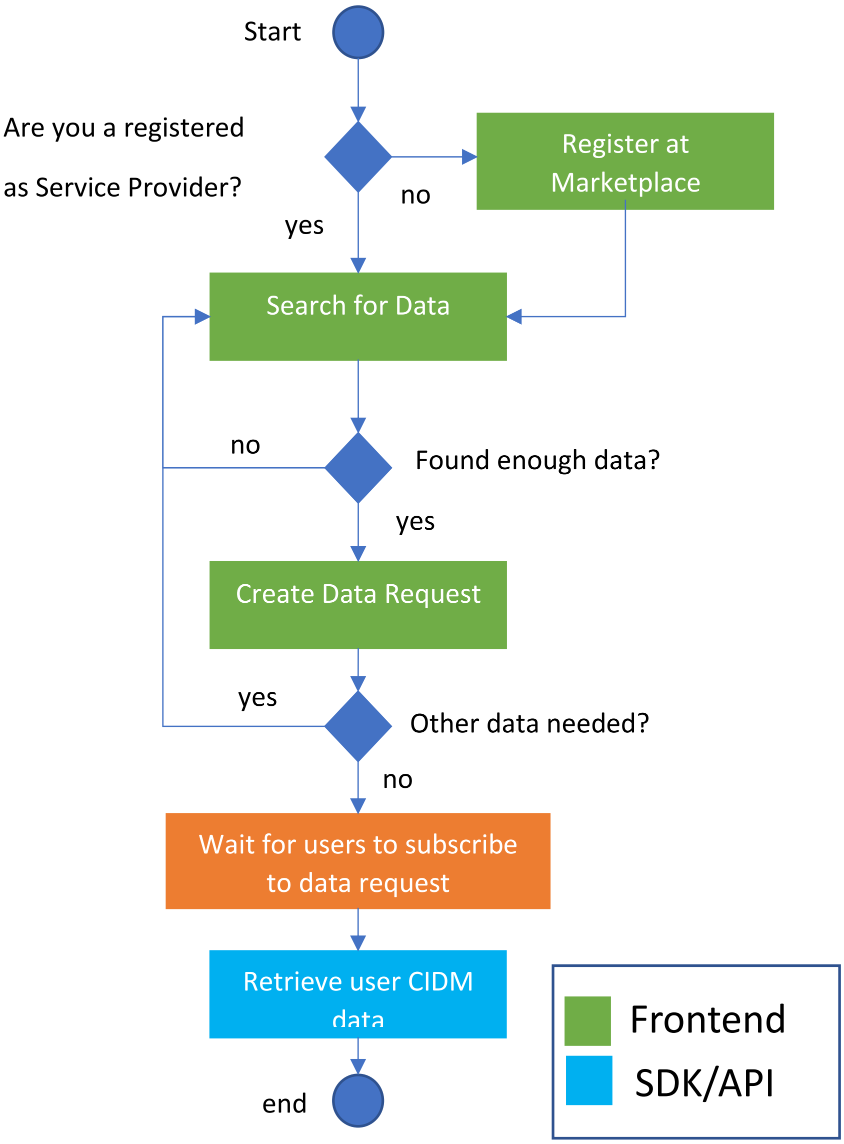


Figure 1. Cross-CPP Marketplace general workflow

1.1. Registration at Marketplace

In order to identify towards the Marketplace, Service Providers need to create an account. In the Sign up! button to create a service provider account request. After filling the sign-up form, the marketplace administrator will receive a notification to validate the service provider. Once the service provider user is validated, the user receives the credentials for Sign In into the Marketplace. See form for options in Figure 2.

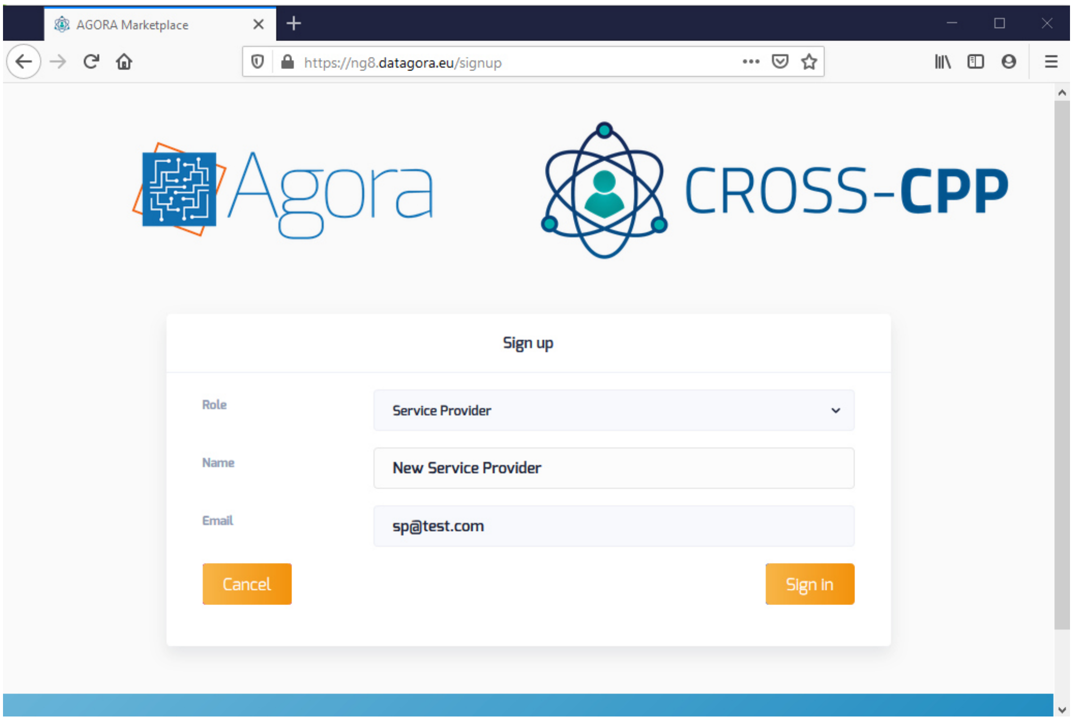


Figure 2. Cross-CPP Marketplace sign up form

In order to interact with the Marketplace API a service provider application needs to include the HTTP header, `X-Auth-Token: <Token>`, in every HTTP request (See more information in 3.1.1 Authentication below). To get the Oauth token the application has to request to `https://idm.datagora.eu/oauth2/token` providing the credentials like in the following example.

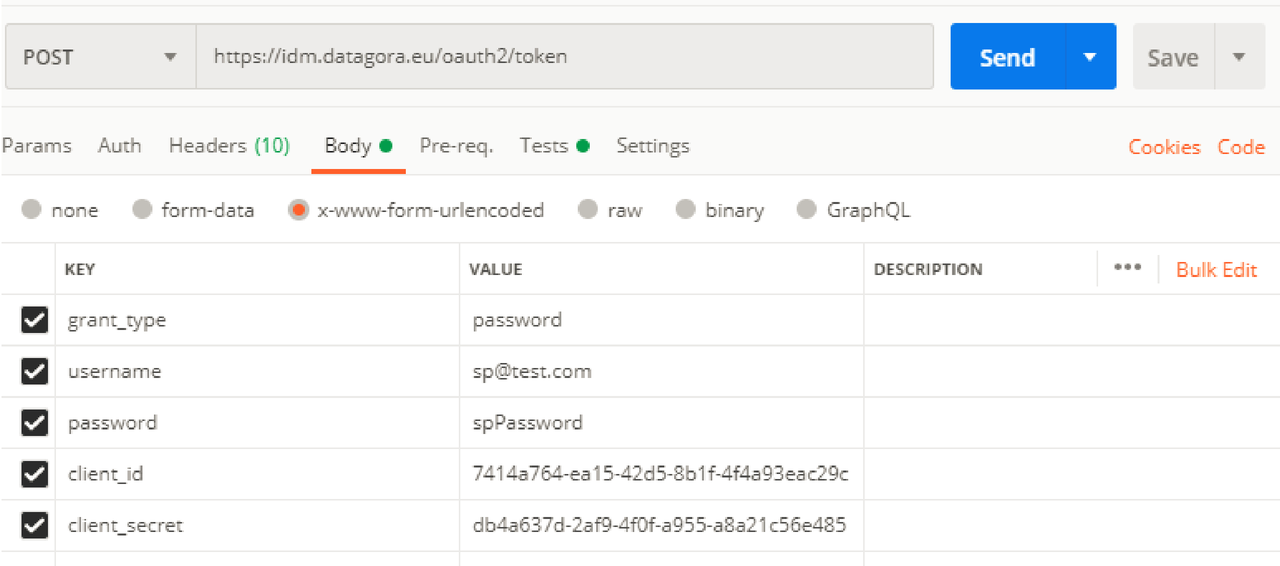


Figure 3. Postman example - Marketplace user authentication request

Response

```
{
    "access_token": "5d0c5c550153fe6d30f8864bd869002adb90c17e",
    "token_type": "Bearer",
    "expires_in": 3599,
    "refresh_token": "836a80b89e74ba2f3399d949ea3405f611bc737e",
    "scope": [
        "bearer"
    ]
}
```


Figure 4. Postman example - Marketplace user authentication response

1.2. Search for Shared Data (Data Discovery)

The second step is the discovery of relevant data. Different queries can be performed selecting different type of filtering like filters based on signals, time, region, etc. Even though the shared data search can be accessed both via the frontend as well as the API, the frontend is the recommended solution for users.

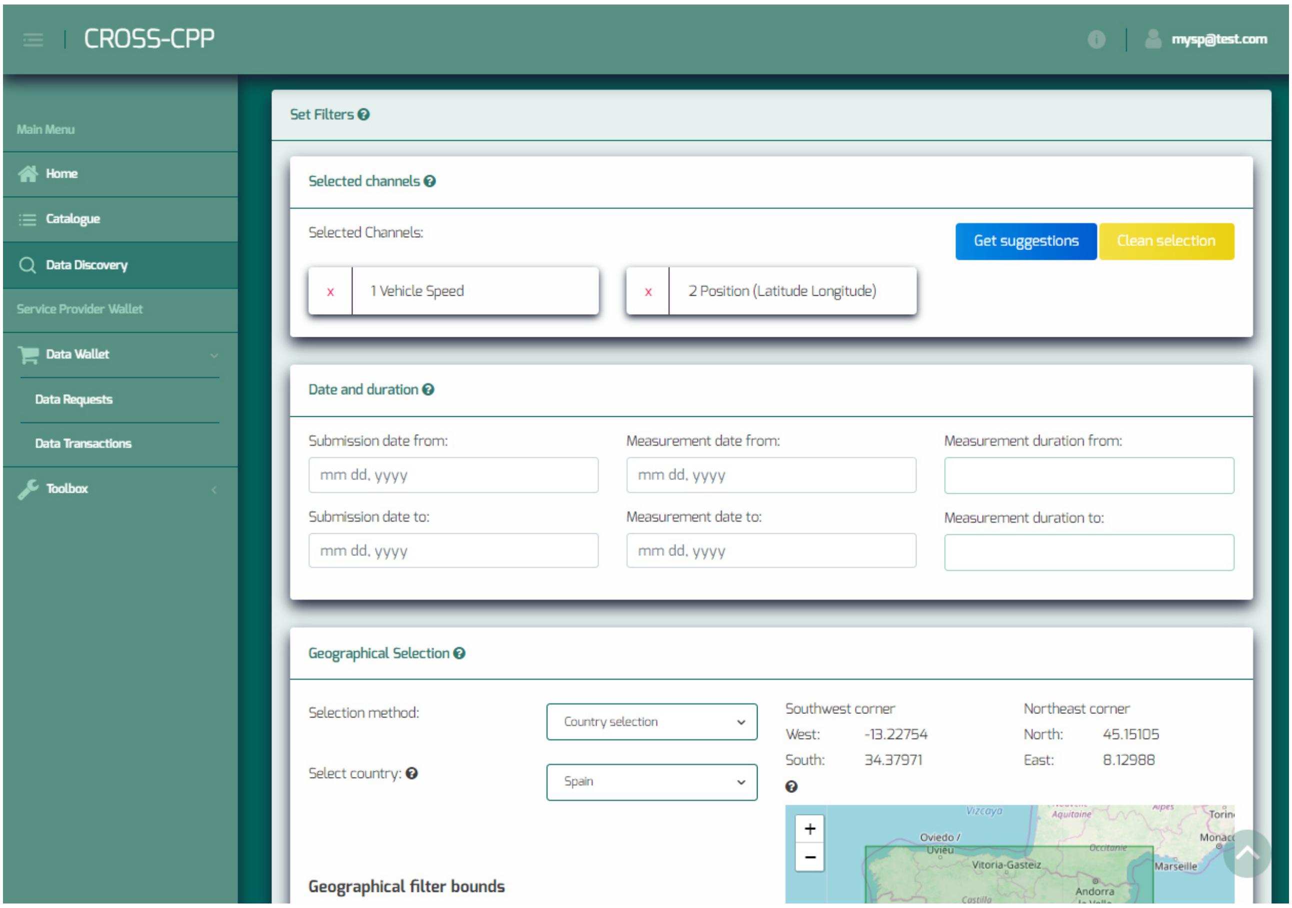


Figure 5. Cross-CPP Data Discovery view

1.2.1. Discovery Definition

The following figure (Figure 6) shows some screen capture of the discovery GUI. On the top, a signal filter can be applied. Signals can be searched for using the input fields in the topmost line. On the lower part of the figure one can see the button to “Get suggestions”. After the first set of channels has been selected this option will invoke the Discovery functionality provided by the

Context Monitoring and Extraction module that makes semantically valid channels suggestion (suggestions are based on a specially created ontology where the signals are in relation to each other). The service provider may add any of the suggestions to the currently selected channels.

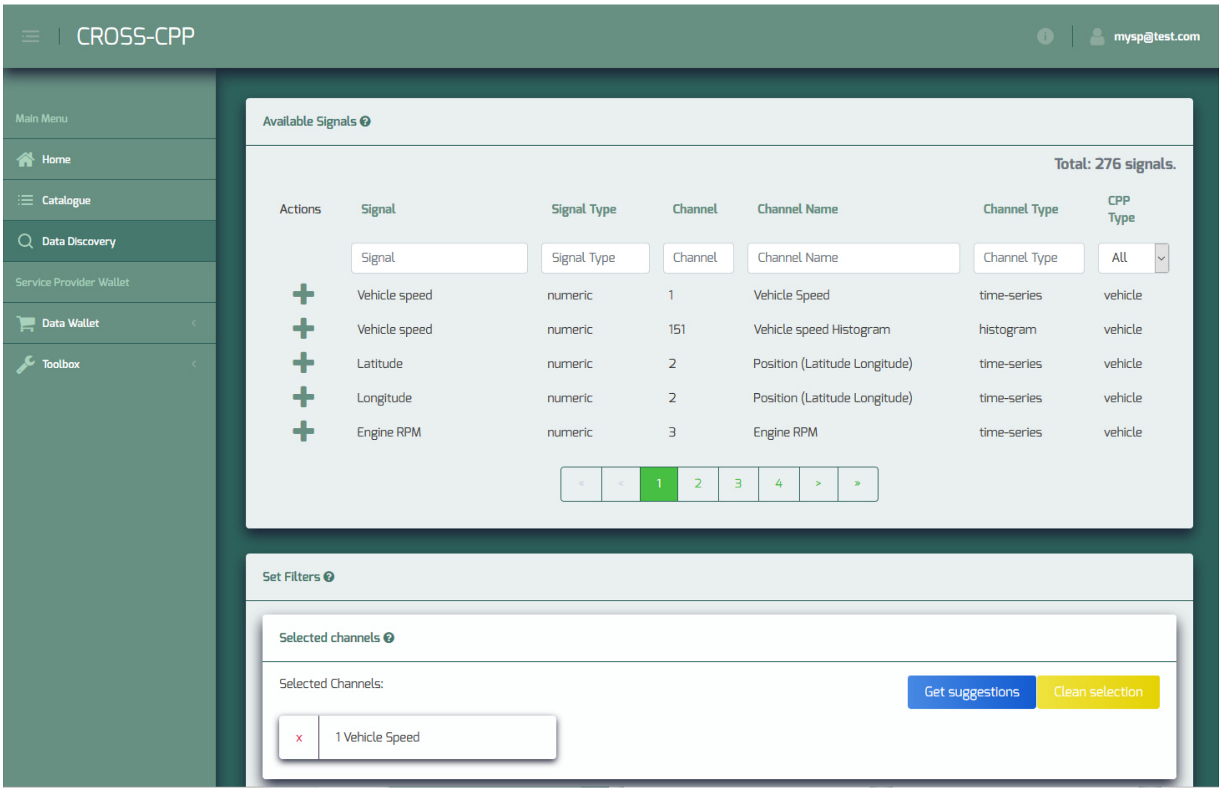


Figure 6. Data Discovery step 1

Below, time-filters can be applied. They cover the submission date of the data as well as the data occurrence time. The duration of data packages is also a filter to ensure a minimum amount of data length.

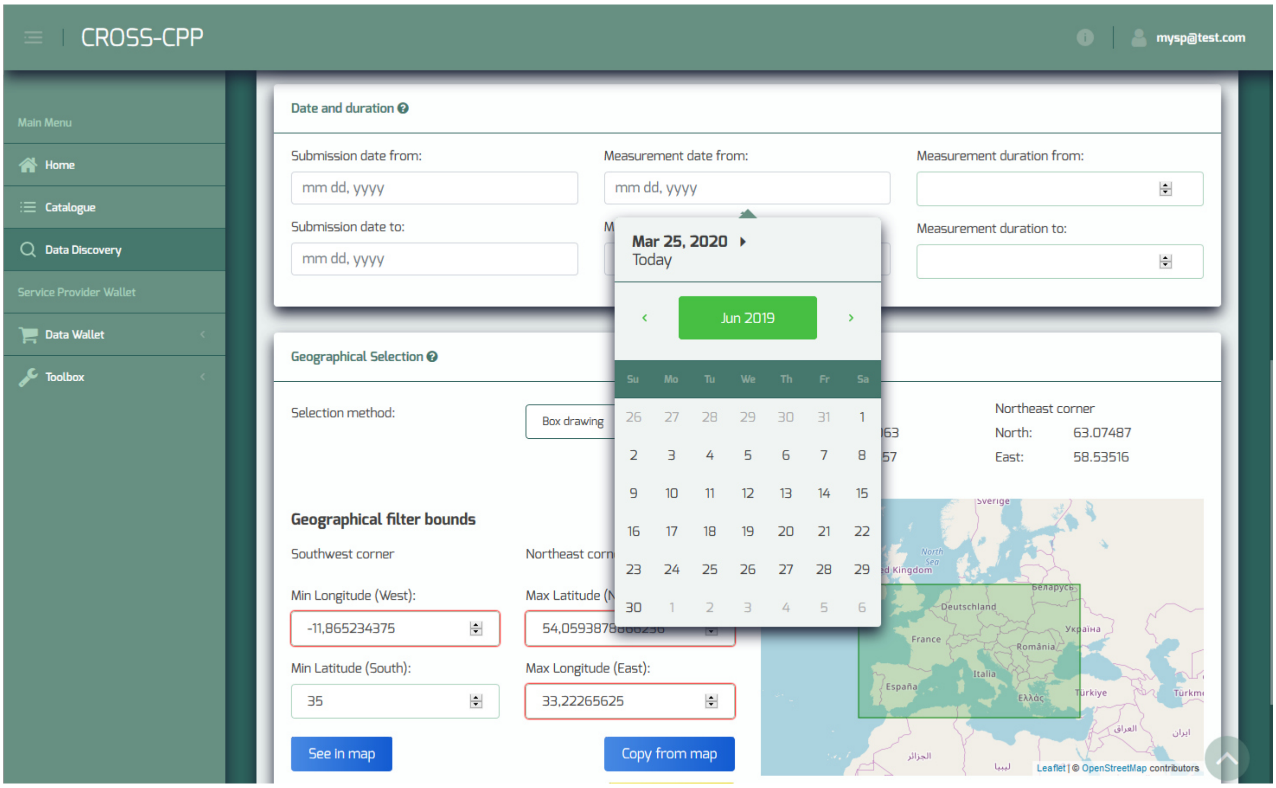


Figure 7. Data Discovery step 2

On the bottom right a geographical filter can be applied. By selecting the map on the screen, inserting the latitude and longitude or selecting countries from the drop-down list. By clicking on "Query" the database is searched for available data.

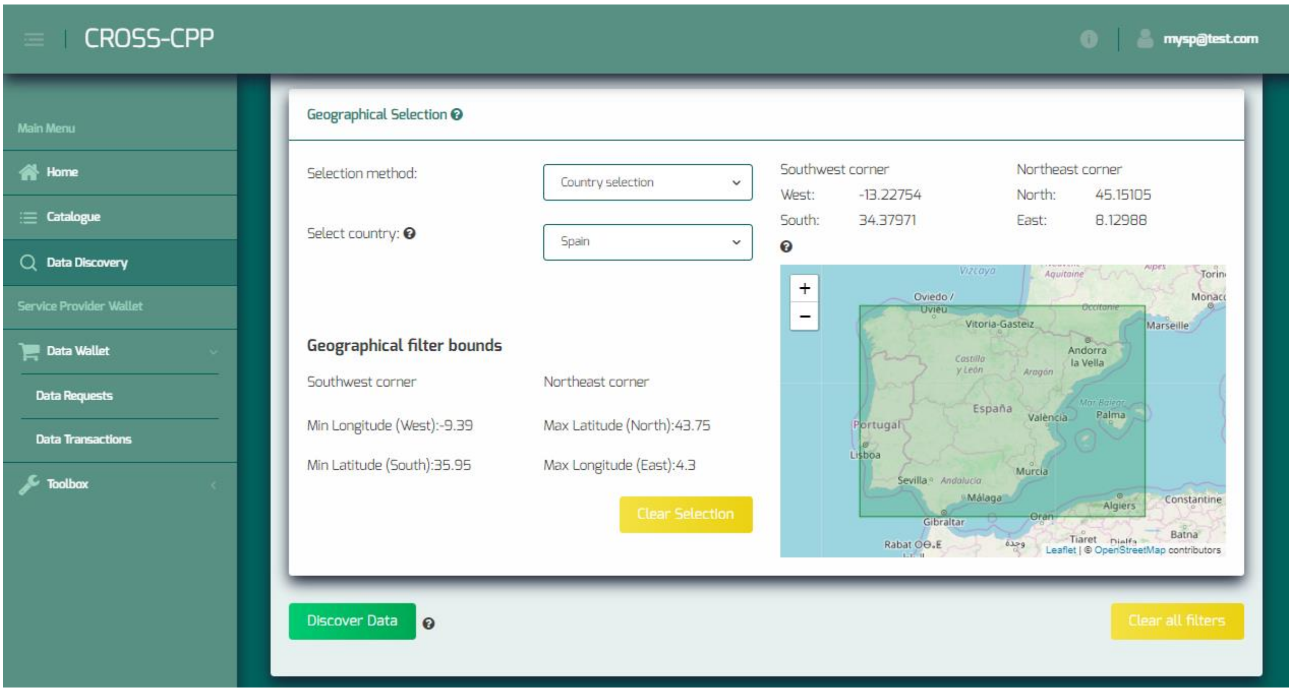


Figure 8. Data Discovery step 3

1.2.2. Discovery Results

After a query has been performed, the result is displayed below. Depending of the amount of found data, this may take some seconds. Figure 9 shows a screenshot of a query result. The result consists of a statistical overview, providing the number of possible data packages and the number of users, who collected those packages. Also, a timeline, when those data packages were generated is provided. A heat map provides a rough estimate of the area, which is covered by the vehicle data discovery (see Figure 10). Afterwards, if the expected availability does not match the needs, the query can be refined. Otherwise, if the availability is satisfactory, an offer needs to be created.

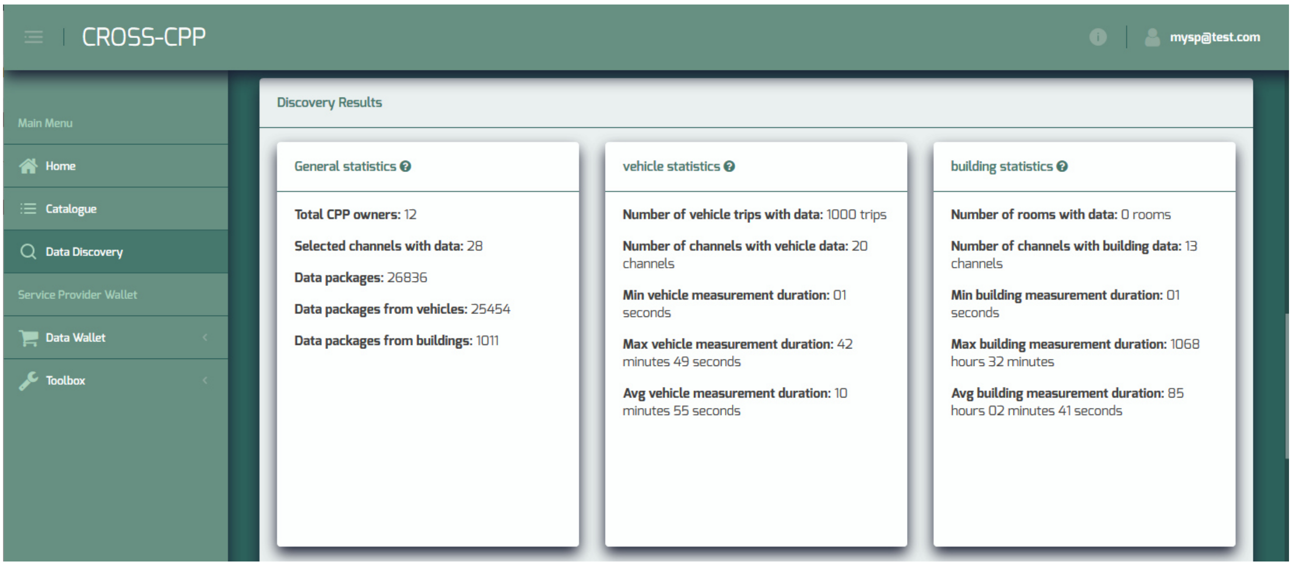


Figure 9. Discovery results - general statistics

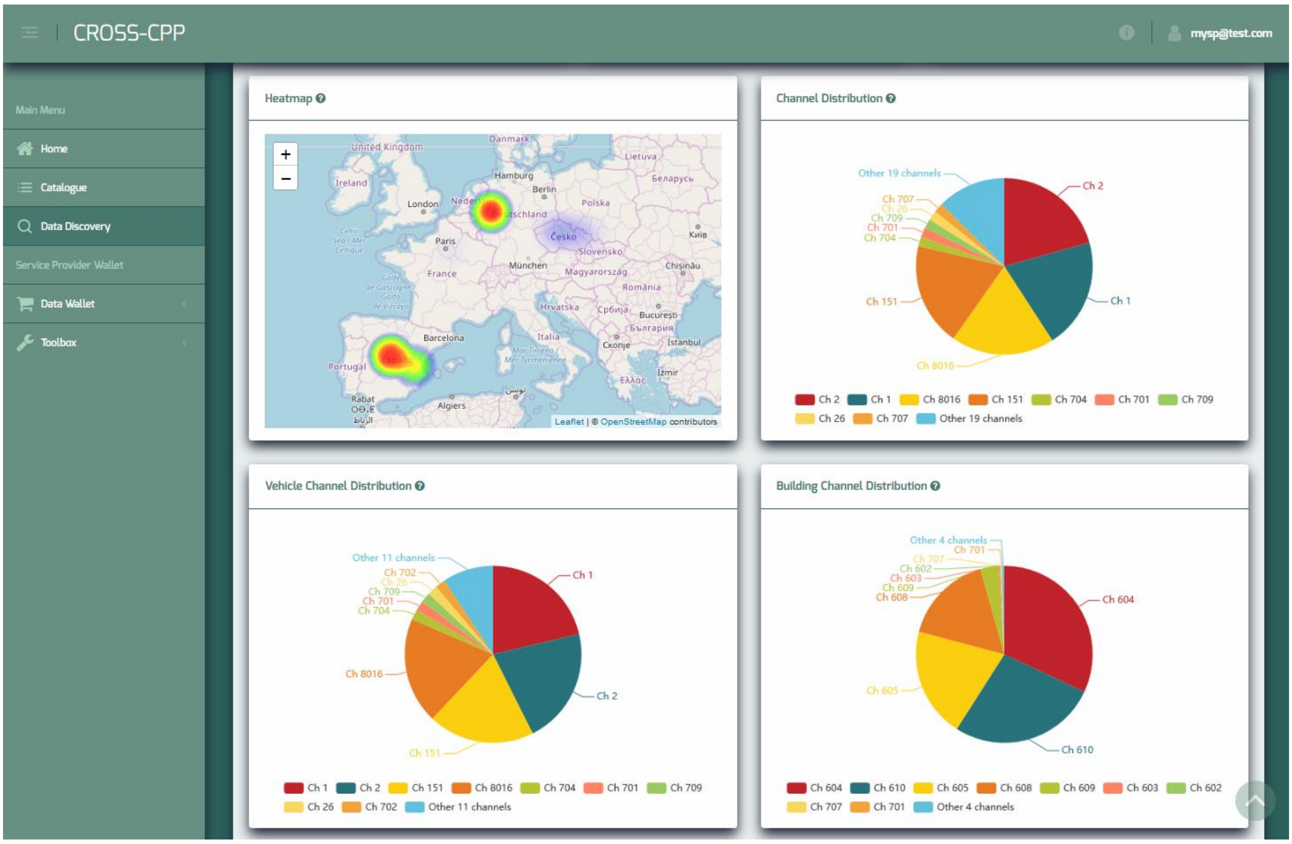


Figure 10. Discovery results - heatmap and charts

1.3. Data Request Creation

When the service provider is satisfied with the availability of data discovered, he can create a Data Request directly below the results of the data discovery. Only a short description must be provided to derive a Data Request from the previous search.

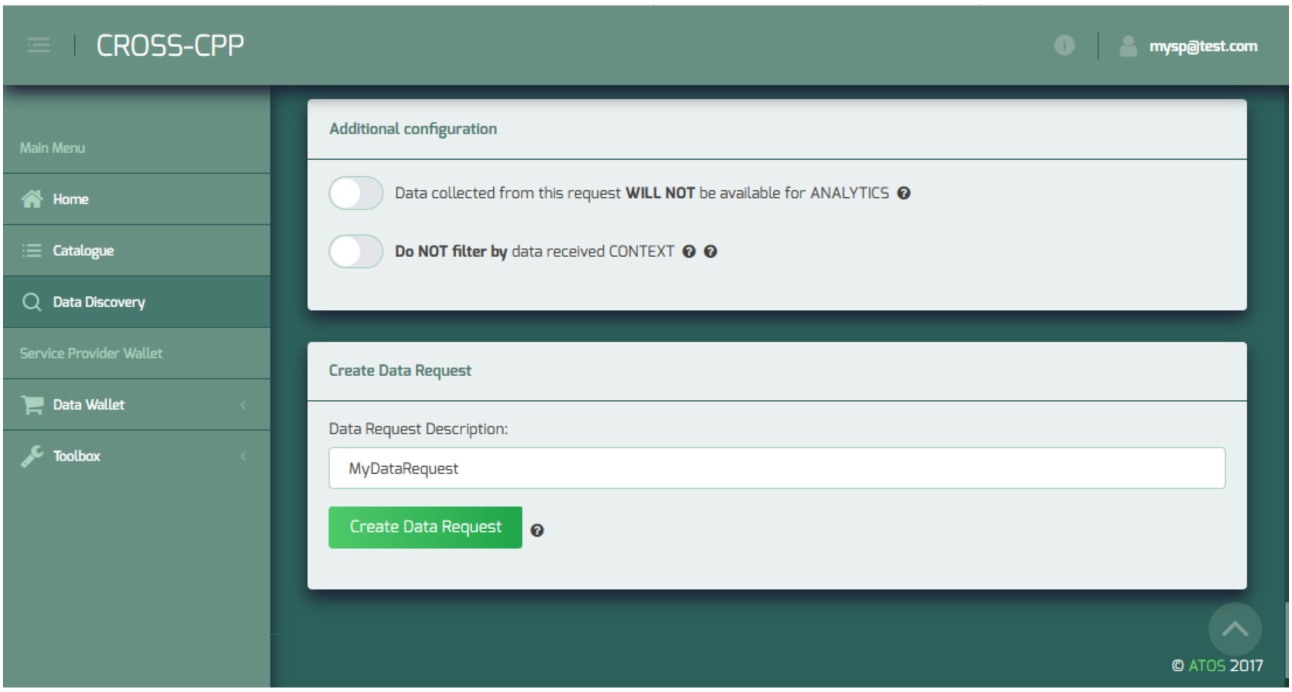


Figure 11. Create data Request from Data Discovery

Data owner users must accept a data request to give their consent to the service provider to access the data. The Marketplace does not store the shared data, but only caches metadata to locate the data in the Cloud Storages.

All created data request can be displayed under the menu

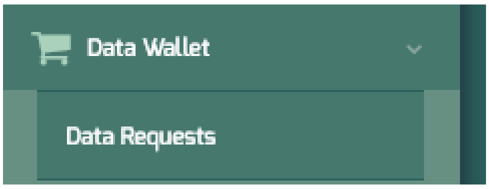


Figure 12 shows the detailed data request view. It displays the id as well as the subscription URL, which are needed to retrieve CIDM data from the offer. Single data packages can be seen by clicking on the “eye” icon in the data packages section (see Figure 13).

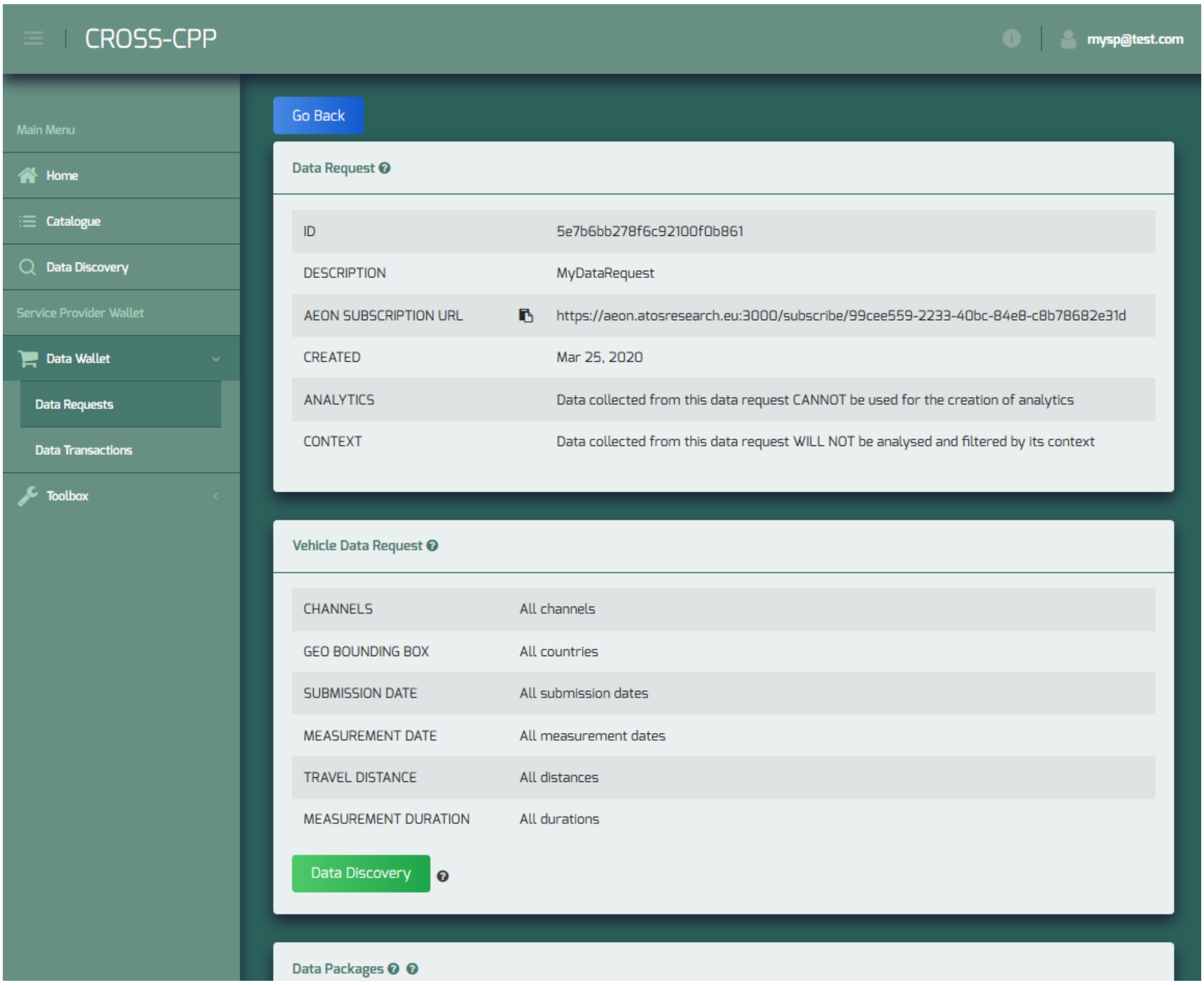


Figure 12. Data Request details

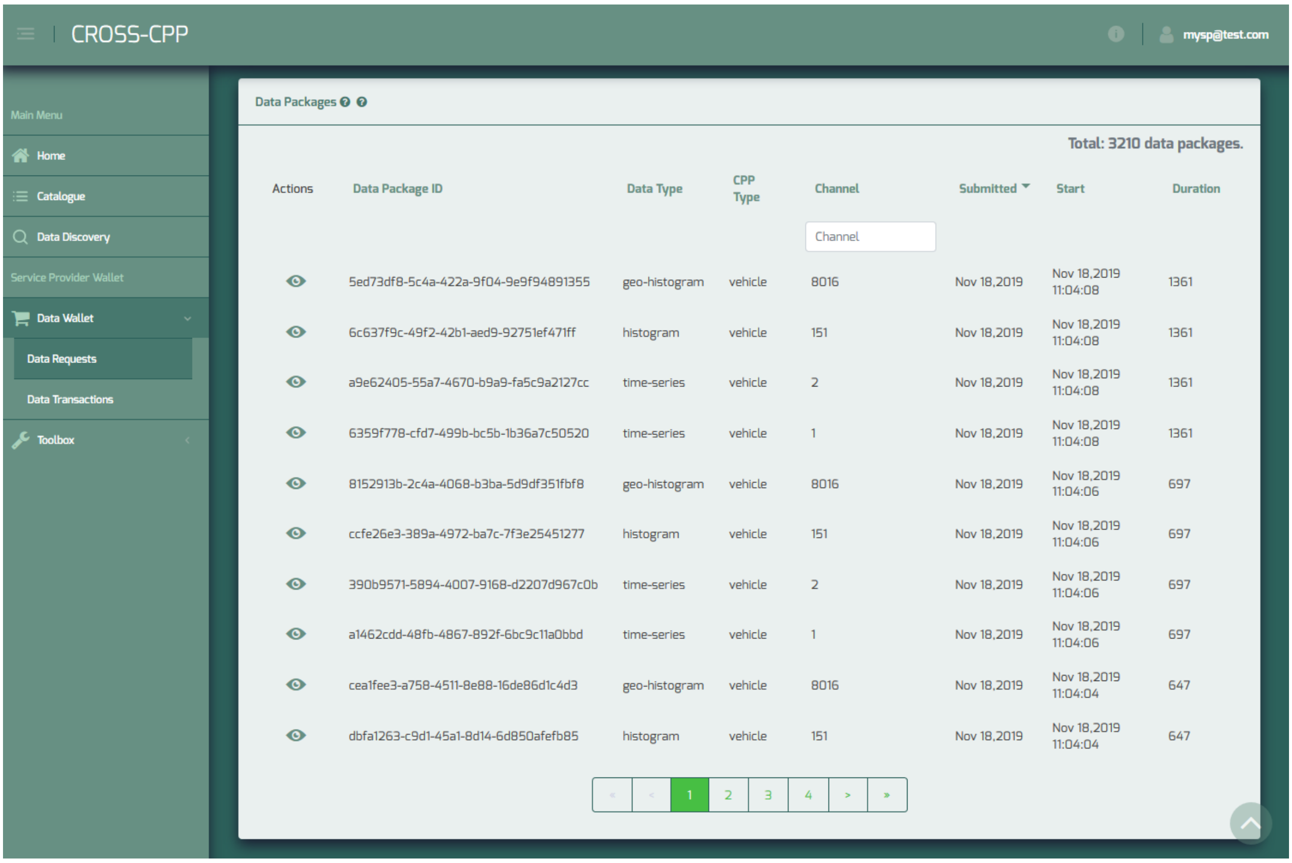


Figure 13. Data Request data packages received list

1.4. Data Retrieval

Data retrieval is the most crucial part of the whole Cross-CPP project. There are two possible methods to retrieve data:

- **Pull approach**: Data must be pulled by the Service Provider application from the Marketplace API.
- **Push approach:** Data is automatically pushed from the MP to the Service Provider application by means of a message queueing systems (AEON).

1.4.1. Pull Approach Data Retrieval

The pull approach uses the Marketplace REST API endpoints to retrieve data or metadata of a specific data request. The basic request consists of a HTTP-GET request providing the ID of the data request (offer) and additional query parameters to filter, sort, and pagination. This section provides some basic information on the procedures. For more detailed information see section 3 API below.

1.4.1.1. Metadata

The screenshot below shows the metadata CIDM pull request performed using Postman[2] tool. https://api.datagora.eu/api/ServiceProviderOffer/{OfferId}/CvimMetadata

Metadata default page size is 1000 (limit=1000) and default page number is 0.

The response header "x-total-count" provides the total number of packages that match the filter. Also, total and page size is provided in the response of the body to facilitate the page iteration.

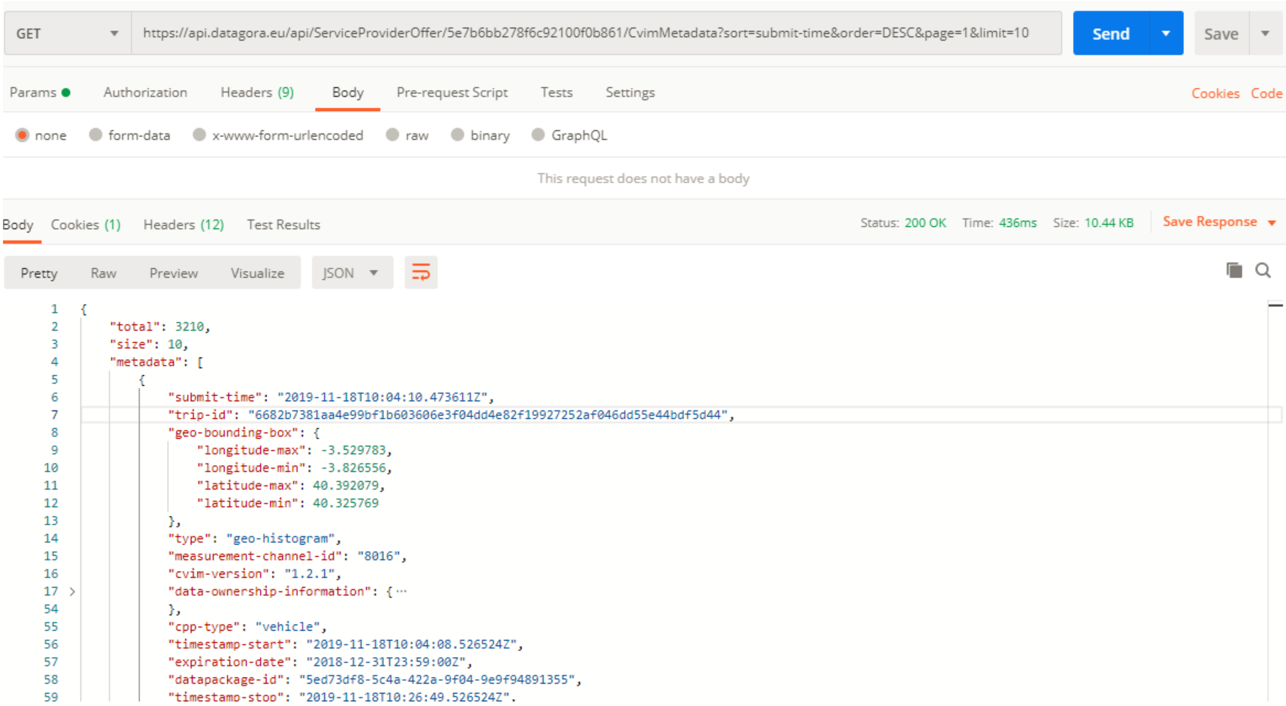


Figure 14. Data Request metadata pull example

1.4.1.2. Data Packages

The screenshot below shows the data CIDM pull request. https://api.datagora.eu/api/ServiceProviderOffer/{OfferId}/cvimDataPackages

Data packages default page size is 10 (limit=10) and default page number is 0.

The response header "x-total-count" provides the total number of packages that match the filter.

[2] https://www.postman.com

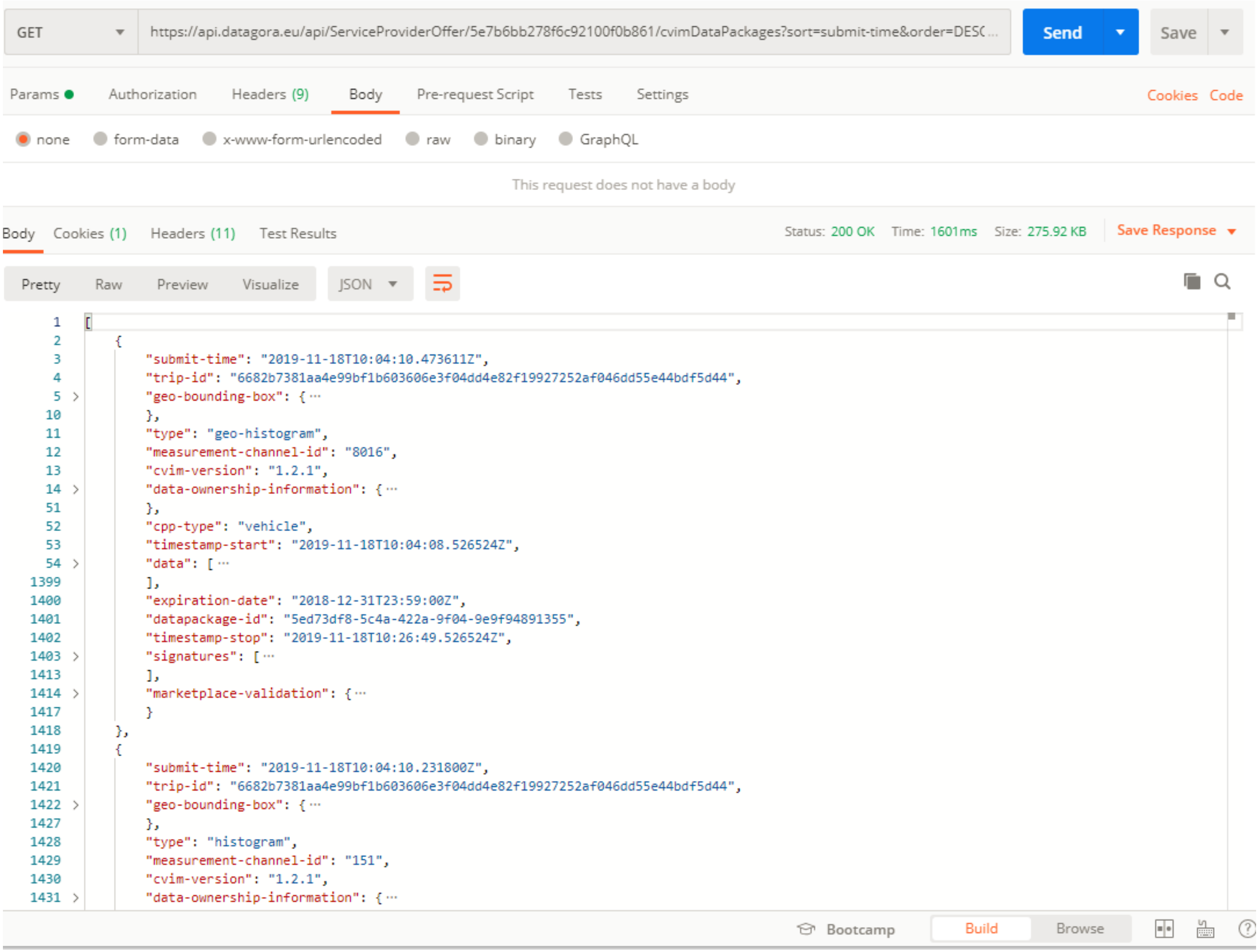


Figure 15. Data Request data pull example

The screenshot below shows a data CIDM pull request to get one data package.
https://api.datagora.eu/api/ServiceProviderOffer/{OfferId}/cvimDataPackages/{datapackageId}

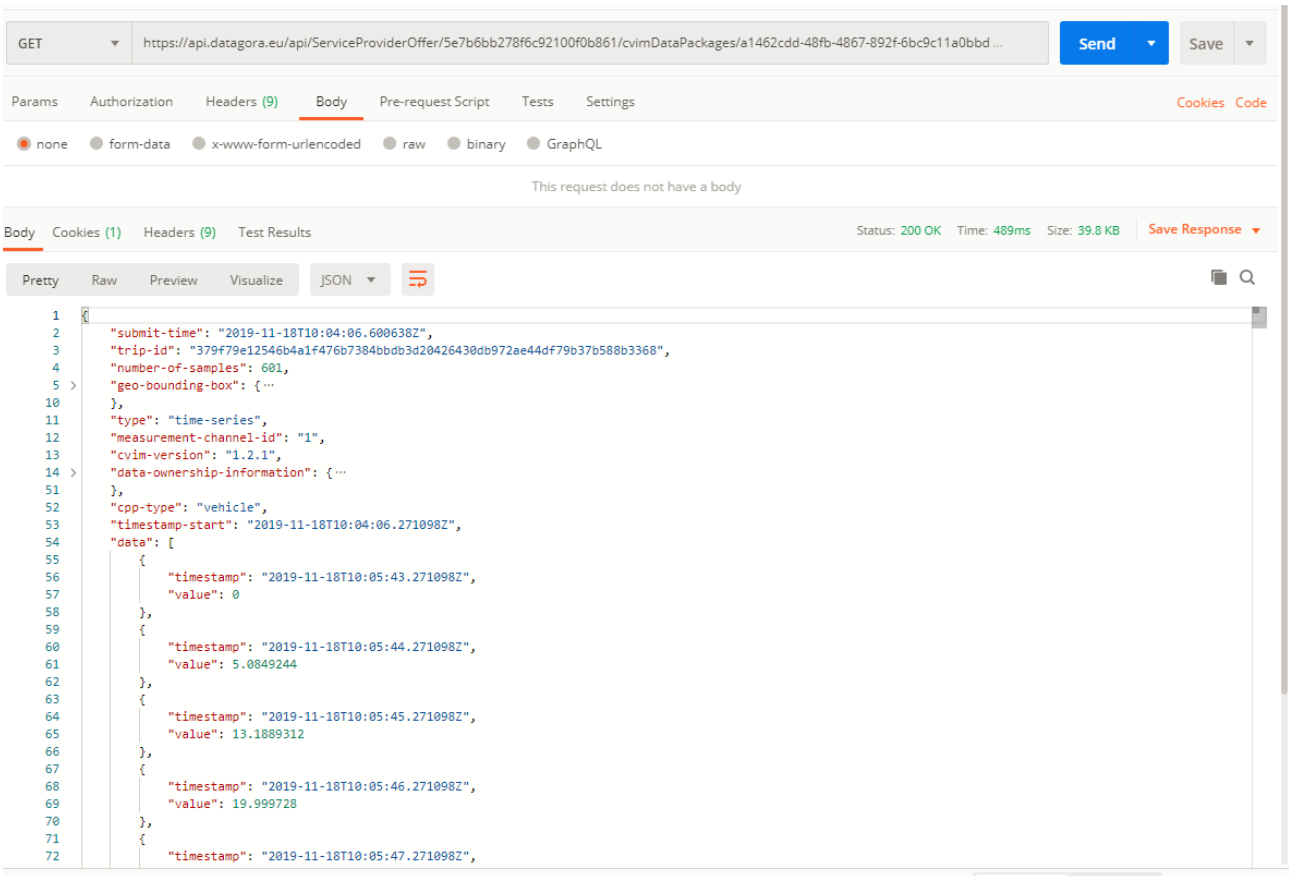


Figure 16. Data package details pull request

1.4.1.3.Query Parameters

Previous queries can filter packets providing query parameters as follow.

page: Number of page [1..x-total-count].

limit: Number of datapackages or metadata to receive in 1 page. Default value is 100 for datapackage and 1000 for metadata.

from: Offset from the first result you want to fetch. Default value is 0.

sort: Field to sort results.

order: Order to sort results (asc or desc).

where: Allow to filter data or metadata. It is JSON object which contains a channels, trip-id, submit-time.min and submit-time.max, timestamp.min and timestamp.max, latitude.min and latitude.max, and longitude.min and longitude.max.

Note: Result window is too large, from must be less than or equal to: [10000].

Where example: where = { "channels" : ["1","2"], "submit-time.min": "2019-07-01T12:39:30Z", "submit-time.max":"2020-05-09T11:38:00Z", "timestamp.min": "2019-07-01T12:39:30Z", "latitude.min" : 49.836, "latitude.max" : 49.840}

1.4.1.4.Last value query

These queries are similar to the previous one but they provide the last datapackage or metadata of every device and every measurement-channel of the data request. They are useful for basic-cpp-information and event-values where the most relevant values are the last ones.

/ServiceProviderOffer/{OfferID}/lastcidmDataPackages

/ServiceProviderOffer/{OfferID}/lastcidmMetadata

1.4.2. Push Approach Data Retrieval

The push approach data retrieval allows the processing of CIDM data packages as soon as they are available in the Marketplace. The push approach uses the AEON platform[3], that is a cloud platform to create applications with real time communications channels. This section provides some basic information on the procedures. For more detailed information on AEON see section 4 AEON below.

AEON platform offers a cloud-based message queuing framework enabling messaging between various entities that wish to communicate with each other seamlessly and reliably using standard protocols.

When a new data request is created, the marketplace creates a new AEON Channel to provide data streaming between the service provider application and the marketplace. The service provider application needs to subscribe to the AEON Channel to receive the shared data. The information to subscribe is provided in the Data Request details (AEON Subscription URL).

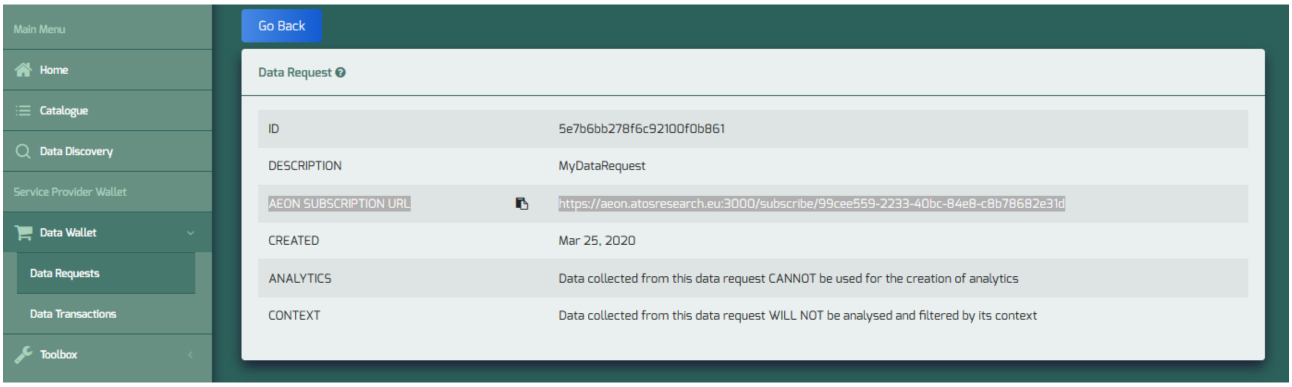


Figure 17. Data Request AEON subscription url

The AEON SDK is provided for Java, Node.js and JavaScript programming languages (https://scm.atosresearch.eu/ari/aeon-sdk/tree/master/SDK/releases).

SDK instance

[3] https://aeon.atosresearch.eu/app/index.html

To start publishing/subscribing with the SDK, it is necessary to instantiate a new SDK object.

```
sdk = new AeonSDK(url, [subscriptionData])
```

* url: AEON Subscription URL provided for the data request created.

* subscriptionData: It is a JSON containing information about the service provider subscription in order to make a unique client subscription.

```
{
      "id": " 7735901e3943424791f19f0dad5e428e",
      "desc": "MyDataRequest unique connection"
  }
```

The combination of "id" and "desc" (strings) makes your subscription unique in the AEON network, the values are provided by de service provider application.

Subscribe

This functionality allows to subscribe to a specific channel and receive all the published information by the data request.

```
sdk.subscribe(received, [control])
```

* received: callback function to receive published data messages coming from the Marketplace.

* control: callback function to receive AEON control messages.

Each time that a message arrives over the subscription, the callback “received” will be executed in order to process the incoming message, the example below shows the messages on the console but you probably would put the data into a database or make a complex process of it .

Control messages are received over the subscription like messages but includes information regarding the status of the connection and the subscription. You can be subscribed (not recommended) without a “control” callback function. If control callback is passed, control messages will be received: see errors table.

Code	Message
1	Bad URL
3	Communication Infrastructure Down
50	Communication Infrastructure up
100	SDK operating in Publication Mode
101	SDK operating in Subscription Mode
201	Subscription in use
202	You are not subscribed
203	Subscription incorrect
250	You have been subscribed

251	Your subscription has been deleted
252	You have been unsubscribed

The example below shows the specification of the *control* and the *received* callbacks:

```
var messagesReceived = 0;
// Print control messages on the console
var control = function control(msg) {
    console.log("Control: ", msg);
    // Control: {"code":200,"desc":"ok","result": [{"subkey":"channelEdited-95504801-queue"}]}
}
// Print data messages and the number of message on the console
var received = function received(msg) {
    console.log("Received: ", msg)
    // Received: {"Message":"<first message from marketplace regarding a data request>"}
    messagesReceived++;
    console.log("Received messages: "+ messagesReceived);
    // Received messages: 1
}
```

2. Toolbox Guide

2.1. General Workflow of Toolbox Usage

The main purpose of the Toolbox is to provide Data Analytics of the shared data. The following Figure 18 shows the usage flow of the Marketplace Toolbox. First, a data request is needed for creating an analysis. Different types of analysis require different types of data. After creating an analysis, analysis data can be retrieved by the service provider using the Marketplace API or the SDK regarding the type of analysis created.

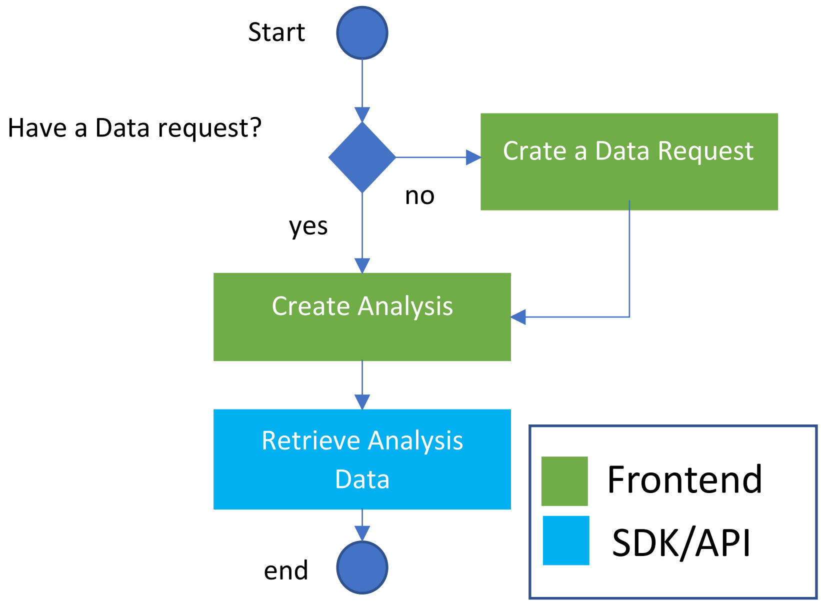


Figure 18. Analytics Toolbox workflow

There are two distinct sections inside the Toolbox: Analytics, which provides the analytical tools to analyse your data, and Data Views, which provides a different way of receiving the data and are used for the Machine Learning analytics.

2.2. Data Views

Data Views enable combining and joining data from measurement channels in a given Data Request. Service providers can use them to receive data filtered by defined criteria or to provide input to machine learning services. The filtered output can be directly sent in a specific AEON channel, or it can be retrieved by invoking the Retrieve function.

Data Views are created from Data Requests. It means that service providers have to choose relevant channels and create a Data Request first. Data owners then found the request and can decide to accept it. The Data View is then formed from channel values in data packages of the data owners who decided to provide the data.

2.2.1. Categories

A Category is a way to label and use the Data View to analyse the collected data for a certain purpose (i.e.: “charger availability” or “dangerous driving”). Categories consists on an enumeration of values provided by the Service Provider to categorize the collected data (i.e.: “high”, “regular”, “low”).

Service Providers can assign a category value to each of the rows generated by the Data View.

Service Providers can create any number of categories, but a Data View can only have one assigned category, and once assigned it cannot be changed.

2.3. Analytics

2.3.1. Time Series Analytics

The Time Series Analytics are a set of analysis that explodes time series data packages. The analysis works in a streaming mode, receiving input data form the marketplace, calculating the results and sending the results to de service provider application subscribed to the analysis.

In order to create the analytics, the first step is to select different types of analytics functions like, sample entropy, permutation entropy, irreversibility, neural networks, regression tree, Arima, Pearson and Spearman.

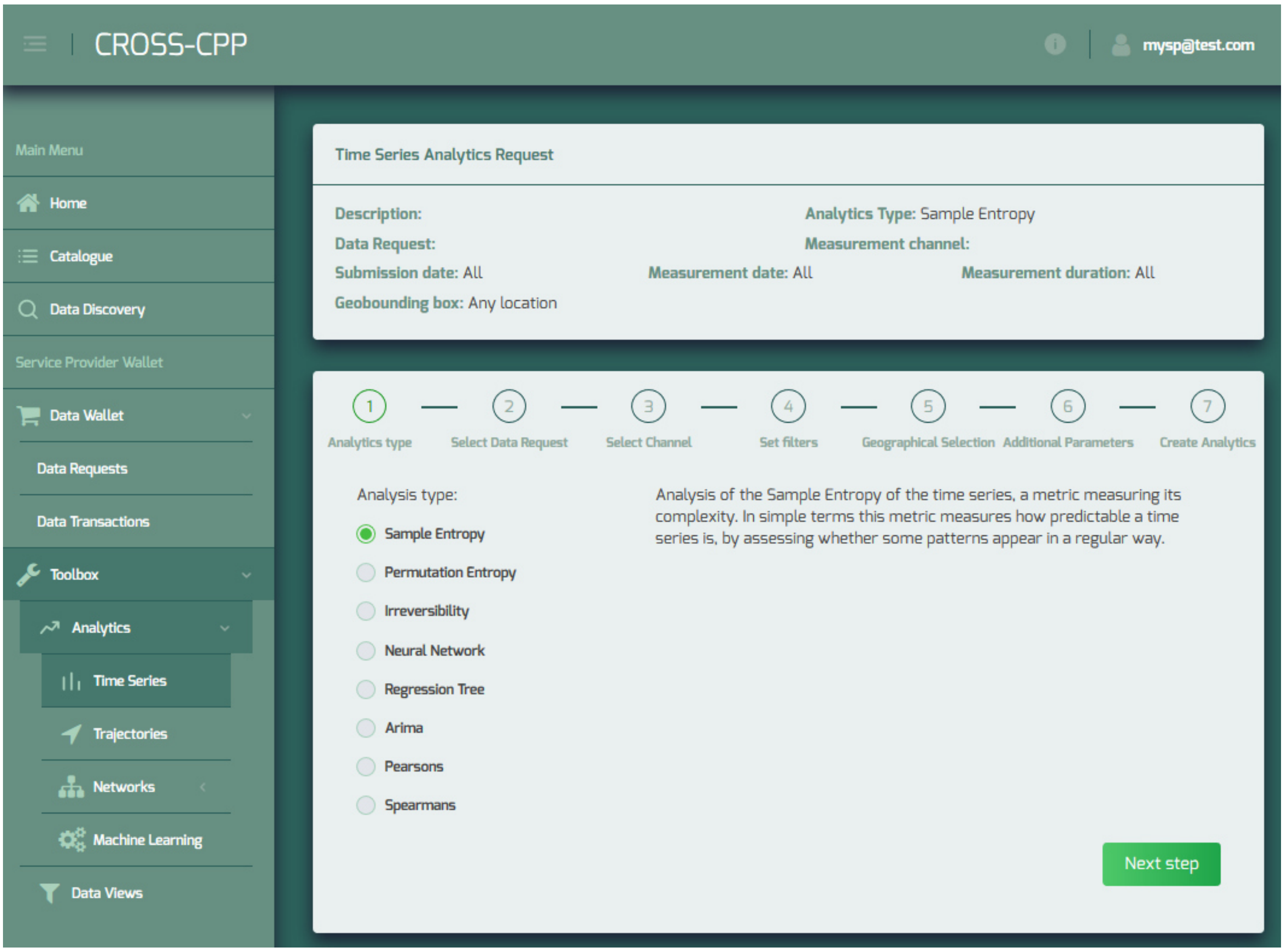


Figure 19: Create Time Series analytics request – select analysis type

The second step is to select the data request that is going to provide the input data of the analysis. The third step is to select the target datapackages of the analytics that has to be time-series. The fourth and fifth steps are to define time filters and location filters. The sixth step is to provide additional parameter regarding the function type selected in the first step. Finally, the last step is to specify the name or description of the analysis. See user guide for further information about the types of analytics and the configuration.

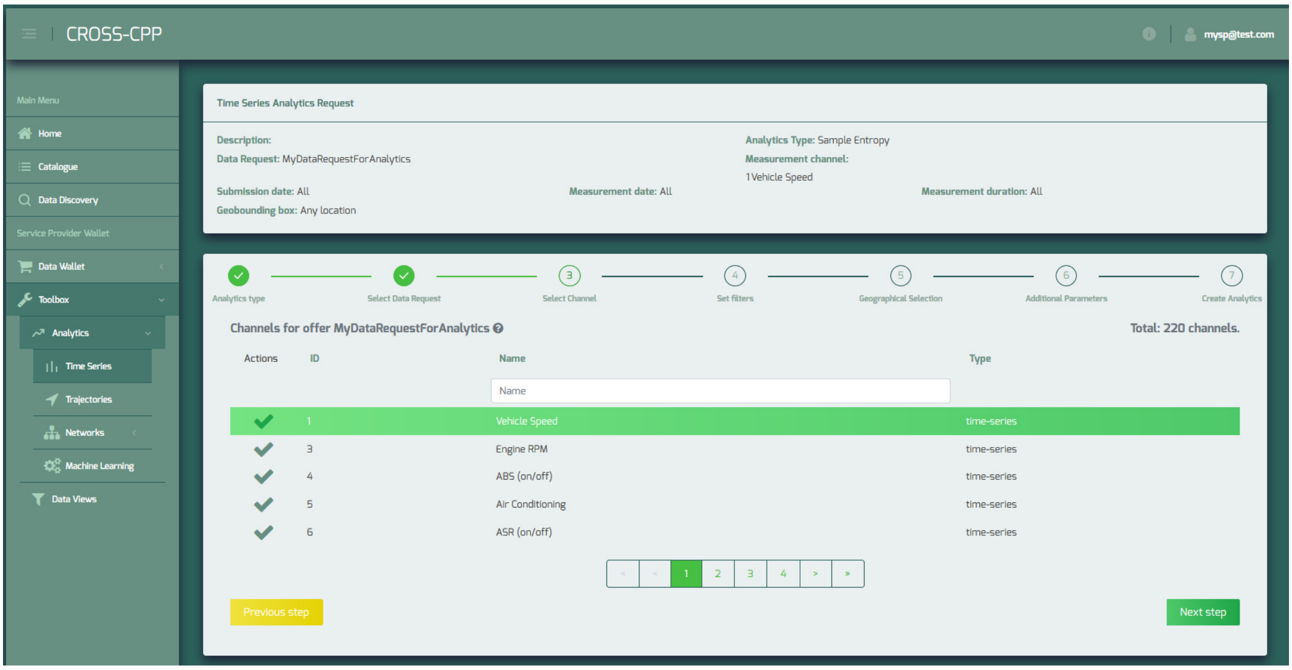


Figure 20: Create Time Series analytics request – select Data Request

Once the analysis is created, the marketplace provides the AEON subscription URL for the service provider application to receive the analytics results.

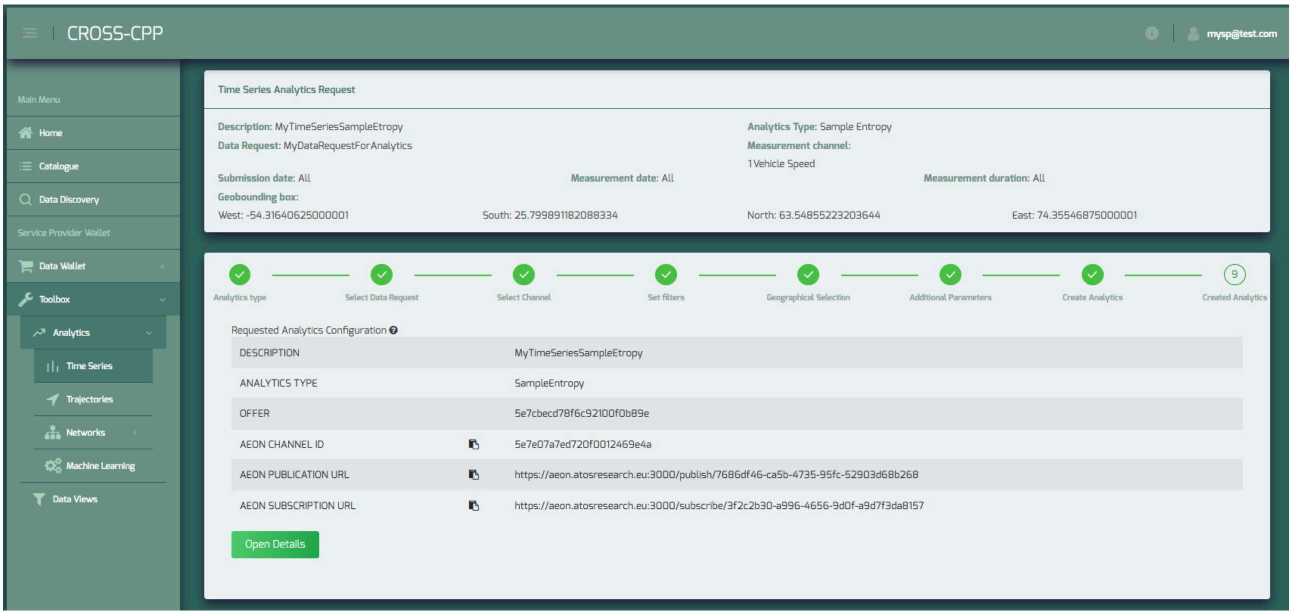


Figure 21: Create Time Series analytics request – requested analytics information

To subscribe and receive the results you have follow the instruction provided for the Push Approach Data Retrieval

2.3.2. Trajectory Analysis

The Trajectory Analytics are a set of analysis that target vehicle information that includes position timeseries datapackages. In order to create a trajectory analysis a data request that includes location is needed. The data request should be configured to collect data for analytics.

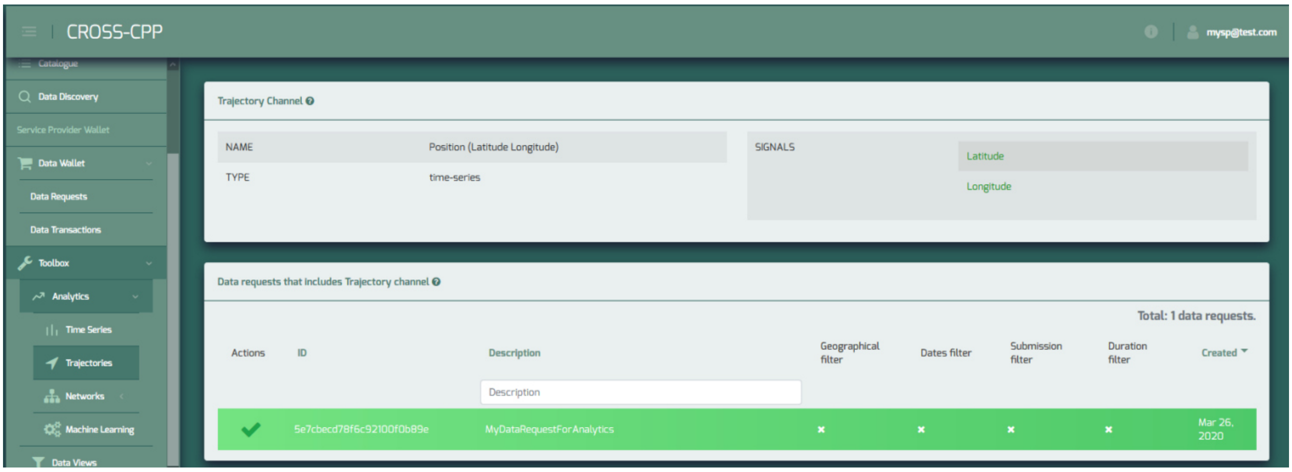


Figure 22: Request trajectory analytics – select Data Request

The filters can be configured for the analysis, like travel dates, submission dates, duration of the trip and location.

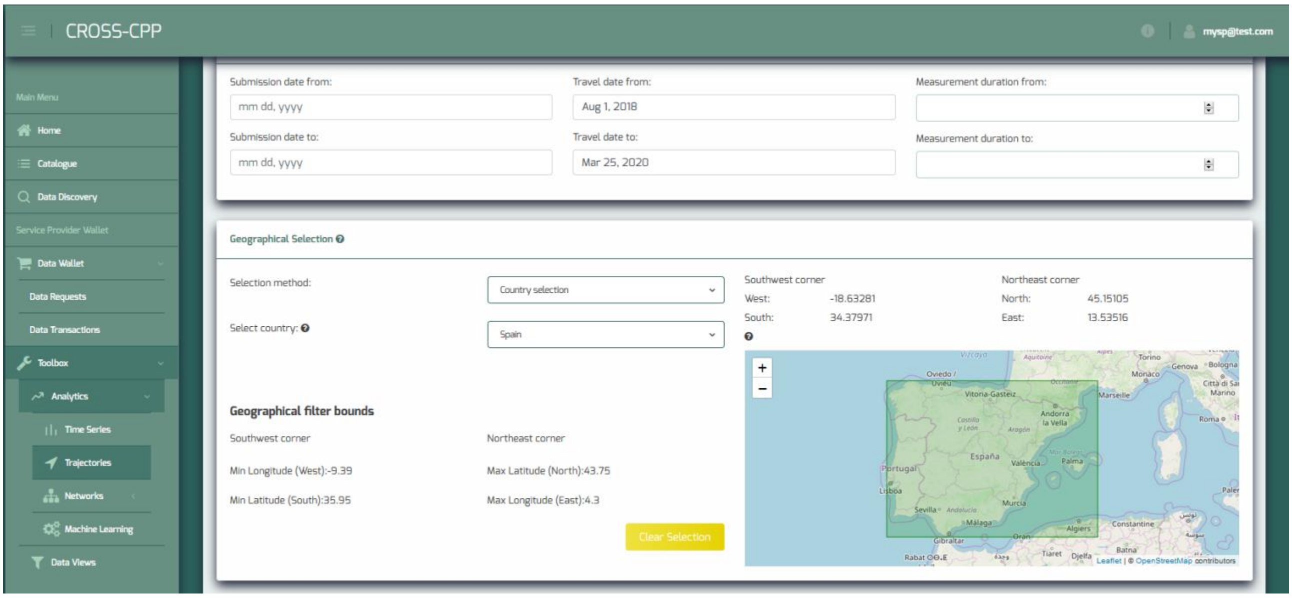


Figure 23: Request trajectory analytics – configure filters

There are many Analytics functions that can be performance selecting the analysis type and providing the parameters values when needed. To learn more about the analysis the user guide.

Finally, to create the request you have to provide the analysis description.

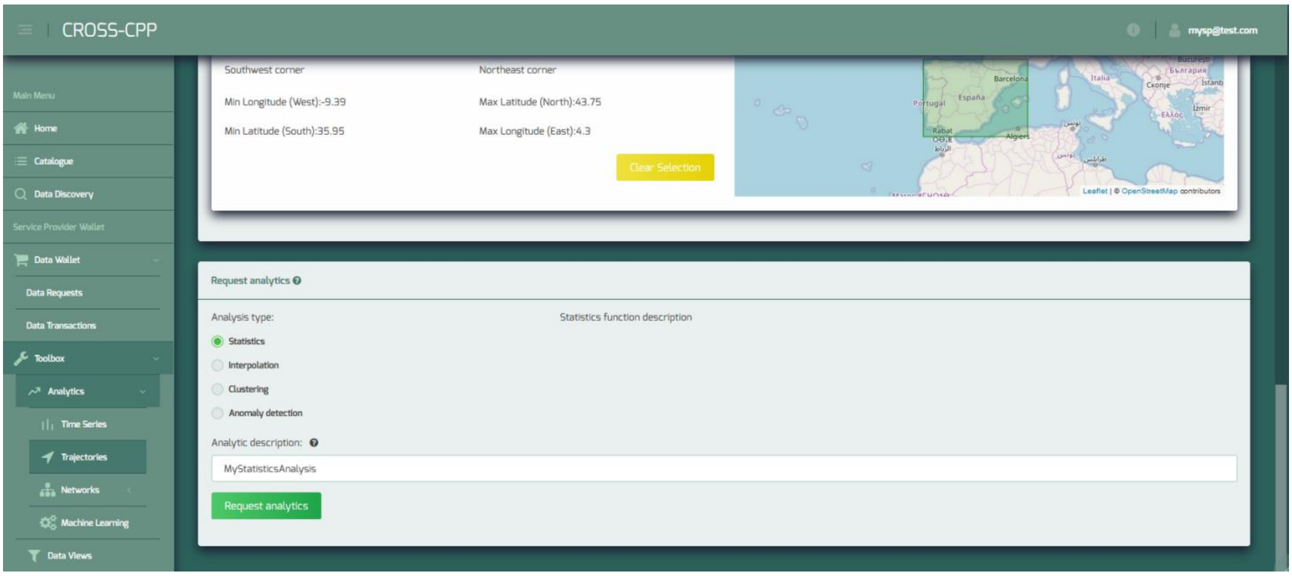


Figure 24: Request trajectory analytics – select analysis type

The process collects all the input data of the data request and apply the filters selected to get the input data of the analysis. The process has stages from the creation of the analysis: processing, OK, failed or no data.

If the status of the process is OK, the results are going to be available to see them in the marketplace application and to be recovered by service provider application via the marketplace API.

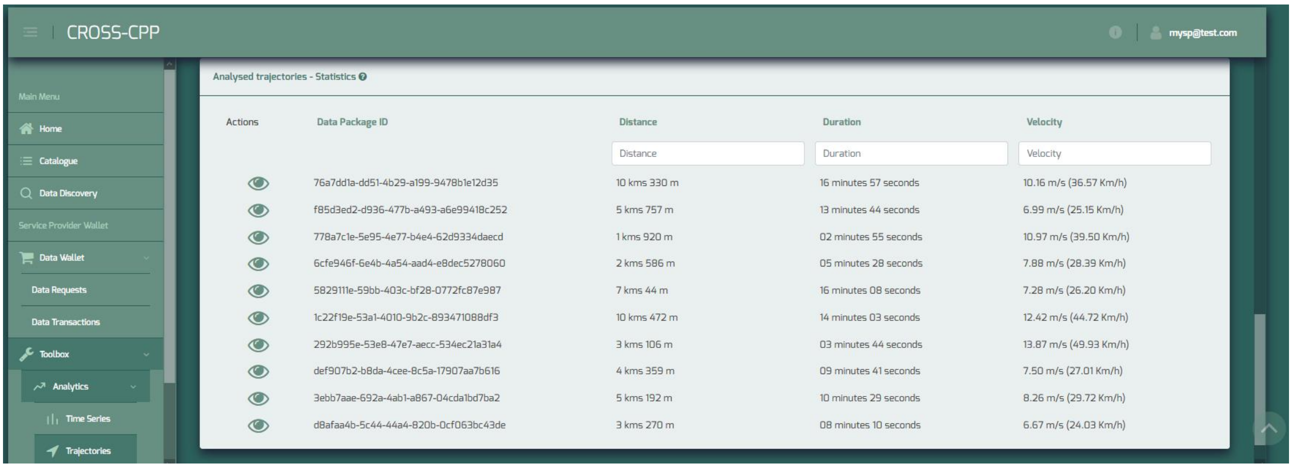


Figure 25: Trajectory analysis results

To retrieve data from the API you need to provide the ID of the analytics to this endpoint https://api.datagora.eu/api/analytics/trajectoryAnalysis/{{id}}

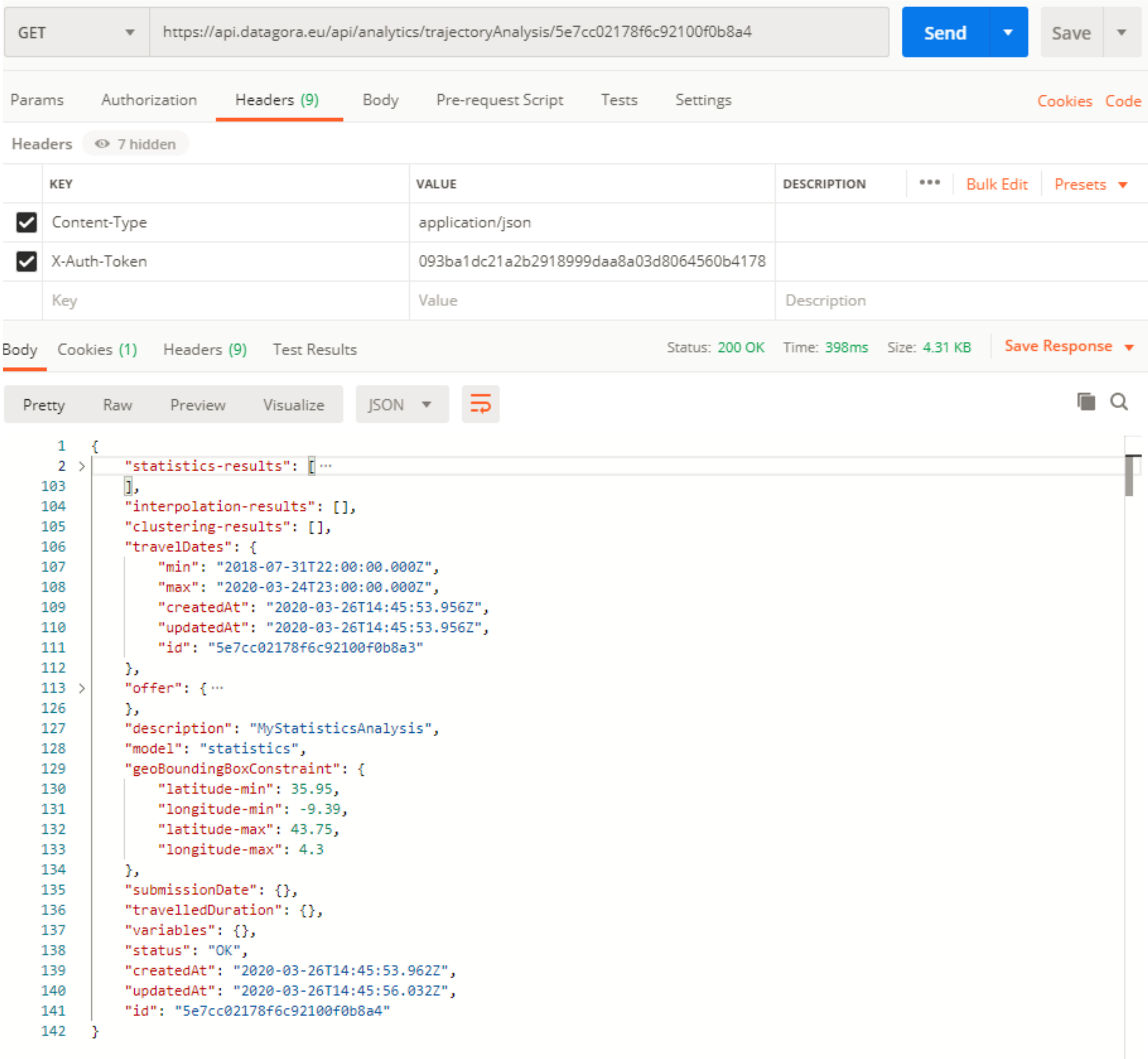


Figure 26: Trajectory analysis API – get analytics information

2.3.3. Networks

The Networks module enables a direct interpretation of the relationship between the signals collected from a specific channel (vehicle or building related). First, analogous to any of the previously detailed functions, the generation of a data request is required to be able to generate a network.

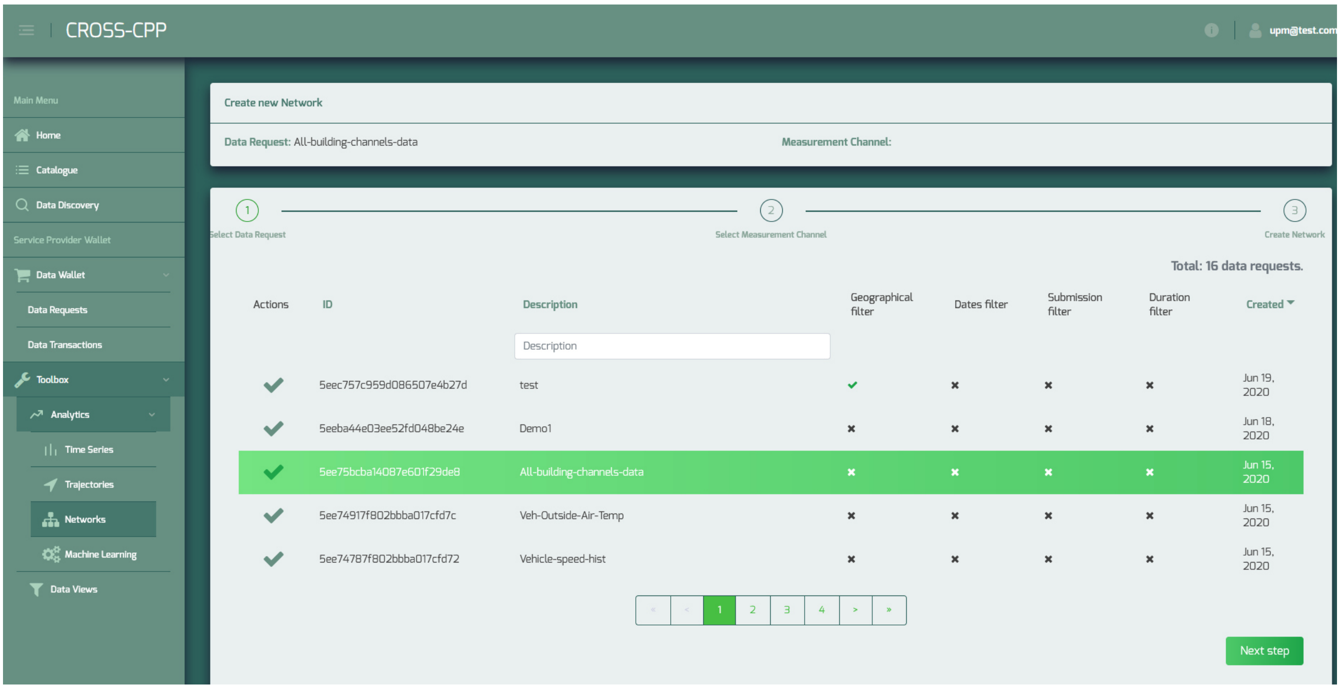


Figure 277: Request Networks module – Data Request Generation

The selected Data Request must have been generated in previous steps (analogous to the procedure indicated in the Trajectories Module).

After selecting the desired Data Request, the user must indicate which of the channels comprised in it is going to be analysed.

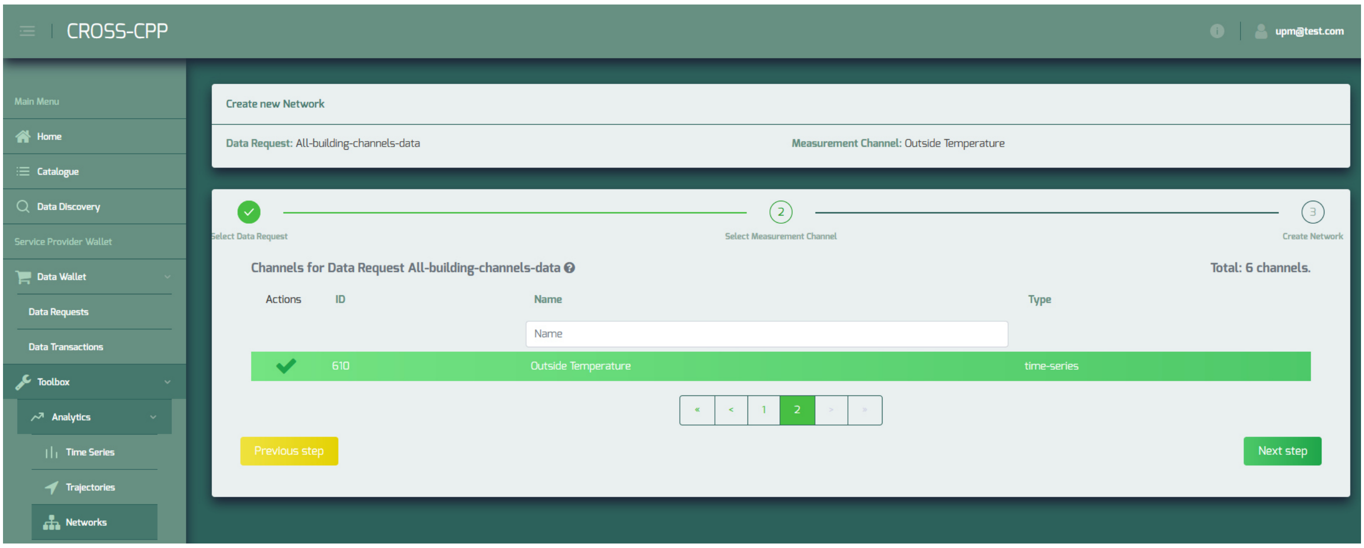


Figure 288: Request Networks module – select signal

Note that, as this module revolves around the idea of relationship between signals from a specific channel, only channels comprising more than one signal will be accepted as input to the network-generation process. Whether or not the selected channel fulfils this requirement is indicated in the interface.

As a final step, just click the “Create” button to proceed with the creation of the network.

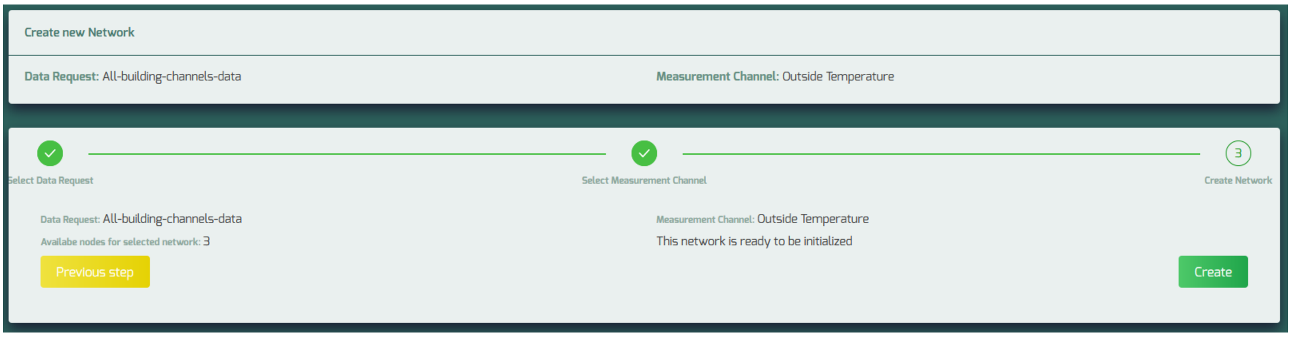


Figure 299: Request trajectory analytics – select analysis type

After the process is completed, the user will be presented an interactive visualization of the network as connected nodes (signals). A list of metrics regarding the structure of the network and the distribution of its nodes is also provided.

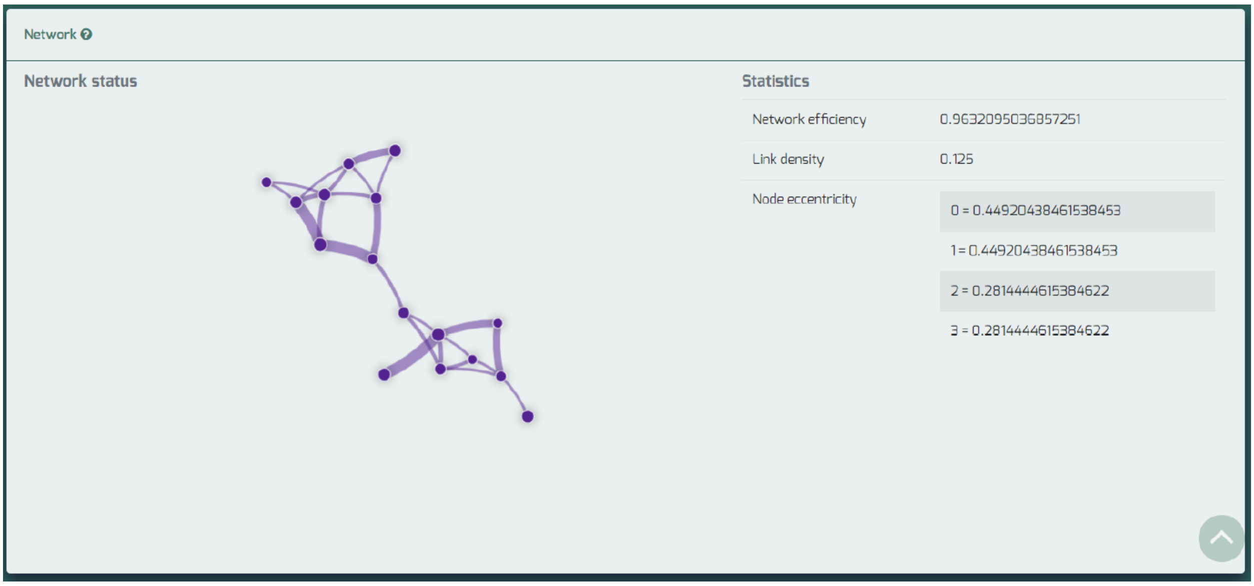


Figure 30: Request trajectory analytics – select analysis type

2.3.4. Machine Learning

Machine learning functions are based on API, which is specified in [OpenApi Specification version 3 format](). Functions are divided into following categories:

- Initial machine-learning processing function (build model)
- Common control functions (status of process, kill process, drop model, export or import model)
- Main machine-learning processing functions (update model, apply model and evaluate model)

All of main machine-learning processing functions need channel data in context for their processing. This is done by obtaining data from data views. Thus, first of all, it is needed to create particular data views for these functions.

In case of classification, there is needed to create category column by endpoint /sei/category/create and its filling by endpoint /sei/category/assign.

2.3.4.1. Initial machine-learning processing function

2.3.4.1.1. Create (build) machine-learning model

The basis of remaining main machine-learning processing operations is to create (build) machine-learning model. Model is created by endpoint ml_model/batch/build, which returns unique identifier of this model, which is used in other endpoints as an input argument.

Example – Request for creation of machine learning model for channels 1 (speed), 2 (location) and 150 (rain intensity):

```
{
  "algname": "sklearn.neureal_network.MLPClassifier",
  "config": {},
  "data": {
    "view_id": "demo_driver_behavior_train_dataset",
    "limit": 100000,
    "page": 0
  }
}
```

Example – Response for creation of machine learning model for channels 1 (speed), 2 (location) and 150 (rain intensity):

```
{
  ″status″: ″success″,
  ″data″: {
    ″alguuid″: ″e85986bc-e80f-4ba8-b9c3-d2e1010b6755″
  }
}
```

2.3.4.2. Common control functions

2.3.4.2.1.1. Status of process

When this request was sent, it can be found out the state of model creation process by endpoint /ml_model/status, where we can find, if the creation process already running, was successfully finished or was finished with some error.

Example – Request for status of machine learning model creation process for channels 1 (speed), 2 (location) and 150 (rain intensity):

```
{
  ″alguuid″: ″e85986bc-e80f-4ba8-b9c3-d2e1010b6755″
```

}

Example – Response for status of machine learning model creation process for channels 1 (speed), 2 (location) and 150 (rain intensity):

{
  "status": "success",
  "data": {
    "status": "finished"
  }
}

2.3.4.2.2. Kill a process

In case that creation process still running and we need to stop this process, we can use endpoint /ml_model/kill – this endpoint only stops the processing of model creation, an entry of model creation is still preserved.

Example – Request of kill some machine learning model creation process:

{
  "alguuid": "01720cf6-d4d4-4e4f-93bc-76de3e08f8c3"
}

Example – Response of kill some machine learning model creation process:

{
  "status": "success",
  "data": []
}

2.3.4.2.3. Drop a process

If the creation process is no longer running, it is possible to drop it /ml_model/drop. Otherwise it is needed to kill it by previously mentioned endpoint.

Example – Request of drop some machine learning model entry:

{

```
  ″alguuid″: ″01720cf6-d4d4-4e4f-93bc-76de3e08f8c3″
}
```

Example – Response of drop some machine learning model entry:

```
{
  ″status″: ″success″,
  ″data″: []
}
```

2.3.4.2.4. Export model

Once a model is created, it may be exported by endpoint /ml_model/export to its internal representation, from which can be imported again. It can be used for backup or experimentation purposes.

Example – Request of export some machine learning model entry:

```
{
  ″alguuid″: ″01720cf6-d4d4-4e4f-93bc-76de3e08f8c3″
}
```

Example – Response of export some machine learning model entry is a file with following content:


```
{
  "type": "batch",
  "algname": "sklearn.neural_network.MLPClassifier",
  "config": {},
  "data": {
    "sei_uri_base": "https:\/\/vian-dev.fit.vutbr.cz\/cross-cpp\/",
    "view_id": "demo_driver_behavior_train_dataset",
    "limit": 100000,
    "page": 0
  },
  "model":
"\\200\u0003csklearn.neural_network.multilayer_perceptron\nMLPClassifier\nq\\000
)\\201q\u0001}q\u0002(X\n\\000\\000\\000activationq\u0003X\u0004\\000\\000\\000r
eluq\u0004X\u0006\\000\\000\\000solverq\u0005X\u0004\\000\\000\\000adamq\u0006X\
u0005\\000\\000\\000alphaq\u0007G?\u001a6\\342\\353\u001cC-
X\n\\000\\000\\000batch_sizeq\bX\u0004\\000\\000\\000autoq\tX\r\\000\\000\\000le
arning_rateq\nX\b\\000\\000\\000constantq\u000bX\u0012\\000\\000\\000learning_ra
te_initq\fG?PbM\\322\\361\\251\\374X\u0007\\000\\000\\000power_tq\rG?\\340\\000\
\000\\000\\000\\000\\000X\b\\000\\000\\000max_iterq\u000eK\\310X\u0004\\000\\000
\\000lossq\u000fX\b\\000\\000\\000log_lossq\u0010X\u0012\\000\\000\\000hidden_la
yer_sizesq\u0011Kd\\205q\u0012X\u0007\\000\\000\\000shuffleq\u0013\\210X\f\\000\
\000\\000random_stateq\u0014NX\u0003\\000\\000\\000tolq\u0015G?\u001a6\\342\\353
```

```
\u001cC-
X\u0007\\000\\000\\000verboseq\u0016\\211X\n\\000\\000\\000warm_startq\u0017\\21
1X\b\\000\\000\\000momentumq\u0018G?\\354\\314\\314\\314\\314\\314\\315X\u0012\\
000\\000\\000nesterovs_momentumq\u0019\\210X\u000e\\000\\000\\000early_stoppingq
\u001a\\211X\u0013\\000\\000\\000validation_fractionq\u001bG?\\271\\231\\231\\23
1\\231\\231\\232X\u0006\\000\\000\\000beta_1q\u001cG?\\354\\314\\314\\314\\314\\
314\\315X\u0006\\000\\000\\000beta_2q\u001dG?\\357\\367\\316\\331\u0016\\207+X\u
0007\\000\\000\\000epsilonq\u001eG>Ey\\216\\3420\\214:X\u0010\\000\\000\\000n_it
er_no_changeq\u001fK\nX\u0010\\000\\000\\000_label_binarizerq
csklearn.preprocessing.label\nLabelBinarizer\nq!)\\201q\"}q#(X\t\\000\\000\\000n
eg_labelq$K\\000X\t\\000\\000\\000pos_labelq%K\u0001X\r\\000\\000\\000sparse_out
putq&\\211X\u0007\\000\\000\\000y_type_q'X\n\\000\\000\\000multiclassq(X\r\\000\
\000\\000sparse_input_q)\\211X\b\\000\\000\\000classes_q*cnumpy.core.multiarray\
n_reconstruct\nq+cnumpy\nndarray\nq,K\\000\\205q-
C\u0001bq.\\207q\/Rq0(K\u0001K\u0004\\205q1cnumpy\ndtype\nq2X\u0002\\000\\000\\0
00U9q3K\\000K\u0001\\207q4Rq5(K\u0003X\u0001\\000\\000\\000<q6NNNK$K\u0004K\btq7
b\\211C\\220d\\000\\000\\000a\\000\\000\\000n\\000\\000\\000g\\000\\000\\000e\\0
00\\000\\000r\\000\\000\\000o\\000\\000\\000u\\000\\000\\000s\\000\\000\\000h\\0
00\\000\\000a\\000\\000\\000z\\000\\000\\000a\\000\\000\\000r\\000\\000\\000d\\0
00\\000\\000o\\000\\000\\000u\\000\\000\\000s\\000\\000\\000n\\000\\000\\000o\\0
00\\000\\000r\\000\\000\\000m\\000\\000\\000a\\000\\000\\000l\\000\\000\\000\\00
0\\000\\000\\000\\000\\000\\000\\000\\000\\000\\000\\000r\\000\\000\\000i\\000\\
000\\000s\\000\\000\\000k\\000\\000\\000y\\000\\000\\000\\000\\000\\000\\000\\00
0\\000\\000\\000\\000\\000\\000\\000\\000\\000\\000\\000q8tq9bX\u0010\\000\\000\
\000_sklearn_versionq:X\u0006\\000\\000\\0000.21.1q;ubh*h0X\n\\000\\000\\000n_ou
tputs_q<K\u0004X\r\\000\\000\\000_random_stateq=cnumpy.random\n__RandomState_cto
r\nq>)Rq?(X\u0007\\000\\000\\000MT19937q@h+h,K\\000\\205qAh.\\207qBRqC(K\u0001Mp
\u0002\\205qDh2X\u0002\\000\\000\\000u4qEK\\000K\u0001\\207qFRqG(K\u0003h6NNNJ\\
377\\377\\377\\377J\\377\\377\\377\\377K\\000tqHb\\211B\\300\t\\000\\000\\243S\\
377\u0011\\211\u0010\\334\u0018X\\327\\314\\267{\\363\\303\\3701\\364\\372]\u001
9\\224]D\\361t\u0007\\366\\273\\271\\351\\255g\\266+\\201Q4y^\/\\254\\277\\247\\
276\\262\\314}c\\351)\\230\\250\\342\\212\\336\\3251\\273\\353\\212\\316\\314]\\
212\u0011d\u001f(\\227\\225\\367\\203\\211:Y\\360\\340oWSH\\317\\374L\"%Z\\373\\
333v\\214\\313\u0010D\\257$\\234\\344\\212\\277\u001b\\316%\\365\u0007g[u\\245\\
2654E9\\273\\304\\332\\275%\\214\u0006\\215\u0003\u0004\\332%\\264\\314\\247\t\u
0005\\251\\367\\265\\344\\273F\\320\\373\\366^]\\322\\355_\\365RL\\202Lkv\u001f\
\325l\\345\\371\\235Qa\\225\\250\\353\\230\\276:\u000e|\\000\\342x]7{f\\356{\\27
7q\\222~\\3204\\360x\\216-
\u001b\\3175\/\\237\u0014\\271\\250[\\235\\226_\\354\u0007\\220\\000\\301A\u001e
\\360 \\277\\235\u001d@ia\\307\u000f\\2707M\\267\\374\u0018\nW\\
[...]
Eq\\321\\246\u0004r4?`i5\\325\\367\\277\\257>:\\345\\261\\275'\\322O?\\275\\253I
\\223a\u001aG?r\u0005\u0002\\000\\000tr\u0006\u0002\\000\\000beubX\u0005\\000\\0
00\\000loss_r\u0007\u0002\\000\\000j\\306\u0001\\000\\000h:h;ub."
}
```


2.3.4.2.5. Import model

Instead of creating a model by build endpoint you can also use import endpoint (/ml_model/import) to import model from internal representation created from the previous use of export endpoint

Example – Request of import some machine learning model entry:

```
{
  "type": "batch",
```

```
  "algname": "sklearn.neural_network.MLPClassifier",
  "config": {},
  "data": {
    "sei_uri_base": "https://vian-dev.fit.vutbr.cz/cross-cpp/",
    "view_id": "demo_driver_behavior_train_dataset",
    "page": 100000,
    "limit": 0
  },
  "model":"\\200\u0003csklearn.neural_network.multilayer_perceptron\n
MLPClassifier\nq\\000)\\201[...]"
}
```

Example – Response of import some machine learning model entry:

```
{
  "status": "success",
  "data": {
    "alguuid": "91b33944-c0d3-4115-89e3-d3b148465805"
  }
}
```

2.3.4.3. Main machine-learning processing functions

2.3.4.3.1. Update machine-learning model

Already created model can be improved by extra data (for example other page from training data or another data view) by /ml_model/batch/update endpoint. This endpoint creates new model and assign to it new alguuid identifier, which is returned.

Example – Request of update some machine learning model:

```
{
  "alguuid": "e85986bc-e80f-4ba8-b9c3-d2e1010b6755",
  "config": {},
  "data": {
```

```
    "view_id": "demo_driver_behavior_train_dataset",
    "limit": 100000,
    "page": 1
  }
}
```

Example – Request of update some machine learning model:

```
{
  "status": "success",
  "data": {
    "alguuid": "a94ee08e-90cb-4c9e-b043-0ba70d8d584f"
  }
}
```

2.3.4.3.2. Apply machine-learning model to testing data

Once a model is trained, we can apply it to testing data to predict some value or classify to some class by /ml_model/batch/apply endpoint.

Example – Request of apply some machine learning model to testing data:

```
{
  "alguuid": "e85986bc-e80f-4ba8-b9c3-d2e1010b6755",
  "data": {
    "view_id": "demo_driver_behavior_test_dataset",
    "limit": 1000,
    "page": 0
  }
}
```

Example – Request of apply some machine learning model to testing data:

```
{
  "status": "success",
  "data": [
    ...
    {
```

"1": "69.79254800283908",
"150": "0.8944514838653745",
"2_Latitude": "49.23487606034672",
"2_Longitude": "16.591834275131614",
"ml_prediction_output": "normal"
},
{
"1": "75.11103007668163",
"150": 0,
"2_Latitude": "49.23502669317959",
"2_Longitude": "16.591701072103056",
"ml_prediction_output": "normal"
},
{
"1": "69.7968282329311",
"150": "0.7510775046095213",
"2_Latitude": "49.235188804860044",
"2_Longitude": "16.591557718450403",
"ml_prediction_output": "dangerous"
},
{
"1": "70.80922943008888",
"150": 0,
"2_Latitude": "49.23533944693091",
"2_Longitude": "16.591424507252782",
"ml_prediction_output": "normal"
},
{

```
      "1": "72.56495706486243",
      "150": "0.5417261819678416",
      "2_Latitude": "49.23549227406114",
      "2_Longitude": "16.59128936383017",
      "ml_prediction_output": "risky"
    },
    {
      "1": "70.2379894739633",
      "150": "0.4606970283059384",
      "2_Latitude": "49.23564889056766",
      "2_Longitude": "16.59115086950198",
      "ml_prediction_output": "normal"
    },
    ...
  ]
}
```

2.3.4.3.3. Evaluate machine-learning model to evaluation data

To obtain a quality of trained machine-learning model we can evaluate it by /ml_model/batch/evaluate endpoint.

Example – Request of evaluate some machine learning model according to evaluation data:

```
{
  "alguuid": "01720cf6-d4d4-4e4f-93bc-76de3e08f8c3",
  "config": {},
  "data": {
    "view_id": "demo_driver_behavior_eval_dataset",
    "limit": 5000,
    "page": 0
  }
}
```

Example – Request of evaluate some machine learning model according to evaluation data:

{
  "status": "success",
  "data": {
    "score": 0.8614691276516347
  }
}

3. API

The API url (from now on *api_url*) is always: **https://api.datagora.eu/**

An online reference can be found in https://swagger.datagora.eu/ under sections:

- Catalogue: signals and measurement channels available
- CIDM data: request of active data requests data received
- Service Providers: requests service provider profile, data requests and data transactions
- Discovery: perform a filtered search of data available in the marketplace
- Analytics: different requests regarding Analytics Toolbox

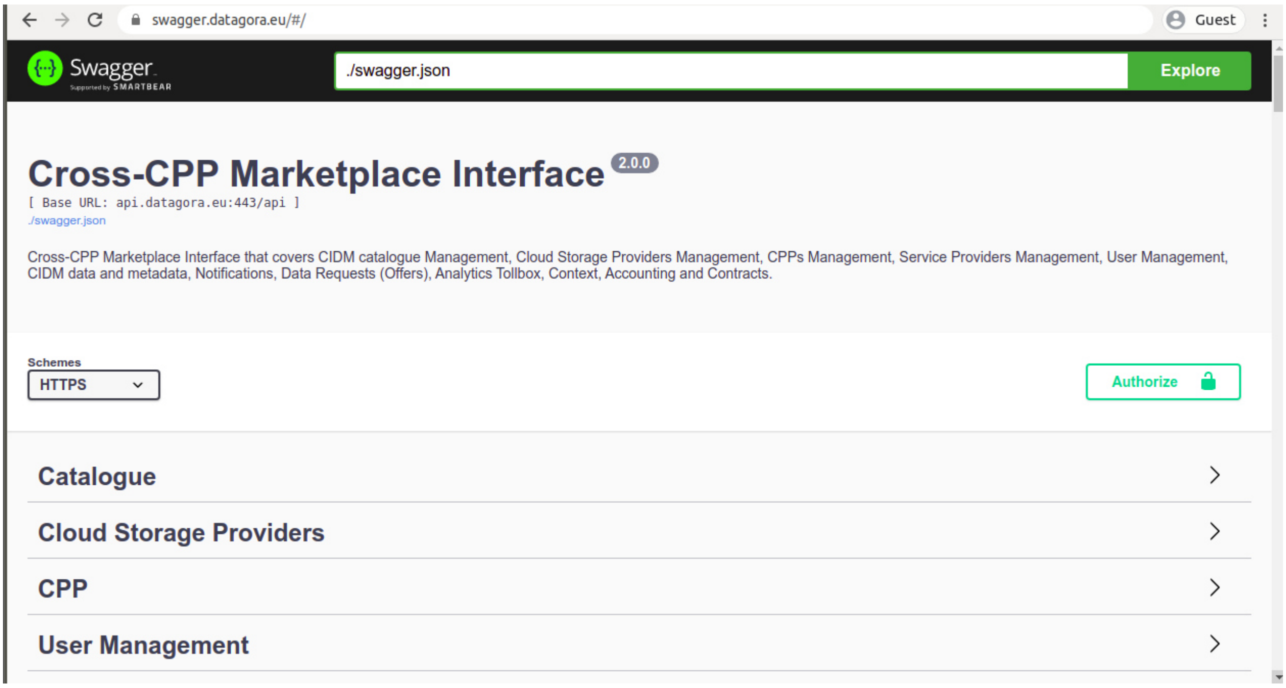


Figure 30. Cross-CPP OpenAPI Specification

3.1. First Steps

3.1.1. Authentication

The Cross-CPP Marketplace API is secured by an Identity Manager (IdM) and OAuth v2 protocols, meaning the endpoints cannot be accessed without first identifying and retrieving an access token.

The IdM has its own server, to secure resource allocation, and therefore has a different url.

Once the token is retrieved it can be used to access the Marketplace resources. All Marketplace API requests must include the following headers:

Parameter	Value	Required	Type	Format	Description
X-Auth-Token	{{access_token}}	Always	string	RFC 6750	OAuth2 token provided by IDM

					component
Content-type	application/json	POST requests	-	-	-

Table 1. Request headers

3.1.2. Get Access Token

This request to the IdM allows the service provider to generate a token to access the Marketplace resources.

To access the IdM resources the request needs the client id and secret that will be provided by Cross-CPP administrators. The Authorization header is a Basic auth generated using the client id and client secret provided.

REQUEST	**GET ACCESS TOKEN**	
Method	POST	
Url	https://idm.datagora.eu	
Endpoint	/oauth2/token	
Headers	Content-Type	application/x-www-form-urlencoded
	Authorization	Basic …DE4Yg==
Body	grant_type	"password"
	username	string
	password	string

Table 2. Get Access Token

Responses:

Code		**Description**
200	OK	Request succesful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing client_id or incorrect
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body example:

```
{
    "access_token": "5FOW3KmS7Jv1y3j5ARnabzNryn7kV3",            // string
    "token_type": "Bearer",                                        // string
    "expires_in": 3600,                                            // integer
    "refresh_token": "6pRcNbpaHkMuH2rFRF2feuYllsCX1L"              // string
}
```

Listing 1. Get Access Token response ok

The “access_token” parameter of the response needs to be stored to be used as an authenticator in every Marketplace request header with key ***X-Auth-Token***.

3.1.3. Service Provider Profile

This request allows the service provider to retrieve its profile, containing the Service provider ID (serviceProviderId from now on), which is needed for most of the service provider requests.

REQUEST	GET USER PROFILE	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/UserProfile	
Headers	X-Auth-Token	{{access_token}}

Table 3. Get user profile request

Responses:

Code		Description
200	OK	Request succesful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok fresponse body schema:

```
{
    "serviceProviders": [
        {
            "name": string,
            "email": string,
            "createdAt": date,
            "updatedAt": date,
            "id": id
        }
    ],
    "roles": [
        "serviceProvider"
    ],
    "email": string
```

```
}
```

Listing 2. Get user profile response ok

The "serviceProviders.id" parameter of the response needs to be stored to be used as a parameter in most of the Service Provider requests in the Marketplace.

3.2. General Purpose API

This API can be accessed by any authenticated user in the Marketplace. It includes general information functions such as listing available signals and perform a search through the Data Discovery Component.

3.2.1. Catalogue

This section includes functions regarding the retrieval of available signals information.

3.2.1.1. Signal Catalogue

This request allows the user to retrieve the list of currently available signals.

REQUEST	GET CATALOGUE	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/catalogue/signals	
Headers	X-Auth-Token	{{access_token}}

Table 4. Get Catalogue request

Responses:

Code		Description
200	OK	Request succesful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body example:

```
[
    {
        "id": "---",                                    // id
        "name": "Vehicle speed",                        // string
```

```
        "type": "numeric",                              // enumeration string
        "cpp-type": "vehicle",                          // enumeration string
        "unit": "km/h",                                 // string
        "measurement-channel-id": "1",                  // string
        "measurement-channel-name": "Vehicle Speed",    // string
        "measurement-channel-type": "time-series"       // enumeration string
    },
    {
        "id": "---",
        "name": "Latitude",
        "type": "numeric",
        "cpp-type": "vehicle",
        "unit": "°",
        "measurement-channel-id": "2",
        "measurement-channel-name": "Position (Latitude Longitude)",
        "measurement-channel-type": "time-series"
    },
    ...
]
```

Listing 3. Get Catalogue response ok

Each entity in the response array represents a relationship between a signal being sampled by a measurement channel. Both signal and channel ids can be found inside each entity.

3.2.1.2. Signal

This request allows the user to retrieve a Signal details.

REQUEST	**GET SIGNAL**	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/signal/{{signalId}}	
Headers	X-Auth-Token	{{access_token}}

Table 5. Get single Signal request

Responses:

Code		**Description**
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body example for "signalId = 'Vehicle speed' uuid":


```
{
    "channels": [
        {
            "measurement-channel-id": "1",                 // string
            "name": "Vehicle Speed",                       // string
            "type": "time-series",                         // enumeration string
            "on-change": false,                            // boolean
            "dimensions": 1,                               // integer
            "format": "Double",                            // string
            "sample-strategy": "last-known-value",         // enumeration string
            "capture-interval": 0,                         // number
            "comment": "...",                              // string
            "createdAt": "2018-03-03T14:50:32.467Z",       // date
            "updatedAt": "2018-04-05T15:30:59.811Z",       // date
            "id": "---"                                    // id
        },
        {
            "measurement-channel-id": "151",
            "name": "Vehicle speed Histogram",
            "type": "histogram",
            "aggregation-strategy": "count",
            "capture-interval": 1,
            "dimensions": 1,
            "comment": "Counter will never be resetted",
            "createdAt": "2018-03-03T14:51:21.943Z",
            "updatedAt": "2018-03-28T09:24:33.203Z",
            "id": "---"
        }
    ],
    "name": "Vehicle speed",                               // string
    "type": "numeric",                                     // string
    "unit": "km/h",                                        // string
    "sample-rate": "differs",                              // enumeration string
    "createdAt": "2018-03-03T14:50:32.365Z",               // date
    "updatedAt": "2018-04-05T15:30:59.279Z",               // date
    "cpp-type": "vehicle",                                 // enumeration string
    "id": "---"                                            // id
}
```


Listing 4. Signal response

A signal entity will be sent with all the channels that sample the data received by the signal.

3.2.1.3. Signals

This request allows the user to retrieve a list of signals.

REQUEST	GET SIGNALS
Method	GET

Url	https://api.datagora.eu/api	
Endpoint	/signal	
Headers	X-Auth-Token	{{access_token}}

Table 6. Get all signals request

Responses:

Code		Description
200	OK	Request succesful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of signals. See 3.2.1.2 Signal above

3.2.1.4. Channel

This request allows the user to retrieve a Measurement Channel details.

REQUEST	GET MEASUREMENT CHANNEL	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/measurementchannel/{{measurement_channel_id}}	
Headers	X-Auth-Token	{{access_token}}

Table 7. Get single measurement channel request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user

404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body example for "measurement-channel-id = 1":


```
{
    "signal": [
        {
            "name": "Vehicle speed",                  // string
            "type": "numeric",                        // enumeration string
            "unit": "km/h",                           // string
            "sample-rate": "differs",                 // enumeration string
            "createdAt": "2018-03-03T14:50:32.365Z",  // date
            "updatedAt": "2018-04-05T15:30:59.279Z",  // date
            "cpp-type": "vehicle",                    // enumeration string
            "id": "---"                               // id
        }
    ],
    "measurement-channel-id": "1",                    // string
    "name": "Vehicle Speed",                          // string
    "type": "time-series",                            // enumeration string
    "on-change": false,                               // boolean
    "dimensions": 1,                                  // integer
    "format": "Double",                               // string
    "sample-strategy": "last-known-value",            // enumeration string
    "capture-interval": 0,                            // number
    "comment": "...",                                 // string
    "createdAt": "2018-03-03T14:50:32.467Z",          // date
    "updatedAt": "2018-04-05T15:30:59.811Z",          // date
    "id": "---"                                       // id
}
```


Listing 5. Channel response

3.2.1.5. Channels

This request allows the user to retrieve a list of signals.

REQUEST	GET MEASUREMENT CHANNELS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/measurementchannel	
Headers	X-Auth-Token	{{access_token}}

Table 8. Get all channels request

Responses:

Code	Description

200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of measurement channels. See 3.2.1.4 Channel above

3.2.2. Data Discovery

This request allows the user to get a resume of available data within a configuration of filters.

REQUEST	**DATA DISCOVERY**	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/VehicleDataDiscovery	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	{ "channels": [], // string array "submissionDate": {}, // Object "min": "", // Number "max": "", // Number }, "travelDates": {}, // Object "min": "", // Number "max": "", // Number } "travelledDuration": {}, // Object "min": "", // Number "max": "", // Number } "geoBoundingBox": {}, // Object "latitude-min": "", // Date "longitude-min": "", // Date "latitude-max": "", // Date "longitude-max": "", // Date } "includeContextData": false, // boolean "basicCppInformationFilters": [] // Object array }	

Table 9. Data Discovery request

Body explanation:

** for every filter: if sent empty that filter will be count as NO FILTER (retrieve all)*

- **channels**: Array of “measurement-channel-id”. Filters data packages to those from selected channels.
- **submissionDate**: Object representing the dates in which the data has been stored in the marketplace, including max and min values between which data will be searched.
- **travelDates**: Object representing the dates in which the data has been generated by the signal, including max and min values between which data will be searched.
- **travelledDuration**: Object representing the duration in seconds of each data package, including max and min values between which data will be searched.
- **geoBoundingBox**: Object representing the geographical rectangular area within which the data is desired, including longitude and latitude max and min values.
- **includeContextData**: Boolean activating the receiving of additional context data.
- **basicCppInformationFilters**: Array of objects representing additional filters based on basic-cpp-information signals, otherwise known as CPP metadata. These objects include the channel, name and value of the filters, such as “red car” (see example)

Body example:

This example represents a search for data packages belonging to:

- Vehicle speed and location
- Taken and submitted in August 2019
- Not longer than 3 seconds
- Within the geographical area of Austria
- Without context data

```
"channels": [
    "1",
    "2"
],
"submissionDate": {
    "min": "2019-08-01",
    "max": "2019-08-31"
},
"travelDates": {
    "min": "2019-08-01",
    "max": "2019-08-31"
},
"travelledDuration": {
    "min": 0,
    "max": 3
},
"geoBoundingBox": {
    "longitude-min": 9.48,
    "latitude-min": 46.43,
    "longitude-max": 16.98,
    "latitude-max": 49.04
},
"includeContextData": false,
"basicCppInformationFilters": [
    {
        "channel": {
```

```
                "id": "---",
                "name": "Vehicle Colour",
                "type": "information",
                "cpp-type": "vehicle",
                "measurement-channel-id": "702",
                "measurement-channel-name": "Vehicle Colour",
                "measurement-channel-type": "basic-cpp-information"
            },
            "name": "Vehicle Colour",
            "cpp-type": "vehicle",
            "measurement-channel-id": "702",
            "value": "red"
        }
    ]
}
```

Listing 6. Discovery request example

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body with no data:

```
{
    "data-packages": 0
}
```

Request ok response body example with data:

```
{
    "data-packages": 15733,                                 // integer
    "rooms": 0,                                             // integer
    "data-packages-trip": 1000,                             // integer
    "trips": 1000,                                          // integer
    "cpp-owners": 8,                                        // integer
    "duration-histogram": {
        "min-duration": 0,                                  // integer
        "max-duration": 3844800,                            // integer
        "bins-size": 3600,                                  // integer
        "bins": [ ... ],                                    // integer array
    },
    "channel-results": [                                    // Object array
        {
            "measurement-channel-id": "2",                  // string
            "count": 2930                                   // integer
        },
        ...
    ],
```

```
 "cpp-type-results": [                                      // Object array
     {
         "minDuration": 1,                                  // integer
         "maxDuration": 1527,                               // integer
         "avgDuration": 478.2052477357485,                  // double
         "cpp-type": "vehicle",                             // enumeration string
         "count": 15016                                     // integer
     },
     { ... }
 ],
 "room-results": [],                                        // Object array
 "cpptype-channel-results": [
     {
         "channel": [                                       // Object array
             {m
                 "measurement-channel-id": "1",             // string
                 "count": 2813                              // integer
             },
             { ... }
         ],
         "cpp-type": "vehicle",                             // string
         "count": 15016                                     // integer
     },
     { ... },
],
 "total-channel-results": 25,                               // integer
 "total-cpp-type-results": 2,                               // integer
 "total-room-results": 0,                                   // integer
 "time-distribution": {
     "min-timestamp-start": "2017-10-03T13:03:13.000Z",     // date
     "max-timestamp-stop": "2019-07-17T14:30:02.000Z",      // date
     "histogram": {
         "timestamp-start": "2017-10-03T00:00:00.000Z",     // date
         "timestamp-stop": "2019-07-17T23:59:59.000Z",      // date
         "bins-size": 86400000,                             // integer
         "bins": [ ... ],                                   // integer array
     }
},
 "geo-distribution": {
     "latitude-min": 48.599998474121094,                    // double
     "latitude-max": 51.522789001464844,                    // double
     "longitude-min": 3,                                    // double
     "longitude-max": 18.35022735595703,
     "heatmap": {
         "latitude-min": 48.5,                              // double
         "latitude-max": 51.6,                              // double
         "longitude-min": 3,                                // double
         "longitude-max": 18.400000000000002,               // double
         "bins-size": 0.1,                                  // double
         "bins": [                                          // Object array
             {
                 "latitude": 51.48193359375,                // double
                 "longitude": 7.44873046875,                // double
                 "count": 4848                              // integer
             },
             { ... }
         ]
```

```
        }
    }
}
```

Listing 7. Discovery response example

3.2.3. Channel Suggestions

This request allows the user to get a list of suggested channels based on the already selected channels during data discovery process.

REQUEST	GET ACCESS TOKEN	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/suggestion/suggestions	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	`{ "channels": [], // string array }`	

Table 10. Get channel suggestions

Body explanation:

**if sent empty, no results will be given*

Channels: Array of "measurement-channel-id". Filters data packages to those from selected channels.

Body example:

This example partly represents a request for suggestions related to:

- Vehicle speed

```
{
    "channels": [
        "1"
    ],
}
```

Responses:

Code		Description
200	OK	Request succesful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role

		Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
[
    {
        "requestedSignal": "VehicleSpeed",                          // string
        "relatedSignals": [                                         // signal-channel array
            {
                "signal-type": "Sensor Measurement Data",           // string
                "cpp-type": "Vehicle",                              // string
                "signal-name": "Wheel RPM - front left",            // string
                "measurement-channel-id": "115",                    // string
                "measurement-channel-type": "time-series",          // string
                "measurement-channel-name": "Wheel RPM - front left" // string
            },
            {
                "signal-type": "Sensor Measurement Data",
                "cpp-type": "Vehicle",
                "signal-name": "Wheel RPM - front right",
                "measurement-channel-id": "116",
                "measurement-channel-type": "time-series",
                "measurement-channel-name": "Wheel RPM - front right"
            },
            { ... }
        ]
    }
]
```

Listing 8. Channel suggestions response

3.3. Service Provider API

This API can be accessed by Service Providers authenticated in the Marketplace. It includes Data Requests, Data Transactions and Analytics related requests that allow Service Providers to consult, review and manage their data.

3.3.1. Data Requests

Data Requests are the main point for Service Providers. Through them all available data fitting each one filter configuration is sent.

3.3.1.1. Create Data Request

To create a Data Request a Service Provider should perform a Data Discovery first in order to check if results fits with what is desired, although a Data Request can be created even if no data is retrieved with the Data Discovery.

REQUEST	CREATE DATA REQUEST	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProviderOffer	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	{ "description": "", // string "serviceProvider": "", // uuid "include-context": true, // boolean "include-analytics": true, // boolean "vehicleDataRequest" {, // data discovery model "channels": [], // string array "submissionDate": {}, // Object "travelDates": {}, // Object "travelledDuration": {}, // Object "geoBoundingBox": {}, // Object "includeContextData": false, // boolean "BasicCppInformationFilters": [] // Object array } }	

Table 11. Create Data Request request

Body explanation:

- **description**: the name / short description so data owners know in a glance what it might be requesting
- **serviceProvider**: identificator of the Service Provider
- **include-context**: boolean stating whether the data collected from this request includes context data or not
- **include-analytics**: boolean stating whether the data collected from this request will be used in the creation of analytics or not
- **vehicleDataRequest**: use Data Discovery model.

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "vehicleDataRequest": "--- ",                                    // id
    "serviceProvider": "---",                                         // id
    "description": "Postman test offer",                              // string
    "include-context": true,                                          // string
    "include-analytics": true,                                        // string
    "maxNumberOfDataSources": 0,                                      // integer
    "createdAt": "2019-09-23T14:52:35.439Z",                          // string
    "updatedAt": "2019-09-23T14:52:35.439Z",                          // string
    "id": "---",                                                      // id
    "aeonChannelId": "---",                                           // id
    "aeonSubscriptionUrl": "https://{aeon_url}/subscribe/{uuid}",     // string
    "aeonPublicationUrl": "https://{aeon_url}/publish/{uuid}"         // string
}
```

Listing 9. Create data request response

3.3.1.2. Delete Data Request

This method can be used to delete an existing Data Request. This action cannot be undone and will prevent the Service Provider to receive any more data from users that had accepted the request before deleting it.

REQUEST	DELETE DATA REQUEST	
Method	DELETE	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProviderOffer/{{serviceProviderOfferId}}	
Headers	X-Auth-Token	{{access_token}}

Table 12. Delete Data Request request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

3.3.1.3. Get my Data Requests

This request can be used to retrieve all active Data Request belonging to the Service Provider

REQUEST	GET SERVICE PROVIDER DATA REQUESTS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProvider/{{serviceProviderId}}/offers	
Headers	X-Auth-Token	{{access_token}}

Table 13. Get Data Requests request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
[
    {
        "id": "---",                                              // id
        "description": "description",                             // string
        "vehicleDataRequest": {                                   // data request model
            "id": "---"                                           // id
            "channels": [],                                       // string array
            "submissionDate": {},                                 // object
            "travelDates": {},                                    // object
            "travelledDuration": {},                              // object
            "geoBoundingBox": {},                                 // object
        },
        "serviceProvider": {                                      // Service Provider model
            "id": "---",                                          // id
            "name": "",                                           // string
            "email": "@."                                         // string
        },
        "createdAt": "2019-05-09T12:50:00.638Z",                  // date
        "updatedAt": "2019-05-09T12:50:00.926Z",                  // date
        "maxNumberOfDataSources": 0,                              // integer
        "include-context": true,                                  // boolean
        "include-analytics": true,                                // boolean
        "aeonChannelId": "---",                                   // id
```

```
        "aeonSubscriptionUrl": "https://{aeon_url}/subscribe/{uuid}", // string
        "aeonPublicationUrl": "https://{aeon_url}/publish/{uuid}"     // string
    },
    { ... }
]
```

Listing 10. Get my Data Requests

3.3.1.4. Get a Data Request

This request can be used to retrieve a Data Request details

REQUEST	GET DATA REQUEST DETAILS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProviderOffer/{{spOfferId}}	
Headers	X-Auth-Token	{{access_token}}

Table 14. Get Data Request details request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "id": "---",                                                    // id
    "description": "test for tables",                               // string
    "serviceProvider": {                                            // service provider object
        "name": "name",
        "email": "@.",
        "id": "---"
    },
    "createdAt": "2019-06-27T11:21:13.903Z",                        // date-time
    "updatedAt": "2019-06-27T11:21:14.273Z",                        // date-time
    "aeonChannelId": "---",                                         // id
    "aeonSubscriptionUrl": "https://{aeon_url}/subscribe/{uuid}",   // string
    "aeonPublicationUrl": "https://{aeon_url}/publish/{uuid}",      // string
    "include-context": true,                                        // boolean
```

```
    "include-analytics": true,                                      // boolean
    "vehicleDataRequest": {                                         // filter config object
        "channels": [                                               // string array
            "1",
            "2"
        ],
        "travelledDuration": {
            "min": -,                                               // date *
            "max": -                                                // date *
        },
        "submissionDate": {
            "min": "2019-06-03",                                    // date *
            "max": "2019-06-27"                                     // date *
        },
        "travelDates": {
            "min": -,                                               // date *
            "max": -                                                // date *
        },
        "geoBoundingBox": {
            "latitude-max": 58,                                     // number **
            "latitude-min": 38,                                     // number **
            "longitude-max": 25,                                    // number **
            "longitude-min": -5                                     // number **
        }
    },
}
```

Listing 11. Get Data Request details

** If this value is null or undefined the object will be sent but will not contain the key*
*** If any of the values is null or undefined, the object will be sent empty, as if there is no filter*

3.3.1.5. Get Data Request Contracts

This request can be used to retrieve currently active contracts for an active data request (acceptance of the data request from data owners)

REQUEST	GET DATA REQUEST CONTRACTS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProviderOffer/{{spOfferId}}/contracts	
Headers	X-Auth-Token	{{access_token}}

Table 15. Get Data Request contracts request

Responses:

Code		Description
200	OK	Request successful

400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
[                                                               // contracts array
    {
        "id": "---",                                            // id
        "spOffer": "---",                                       // id
        "effectiveDate": "2019-07-03T09:17:38.492Z",            // date-time
        "createdAt": "2019-07-03T09:17:37.787Z",                // date-time
        "updatedAt": "2019-07-03T09:17:37.787Z",                // date-time
        "vehicleOwner": {
            "id": "---",                                        // id
            "name": "Vehicle1",                                 // string
            "email": "@.",                                      // string
            "cloudVaultId": "---",                              // uuid
            "readAccessKey": "---",                             // uuid
            "discoveryClearance": true,                         // boolean
            "testDataClearance": false,                         // boolean
            "aggregationServiceDataClearance": false,           // boolean
            "createdAt": "2019-04-05T11:58:13.188Z",            // date-time
            "updatedAt": "2019-04-05T11:58:13.188Z",            // date-time
            "cloudStorageProvider": {
                "cloudBaseUrl": "https://url.domain",           // string
                "accessToken": "WT4mRvs5w6a7W3KbZN7VeNap",      // string
                "name": "Company Cloud Storage",                // string
                "contactEmail": "@.",                           // string
                "createdAt": "2019-04-04T12:11:29.531Z",        // date-time
                "updatedAt": "2019-04-04T12:11:29.531Z",        // date-time
                "id": "5ca5f471c216151d00fe13e3"                // id
            },
        },
    },
    { ... }
]
```

Listing 12. Get Data Request contracts

3.3.2. Data

Data Requests are sets of configured filters defining which data a service provider will receive through them. The Service Provider can query a resume of this data at any time.

3.3.2.1. Get Data Transactions

This request can be used to retrieve a list of data transactions belonging to the service provider data requests

REQUEST	GET DATA TRANSACTIONS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProvider/{{serviceProviderId}}/dataTransactions	
Headers	X-Auth-Token	{{access_token}}

Table 16. Get Data Transactions resume request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
[
    {
        "serviceProviderOffer": "---",                               // id
        "serviceProvider": "---",                                    // id
        "method": "pull",                                            // string
        "datapackage-id": "---",                                     // uuid
        "accessDatetime": "2019-05-13T13:04:37.473Z",                // date-time
        "size": 1976,                                                // integer
        "datapackageHash": "---=",                                   // encoded string
        "createdAt": "2019-05-13T13:04:37.478Z",                     // date-time
        "updatedAt": "2019-05-13T13:04:37.478Z",                     // date-time
        "id": "5cd96b65ff89151c002d16b3"                             // id
    },
    { ... }
]
```

Listing 13. Get transactions resume

3.3.2.2. Get Data Request Data Transactions

This request can be used to retrieve a list of data transactions belonging to a concrete service provider data request

REQUEST	GET DATA REQUEST TRANSACTIONS
Method	GET

Url	https://api.datagora.eu/api	
Endpoint	/ServiceProviderOffer/{{spOfferId}}/dataTransactions	
Headers	X-Auth-Token	{{access_token}}

Table 17. Get Data Request transactions request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
[
    {
        "serviceProviderOffer": "---",                                // id
        "serviceProvider": "---",                                     // id
        "method": "pull",                                             // string
        "datapackage-id": "---",                                      // uuid
        "accessDatetime": "2019-05-13T13:04:37.473Z",                 // date-time
        "size": 1976,                                                 // integer
        "datapackageHash": "---=",                                    // encoded string
        "createdAt": "2019-05-13T13:04:37.478Z",                      // date-time
        "updatedAt": "2019-05-13T13:04:37.478Z",                      // date-time
        "id": "5cd96b65ff89151c002d16b3"                              // id
    },
    { ... }
]
```

Listing 14. Get Data Request transactions

3.3.2.3. Get Data Package Details

This request can be used to retrieve the details of a single data package.

REQUEST	**GET DATA PACKAGE DETAILS**
Method	GET
Url	https://api.datagora.eu/api

Endpoint	/ServiceProviderOffer/{{spOfferId}}/CvimDataPackages/{{dataPackageId}}	
Headers	X-Auth-Token	{{access_token}}

Table 18. Get Data Package details request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "timestamp-start": "2017-10-03T13:03:13Z",                    // date-time
    "cvim-version": "1.2.0",                                      // string
    "timestamp-stop": "2017-10-03T13:03:19Z",                     // date-time
    "cpp-type": "vehicle",                                        // string
    "measurement-channel-id": "1",                                // string
    "geo-bounding-box": {
        "latitude-max": 48.804771423339844,                       // number
        "longitude-max": 3.060849905014038,                       // number
        "latitude-min": 48.804771423339844,                       // number
        "longitude-min": 3.060849905014038                        // number
    },
    "data": [                                                     // values in the package
        {
            "value": 102,                                         // number
            "timestamp": "2017-10-03T13:03:13Z"                   // date-time
        },
        {
            "value": 100,
            "timestamp": "2017-10-03T13:03:16Z"
        },
        { ... }
    ],
    "submit-time": "2019-04-30T15:17:47.941912Z",                 // date-time
    "type": "time-series",                                        // string
    "datapackage-id": "b72dfe16-b53d-4d69-ba3f-0976289cda4f",     // uuid
    "number-of-samples": 3,                                       // number
    "signatures": [                                               // security signatures array
        {
            "signatory": "MARKETPLACE",                           // string
            "checksum": "---",                                    // encoded string
            "signature": "-------------==",                       // encoded string
            "marketplace-validation": {
                "integrity": true,                                // boolean
```

```
                "authenticated": false                                    // boolean
            }
        }
    ],
    "marketplace-validation": {
        "integrity": true,                                                // boolean
        "authenticated": false                                            // boolean
    }
}
```

Listing 15. Get data package details

3.3.2.4. Get Data Request received Data Packages

This request can be used to retrieve a list of data packages received through a concrete data request.

REQUEST	GET DATA REQUEST DATA PACKAGES	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProviderOffer/{{serviceProviderOfferId}}/CvimDataPackages	
Headers	X-Auth-Token	{{access_token}}

Table 19. Get Data Packages received for a Data Request request

Additional parameters:

This request accepts additional parameters in the request url to act as filters of the amount of data packages to receive:

Parameter	Value	Description
Limit	Integer	Number of data packages to be received. If not provided, default value is 1000. Maximum value is 1000.
From	Integer	If provided, it defines the offset from the first result you want to fetch. If not provided, default value is 0
Page	Integer	If provided, it defines the first page you want to fetch. If not provided, default value is 0

Sort	String	If provided, it defines the field and order to sort results
Order	String	If provided, it defines order to sort results

Table 20. Query Parameters

Request example:

api.datagora.eu/api/ServiceProviderOffer/{{serviceProviderOfferId}}/CvimDataPackages?limit=100&page=1&order=asc

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
[ { DataPackage } ]
```

Response is an array of data packages. See 3.3.2.3 Get Data Package Details above

3.3.2.5. Get Data Request received Metadata Packages

This request can be used to retrieve a list of metadata packages received from a data request.

REQUEST	**GET DATA REQUEST METADATA PACKAGES**	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProviderOffer/{{serviceProviderOfferId}}/CvimMetadata	
Headers	X-Auth-Token	{{access_token}}

Table 21. Get Metadata packages received for a Data Request request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "total": 342,                                                   // integer
    "size": 342,                                                    // integer
    "metadata": [                                                   // metadata packages array
        {
            "datapackage-id": "---",                                // uuid
            "cpp-type": "vehicle",                                  // string
            "timestamp-start": "2017-10-03T13:03:13Z",              // date-time
            "submit-time": "2019-04-30T15:17:47.941912Z",           // date-time
            "number-of-samples": 3,                                 // integer
            "cvim-version": "1.2.0",                                // string
            "type": "time-series",                                  // string
            "timestamp-stop": "2017-10-03T13:03:19Z",
            "geo-bounding-box": {
                "latitude-max": 48.804771423339844,                 // number
                "longitude-min": 3.060849905014038,                 // number
                "latitude-min": 48.804771423339844,                 // number
                "longitude-max": 3.060849905014038                  // number
            },
            "measurement-channel-id": "1",                          // string
            "duration": 6,                                          // integer
            "marketplace-tags": {
                "location": [
                    "u0djt"
                ],
                "geoHashPrecision": 5                               // integer
            }
        },
        { ... }
    ]
}
```

Listing 16. Get Data Request Metadata

3.4. Toolbox API

3.4.1. Data Views

Tool that allows the allocating of certain data through a set of specific filters including signal values, and that allows to categorize the results.

3.4.1.1.Create Data View

REQUEST	CREATE NEW DATA VIEW	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/DataView	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	`{ "offer":"5d1da759f4cd991e00d0a308", // id "channels": ["602","610","604"], // array "filters": [], // object array "category": "5f92c00d9f2b082100062ccc", // id (optional) "name":"Weather control", // string "description": "Assess outside weather" // string }`	

Table 22. Create new Data View

Body explanation:

- **offer**: id of the data request from which data will be collected
- **channels**: array of measurement channels from which data received will be collected
- **filters** *: Array of objects that the data from the selected channels that will be collected.
- **category**: Id of the assigned category. Can be assigned later.
- **name**: String to identify the Data View.
- **description**: String defining the purpose of the Data View.

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

See 3.4.1.3 Get Data View details below

3.4.1.2. Get Data Views list

This request can be used to retrieve a list of your requested Time Series Analytics.

REQUEST	GET DATA VIEWS LIST	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProvider/{{serviceProviderId}}/DataView	
Headers	X-Auth-Token	{{access_token}}

Table 23. Get list of active Data Views

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of Data Views. See 3.4.1.3 Get Data View details below

3.4.1.3. Get Data View details

This request can be used to get a DataView details.

REQUEST	GET DATA VIEW DETAILS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/DataView/{{DataViewId}}	
Headers	X-Auth-Token	{{access_token}}

Table 24. Get Data View details

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "name": "Weather control",                                          // string
    "description": "Assess outside weather",                            // string
    "offer": {...},                                                     // offer object
    "channels": ["602","610","604"],                                    // array
    "filters": [],                                                      // array
    "outputAeonSubscriptionUrl": "https://{aeon_url}/subscribe/{uuid}", // string
    "category": {
        "name": "weather",                                              // string
        "enum": [...]                                                   // string array
    },
    "createdAt": "2020-10-23T11:37:27.568Z",                            // date-time
    "updatedAt": "2020-10-23T11:37:27.978Z",                            // date-time
}
```

Listing 17. Data View response

3.4.1.4. Retrieve Data View

This request returns the latest data collected in the Data View in form of rows. Result is an array of the latest rows collected, up to 100.

REQUEST	RETRIEVE DATA VIEW COLLECTED DATA	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/DataView/{{DataViewId}}/retrieve	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	{}	

Table 25. Retrieve data view collected data

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "status": "success",                                   // string
    "data": [                                              // rows array
        {
            "#": 1,                                        // integer (row number)
            "608": 11,                                     // channel value type
            "900": null,
            "charger_availability": "low"                  // enum string (category value)
            "tstamp": "2020-06-30T02:30:00+02:00",         // string
        },
        { ... }
    ],
}
```

Listing 18. Data View retrieve response

Please note that the category key-pair will only be sent once a category is assigned to the Data View.

3.4.1.5. Data View category

The next requests are used to create and manage category and the assignment of these to Data Views and their collected rows.

3.4.1.5.1. Create new category

This request is used to create a category. Categories are stored without being assigned to a Data View to be used as many times the Service Provider need.

REQUEST	CREATE CATEGORY
Method	POST

Url	https://api.datagora.eu/api	
Endpoint	/SEIcategory	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	{ "serviceProvider": "5cd421a79c0f591e006515a5", "name": "charger availability", "enum": ["low", "normal", "high",], }	

Table 26. Create category request

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

See 3.4.1.5.3 Get Category details below

3.4.1.5.2. Get categories list

This request can be used to retrieve the list of your created categories.

REQUEST	GET DATA VIEWS LIST	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProvider/{{serviceProviderId}}/SEIcategory	
Headers	X-Auth-Token	{{access_token}}

Table 27. Get list of created categories

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of categories. See 3.4.1.5.3 Get Category details below

3.4.1.5.3. Get Category details

This request can be used to get a Category details.

REQUEST	GET CATEGORY DETAILS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/SEIcategory/{{categoryId}}	
Headers	X-Auth-Token	{{access_token}}

Table 28. Get Category details

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

{

```
    "serviceProvider": {...},                                      // service provider object
    "id": "5f7f2fc48d3e0323004fcdc7",                              // string
    "name": "charger availability",                                // string
    "enum": [                                                      // string array
        "low",
        "normal",
        "high"
    ],
    "createdAt": "2020-10-08T15:27:00.953Z",                       // date-time
    "updatedAt": "2020-10-08T15:27:00.953Z",                       // date-time
}
```

Listing 19. Get categories list response

3.4.1.5.4. Assign Data View category

This request assigns a category to an uncategorized Data View. Once assigned the /retrieve response will include the category as an additional key-value in each row.

REQUEST	ASSIGN CATEGORY TO DATA VIEW	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/DataView/{{DataViewId}}/category	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	`{ "category_id": "5f7f2fc48d3e0323004fcdc7", // string }`	

Table 29. Assign category to an uncategorized Data View

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

3.4.1.5.5. Assign category values

This request returns the latest data collected in the Data View in form of rows.

REQUEST	ASSIGN CATEGORY VALUES TO DATA VIEW ROWS	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/DataView/{{DataViewId}}/assign	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	`{ "row": "value", // key value pair ... }`	

Table 30. Assign category values to Data View rows

Body explanation:

- **row**: number of the row (#). This value can be found at each row of the /retrieve response
- **value**: category value. This value must be a string in the category enum.

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

3.4.2. Analytics

Analytics are requests for analytical measurements of the data available for a set of filters, including specific data requests and measurement channels.

These analytics can be created by Service Providers with the data received from created data requests (see Create Data Request) with the `include-analytics` parameter set to `true`. To be sure which data requests are eligible for analytics perform a query for service providers data requests including the filter for said attribute:

```
https://api.datagora.eu/api/ServiceProvider/{{serviceProviderId}}/offers?include-analytics=true
```

Please, take into consideration that although all analytics has a similar model, some attributes may differ.

Currently available analytics types:

- Time Series
- Trajectory Analysis
- Networks
- Machine Learning

3.4.2.1. Time Series Analytics

Toolbox devoted to the analysis of time series, with a special focus on drift detection.

The identification of drifts, i.e. sudden changes in the dynamics, has numerous applications. For instance, a user may decide when to access a given stream of data, like only when significant changes have been detected

3.4.2.1.1. Create new Time Series Analytics

REQUEST	CREATE NEW TIME SERIES ANALYTICS	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/TimeSeries	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	{ "description": "", // string "model": "", // string "offer": "", // id "channels": "", // string "submissionDate": {}, // Object "travelDates": {}, // Object "travelledDuration": {}, // Object "geoBoundingBox": {}, // Object "variables": {}, // Object }	

Table 31. Create new Time Series analytics

Body explanation:

- **description**: descriptive name of the analytics.
- **model** **: type of Time Series analytics.
- **offer**: id of the data request from which data will be analyzed

- **channels**: array of measurement channels from which data received will be analyzed
- **submissionDate** *: Object representing the dates in which the data has been stored in the marketplace, including max and min values between which data will be searched.
- **travelDates** *: Object representing the dates in which the data has been generated by the signal, including max and min values between which data will be searched.
- **travelledDuration** *: Object representing the duration in seconds of each data package, including max and min values between which data will be searched.
- **geoBoundingBox** *: Object representing the geographical rectangular area within which the data is desired, including longitude and latitude max and min values.
- **variables** **: Object containing the different variables needed for each type of time series model. All variables of each model are mandatory.

** for every filter: if sent empty that filter will be count as NO FILTER (retrieve all)*

*** refer to subsections for special treatment depending on the Time Series model*

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

See 3.4.2.1.3 Get Time Series Analytics details below

3.4.2.1.1.1. SampleEntropy

Probabilistic measure of the complexity of a time series.

Given a specific embedding dimension e and tolerance t, the algorithm returns the probability (as a negative logarithm) that if two sets of e simultaneous data points have distance t, two sets of (e+1) simultaneous data points will also have distance t.

Variables

- **embeddingDimension**: (numeric - integer) number of points to consider in each set.
- **tolerance**: (numeric - integer) estimated distance for each set of simultaneous data.
- **normalized**: (boolean) whether or not to normalize the time series' values.

3.4.2.1.1.2. PermutationEntropy

Measure of the complexity of a time series.

Given the ordinal pattern of sequential sets of e points (each spaced by d points from the other) in a time series, it measures the variability of its behavior according to all possible changes in each set's ordinal pattern.

Variables

- **embeddingDimension**: (numeric - integer) number of points to consider in each set.
- **delay**: (numeric - integer) number of spacing points between sets

3.4.2.1.1.3. Irreversibility

Quantitative measure of the behavior of a time series when its sequence of timestamps is reversed. Computed over sets of data points.

Variables

- **embeddingDimension**: (numeric - integer) number of points to consider in each reversed set.

3.4.2.1.1.4. PearsonCorrelation

Correlation coefficient formulas are used to find how strong a relationship is between data. The formulas return a value between -1 and 1, where:

- 1 indicates a strong positive relationship.
- -1 indicates a strong negative relationship.
- A result of zero indicates no relationship at all.

Pearson correlation is the most used measure of correlation in science. It quantifies the strenthg of the association of two variables. To obtain meaningful results the data must fulfill the following requirements:

- Both variables should be normally distributed.
- Absence of outliers. These are values which lie an abnormal distance from other values.
- Variables should be continuous.
- Variables must have a linear relationship.
- Homocedasticity: equeal variance across cariables.

Mandatorily needs at least 2 trajectories to be provided in the data field.

Variables

None

3.4.2.1.1.5. SpearmanCorrelation

Correlation coefficient formulas are used to find how strong a relationship is between data. The formulas return a value between -1 and 1, where:

- 1 indicates a strong positive relationship.
- -1 indicates a strong negative relationship.
- A result of zero indicates no relationship at all.

Spearman's coefficient measures the rank correlation between two functions. To properly understand this correlation, it is necessary to know what a monotonic function is. A monotonic function is one that either never increases or never decreases as its independent variable increases.

Spearman's correlation coefficient is a statistical measure of the strength of a monotonic relationship between paired data.

Mandatorily needs at least 2 trajectories to be provided in the data field.

Variables

None

3.4.2.1.1.6. RegressionTree

A decision tree can be described as a flowchart where each internal node describes a test on a learning variable, each branch represents a result of the test and each sheet contains the value of the numeric variable.

Variables

- **forecast**: (numeric - integer) number of new data points to forecast.
- **division**: (numeric - integer) index of the time series point at which the division between training and test sets is to be performed.

3.4.2.1.1.7. NeuralNetwork

Multilayer Perceptron-based prediction of new data points for a given time series.

Variables

- **forecast**: (numeric - integer) number of new data points to forecast.
- **division**: (numeric - integer) index of the time series point at which the division between training and test sets is to be performed.

3.4.2.1.1.8. Arima

Time series forecasting method based on the following principles:

- Autoregression (AR): uses the dependent relationship between an observation and some number of lagged observations.
- Integrated (I): differencing of raw observations ($obsi - obsI{-}1$) to make the series stationary.
- Moving Average (MA): uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

Variables

- **forecast**: (numeric - integer) number of new data points to forecast.
- **alpha**: (numeric) value for the statistical significance (of the response variable) of the hypothesis test conducted within the model (0.05 is recommended as a widespread practice).
- **arimaOrder**:
  - **p**: (numeric – integer) related to AR's regular part. Number of lag observations included in the model.
  - **d**: (numeric – integer) related to I's regular part. Number of times that the raw observations are differenced.
  - **q**: (numeric – integer) related to Q's regular part. Size of the moving average window.
  - **seasonalP**: (numeric – integer) related to AR's stationary part. Number of lag observations included in the model.
  - **seasonalD**: (numeric – integer) related to I's stationary part. Number of times that the raw observations are differenced.
  - **seasonalQ**: (numeric – integer) related to Q's stationary part. Size of the moving average window.

3.4.2.1.2. Get Time Series Analytics list

This request can be used to retrieve a list of your requested Time Series Analytics.

REQUEST	GET TIME SERIES ANALYTICS LIST	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProvider/{{serviceProviderId}}/analytics/TimeSeries	
Headers	X-Auth-Token	{{access_token}}

Table 32. Get list of requested Time Series analytics

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of Time Series analytics. See 3.4.2.1.3 Get Time Series Analytics details below

3.4.2.1.3. Get Time Series Analytics details

This request can be used to retrieve a list of your requested Time Series Analytics.

REQUEST	GET TIME SERIES ANALYTICS DETAILS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/TimeSeries/{{analyticsId}}	
Headers	X-Auth-Token	{{access_token}}

Table 33. Get list of requested Time Series analytics

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "id": "---",                                                    // id
```

```
    "model": "---",                                                 // string
    "description": "my analytics",                                  // string
    "serviceProvider": {  },                                        // service provider object
    "offer": {  },                                                  // data request object
    "channels": [{  }],                                             // channel array
    "createdAt": "2019-05-20T09:20:54.222Z",                        // date-time
    "updatedAt": "2019-05-20T09:20:54.499Z",                        // date-time
    "inputAeonChannelId": "---",                                    // id
    "inputAeonSubscriptionUrl": "https://{aeon_url}/subscribe/{uuid}",// string
    "outputAeonChannelId": "---",                                   // id
    "outputAeonPublicationUrl": "https://{aeon_url}/publish/{uuid}",  // string
    "outputAeonSubscriptionUrl": "https://{aeon_url}/subscribe/{uuid}"// string
    "travelledDuration": {
        "min": -,                                                  // date *
        "max": -                                                   // date *
    },
    "submissionDate": {
        "min": "2019-06-03",                                       // date *
            "max": "2019-06-27"                                    // date *
    },
    "travelDates": {
        "min": -,                                                  // date *
        "max": -                                                   // date *
    },
    "geoBoundingBox": {
        "latitude-max": 58,                                        // number **
        "latitude-min": 38,                                        // number **
        "longitude-max": 25,                                       // number **
        "longitude-min": -5                                        // number **
    },
    "variables": {  },                                              // variables object
}
```

Listing 20 Time Series analytics response

** If this value is null or undefined the object will be sent but will not contain the key*
*** If any of the values is null or undefined, the object will be sent empty, as if there is no filter*

3.4.2.1.4. Delete Time Series Analytics details

This request can be used to stop the analysis of the data received for a Time Series Analytics.

REQUEST	GET TIME SERIES ANALYTICS DETAILS	
Method	DELETE	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/TimeSeries/{{analyticsId}}	
Headers	X-Auth-Token	{{access_token}}

Table 34. Delete Time Series analytics

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

3.4.2.2. Trajectory Analysis

Toolbox for the analysis of trajectories of cars, both individually and as a group.

Several options are provided, from simple statistics on the trips performed by cars; up to the detection of relationships (like correlations or causalities) between the corresponding movements.

Eligible measurement channel and data requests:

Only location channel (measurement-channel-id = 2) is eligible. Therefore, only data requests including this channel (and those including all channels) are eligible.

3.4.2.2.1. Create new Trajectory Analytics

<table>
<tr><th>REQUEST</th><th colspan="2">CREATE NEW TRAJECTORY ANALYSIS</th></tr>
<tr><td>Method</td><td colspan="2">POST</td></tr>
<tr><td>Url</td><td colspan="2">https://api.datagora.eu/api</td></tr>
<tr><td>Endpoint</td><td colspan="2">/analytics/TrajectoryAnalysis</td></tr>
<tr><td rowspan="2">Headers</td><td>X-Auth-Token</td><td>{{access_token}}</td></tr>
<tr><td>Content-type</td><td>application/json</td></tr>
<tr><td>Body</td><td colspan="2"><pre>{
    "description": "", // string
    "model": "", // string
    "offer": "", // id
    "submissionDate": {}, // Object
    "travelDates": {}, // Object
    "travelledDuration": {}, // Object
    "geoBoundingBoxConstraint": {}, // Object
    "variables": {}, // Object
}</pre></td></tr>
</table>

Table 35. Create new Trajectory analysis

Body explanation:

- **description**: descriptive name of the analytics.
- **model** **: type of trajectory analysis.
- **offer**: id of the data request from which data will be analyzed
- **submissionDate** *: Object representing the dates in which the data has been stored in the marketplace, including max and min values between which data will be searched.
- **travelDates**: Object representing the dates in which the data has been generated by the signal, including max and min values between which data will be searched.
- **travelledDuration** *: Object representing the duration in seconds of each data package, including max and min values between which data will be searched.
- **geoBoundingBoxConstraint**: Object representing the geographical rectangular area within which the data is desired, including longitude and latitude max and min values.
- **variables** **: Object containing the different variables needed for each type of trajectory model. All variables of each model are mandatory.

** if sent empty that filter will be count as NO FILTER (retrieve all)*

*** refer to subsections for special treatment depending on the Trajectory model*

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

See 3.4.2.2.3 Get Trajectory Analysis details below

3.4.2.2.1.1. Statistics

The statistics functionality provides a series of summarizing properties for a given trajectory, namely the total distance traveled in meters, its duration in seconds and its average velocity in meters per second.

Variables

None

3.4.2.2.1.2. Interpolation

The interpolation functionality provides a mean of resampling a given trajectory. The type of interpolation performed is linear.

By providing a desired time resolution (in seconds) the user is able to set the frequency of appearance of each point in the new resampled trajectory.

Variables

- **time-resolution** (numeric-integer): to which the trajectory is intended to be resampled as a numeric value. This value should fulfill the following conditions:
    - Be an integer.
    - Be greater than 0.
    - Be smaller than the time difference between the initial and last coordinates' timestamps.

3.4.2.2.1.3. Clustering

The clustering functionality enables the grouping of similar trajectories. Neighbouring trajectories are labeled as belonging to the same cluster based on a metric of pair-wise distance between coordinates.

Since the adjustment of the parameters intrinsic to the algorithm are adjusted automatically, the user only needs to provide the trajectories to be grouped, with no additional adjustable fields.

Variables

None

3.4.2.2.1.4. Anomaly Detection (anomaly_detection)

The anomaly detection functionality allows to identify abnormal points in a given trajectory. After associating the input trajectory to a cluster from the system's stored information, it compares the pair-wise distance between each of its coordinates and the group-centroid's. If this distance surpasses a certain threshold, the associated coordinates are marked as outlying.

The user has the option to change the type of threshold to apply in this process between a standard deviation protocol and an interquartile range one.

Variables

- **threshold-type** (string): threshold method to apply for the identification of outliers. If the user does not provide this parameter, then the standard deviation protocol is applied by default:
    - 'std' for standard deviation.
    - 'iqr' for interquartile range.

3.4.2.2.2. Get Trajectory Analysis list

This request can be used to retrieve a list of your requested Trajectory Analysis.

REQUEST	GET TRAJECTORY ANALYSIS LIST	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProvider/{{serviceProviderId}}/analytics/trajectoryAnalysis	
Headers	X-Auth-Token	{{access_token}}

Table 36. Get list of requested Time Series analytics

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of Trajectory analytics. See 3.4.2.2.3 Get Trajectory Analysis details below

3.4.2.2.3. Get Trajectory Analysis details

This request can be used to get the details of a Trajectory Analysis.

REQUEST	GET TRAJECTORY ANALYSIS DETAILS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/trajectoryAnalysis/{{analyticsId}}	
Headers	X-Auth-Token	{{access_token}}

Table 37. Get list of requested Time Series analytics

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "id": "---",                                                    // id
    "model": "---",                                                 // string
    "description": "my analytics",                                  // string
    "serviceProvider": {  },                                        // service provider object
    "offer": {  },                                                  // data request object
    "status": "ok" / "pending" / "error",                           // string
    "createdAt": "2019-05-20T09:20:54.222Z",                        // date-time
    "updatedAt": "2019-05-20T09:20:54.499Z",                        // date-time
    "travelledDuration": {
        "min": -,                                                  // date *
        "max": -                                                   // date *
    },
    "submissionDate": {
        "min": "2019-06-03",                                       // date *
        "max": "2019-06-27"                                        // date *
    },
    "travelDates": {
        "min": -,                                                  // date *
        "max": -                                                   // date *
    },
    "geoBoundingBoxConstraint": {
        "latitude-max": 58,                                        // number **
        "latitude-min": 38,                                        // number **
        "longitude-max": 25,                                       // number **
        "longitude-min": -5                                        // number **
    },
    "variables": {  },                                              // variables object
}
```

Listing 21. Trajectory Analysis response

** If this value is null or undefined the object will be sent but will not contain the key*
*** If any of the values is null or undefined, the object will be sent empty, as if there is no filter*

3.4.2.2.4. Get Trajectory Analysis results

This request can be used to get the results of a Trajectory Analysis.

REQUEST	GET TRAJECTORY ANALYSIS DETAILS

Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/trajectoryAnalysis/{{analyticsId}}/results	
Headers	X-Auth-Token	{{access_token}}

Table 38. Get results of requested Time Series analytics

Responses:

Code		**Description**
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of objects that differ depending on the model of the Trajectory Analysis

3.4.2.2.4.1. Statistics

```
[
    {
        "id": "---",                                            // id
        "datapackage-id": "---",                                // id
        "analysis": "---",                                      // id
        "distance": 10330.636916737218,                         // number
        "duration": 1017,                                       // number
        "velocity": 10.157951737204737,                         // number
    },
    ...
]
```

Listing 22. Trajectory statistics response

3.4.2.2.4.2. Interpolation

```
[
    {
        "id": "---",                                            // id
        "datapackage-id": "---",                                // id
        "analysis": "---",                                      // id
        "new_data": [
```

```
            {
                "id": "---",                                        // id
                "interpolation": "---",                             // id
                "value": [
                    51.499995,                                      // number
                    7.5129,                                         // number
                ],
                "timestamp": "---",                                 // date-time
                "createdAt": "---",                                 // date-time
                "updatedAt": "---",                                 // date-time
            },
        ]
    },
    ...
]
```

Listing 23 Trajectory interpolation response

3.4.2.2.4.3. Clustering

```
[
    {
        "_id": "---",                                               // string
        "total": "---",                                             // integer
        "packages": [
            "---",                                                  // datapackage-id
        ]
    },
    ...
]
```

Listing 24. Trajectory clustering results

3.4.2.2.4.4. Anomaly Detection

```
[
    {
    },
    ...
]
```

Listing 25. Trajectory anomaly detection results

3.4.2.3. Networks

Toolbox devoted to the analysis of relation between data collected from different sources for the same measurement channel of a Data Request.

3.4.2.3.1. Create new Network

REQUEST	**CREATE NEW NETWORK**
Method	POST

Url	https://api.datagora.eu/api	
Endpoint	/analytics/network	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	`{ "offer": "", // id "measurement-channel-id": "", // string }`	

Table 39. Create new Network

Body explanation:

- **offer**: id of the data request from which data will be analyzed
- **measurement-channel-id**: array of measurement channels from which data received will be analyzed

Take into account that before creating the network a check is conducted to know the availability of nodes inside it. Only networks with more that one node available will be generated.

Also a network for the combination of Data Request and measurement channel is unique.

3.4.2.3.2. Get Network list

This request can be used to retrieve a list of your generated networks.

REQUEST	GET NETWORK LIST	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProvider/{{serviceProviderId}}/analytics/network	
Headers	X-Auth-Token	{{access_token}}

Table 40. Get list of generated networks

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role

		Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of networks. See 3.4.2.3.3 Get Network below

3.4.2.3.3. Get Network

This request can be used to get a network details.

REQUEST	**GET NETWORK DETAILS**	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/network/{{analyticsId}}	
Headers	X-Auth-Token	{{access_token}}

Table 41. Get Network details

Responses:

Code		**Description**
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "id": "---",                                                   // id
    "offer": {  },                                                 // data request object
    "measurement-channel": {  },                                   // measurement channel object
    "measurement-channel-id": "1",                                 // string
    "inputAeonChannelId": "---",                                               // id
    "inputAeonSubscriptionUrl": "https://{aeon_url}/subscribe/{uuid}",     // string
    "createdAt": "2019-05-20T09:20:54.499Z",                       // date-time
    "updatedAt": "2019-05-20T09:20:54.499Z",                       // date-time

}
```

Listing 26. Network response

3.4.2.3.4. Get Network status

This request can be used to get the current status and information of an active network.

REQUEST	GET NETWORK STATUS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/network/{{analyticsId}}/info	
Headers	X-Auth-Token	{{access_token}}

Table 42. Get Network status

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
  "result": "success",                                              // string
  "response": "---",                                                // id
    "network-id": "---",                                            // id
    "network-efficiency": 0.02…,                                    // number
    "link-density": 0.25,                                           // number
    "adjacency-matrix": [                                           // object array
      {
        "source": 0,                                                // number
        "target": 1,                                                // number
        "value": 34.816…,                                           // number
      },
      ...

    ],
    "node-eccentricity": [                                          // object array
      {
        "node-id": 0,                                               // number
        "value": 34.816…,                                           // number
```

```
        },
        ...
      ]
    }
}
```
Listing 27. Network status response

3.4.2.4. Machine Learning

Toolbox devoted to the analysis of categorization of data rows obtained through Data Views to predict the values of the incomplete rows.

3.4.2.4.1. Create ML model

REQUEST	CREATE NEW ML MODEL	
Method	POST	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/MachineLearning	
Headers	X-Auth-Token	{{access_token}}
	Content-type	application/json
Body	{ "serviceProvider": "5cd421a79c0f591e006515a5", "method": "build", "algname": "sklearn.neural_network.MLPClassifier", "config": {}, "data": { "sei_uri_base": "https://vian-dev.fit.vutbr.cz/cross-cpp/", "view_id": {{dataViewId}}, "limit": -100, "page": 0 } }	

Table 43. Create new ML model

Body explanation:

- **serviceProvider**: id of the service provider
- **method**: method of creation. Can be "build" or "import"
- **algname** *: string identifying the algorithm to be used by the ML model.
- **config**: configuration object. Currently managed by the ML component.
- **data**: object identifying the data to be used to train the ML model.
    - **sei_uri_base**: location of the Data View. Currently not to be changed.
    - **view_id**: id of the Data View to use

- **limit and page**: selector of the rows from the Data View. Recommended to go be limit=-100 and page=0 to use the 100 most recent data rows collected.

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

See 3.4.2.4.3 Get ML model details below

3.4.2.4.2. Get ML model list

This request can be used to retrieve a list of your created Machine Learning models.

REQUEST	**GET ML MODELS LIST**	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/ServiceProvider/{{serviceProviderId}}/analytics/machineLearning	
Headers	X-Auth-Token	{{access_token}}

Table 44. Get list of created ML models

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

Response is an array of Trajectory analytics. See 3.4.2.4.3 Get ML model details below

3.4.2.4.3. Get ML model details

This request can be used to get the details of a Machine Learning model.

REQUEST	GET ML MODEL DETAILS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/MachineLearning/{{analyticsId}}	
Headers	X-Auth-Token	{{access_token}}

Table 45. Get details of a Machine Learning model

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "id": "5f7afd821288ac2100488533"                          // string
    "alguuid": "a1869a51-5992-4243-a3ca-1579450e462b",        // string
    "serviceProvider": {...},                                 // service provider object
    "method": "build",                                        // string
    "algname": "sklearn.neural_network.MLPClassifier",        // enum string
    "config": {},                                             // object
    "data": {                                                 // object
        "sei_uri_base": "https://vian-dev.fit.vutbr.cz/cross-cpp/",
        "view_id": "5ef5ad9dce7ee04300f06b00",
        "limit": -100,
        "page": 0,
    },
    "type": "batch",                                          // string
    "status": "check",                                        // string
    "createdAt": "2020-10-05T11:03:30.648Z",                  // date-time
    "updatedAt": "2020-10-05T11:03:30.648Z",                  // date-time
```

```
}
```

Listing 28. Machine Learning model details

3.4.2.4.4. Export ML model

This request can be used to export the ML model as it is stored in the ML component.

REQUEST	EXPORT ML MODEL	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/MachineLearning/{{analyticsId}}/export	
Headers	X-Auth-Token	{{access_token}}

Table 46. Export Machine Learning model

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "id": "5f7afd821288ac2100488533"                      // string
    "alguuid": "a1869a51-5992-4243-a3ca-1579450e462b",    // string
    "serviceProvider": {...},                             // service provider object
    "method": "build",                                    // string
    "algname": "sklearn.neural_network.MLPClassifier",    // enum string
    "config": {},                                         // object
    "data": {                                             // object
        "sei_uri_base": "https://vian-dev.fit.vutbr.cz/cross-cpp/",
        "view_id": "5ef5ad9dce7ee04300f06b00",
        "limit": -100,
        "page": 0,
    },
    "type": "batch",                                      // string
    "status": "check",                                    // string
    "createdAt": "2020-10-05T11:03:30.648Z",              // date-time
```

```
    "updatedAt": "2020-10-05T11:03:30.648Z",                    // date-time
    "model": "…",                                               // string with the exported model
}
```

Listing 29. Export Machine Learning model

3.4.2.4.5. Get ML model status

This request can be used to get the current status of the ML model training.

REQUEST	GET ML MODEL STATUS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/MachineLearning/{{analyticsId}}/status	
Headers	X-Auth-Token	{{access_token}}

Table 47. Export Machine Learning model

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "status": "success"                                         // string
    "data": {                                                   // object
        "status": ""                                            // string
    }
}
```

Listing 30. Get Machine Learning model status

Status can be one of these possible values:

- **failed**: the training has failed. Model should be deleted
- **initialized**: the model has been instantiated but is not yet running.
- **running**: the model is currently under training
- **finished**: the model training has finished successfully and can be used

- **imported**: the model has been imported successfully and can be used

3.4.2.4.6. Evaluate ML model

This request can be used to get an evaluation of a ML model training.

REQUEST	GET ML MODEL STATUS	
Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/MachineLearning/{{analyticsId}}/evaluate	
Headers	X-Auth-Token	{{access_token}}

Table 48. Export Machine Learning model

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "status": "success"                                     // string
    "data": {                                               // object
        "score": 1                                          // double
    }
}
```

Listing 31. Evaluate Machine Learning model

3.4.2.4.7. Apply ML model

This request can be used to apply the current training of the ML model to the whole Data View.

It will assign a category value on every row based on the training with the manually assigned values.

REQUEST	APPLY ML MODEL

Method	GET	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/MachineLearning/{{analyticsId}}/apply	
Headers	X-Auth-Token	{{access_token}}

Table 49. Export Machine Learning model

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

Request ok response body:

```
{
    "status": "success"                                   // string
    "data": [                                             // rows array
        {
            "#": 1,                                       // integer (row number)
            "608": 11,                                    // channel value type
            "900": null,
            "charger_availability": "low"                 // enum string (category value)
            "tstamp": "2020-06-30T02:30:00+02:00",        // string
        },
        { ... }
    ],
}
```

Listing 32. Apply Machine Learning model

3.4.2.4.8. Kill ML model

This request can be used to stop a ML model.

REQUEST	KILL ML MODEL
Method	GET
Url	https://api.datagora.eu/api

Endpoint	/analytics/MachineLearning/{{analyticsId}}/kill	
Headers	X-Auth-Token	{{access_token}}

Table 50. Kill Machine Learning model

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

3.4.2.4.9. Delete ML model

This request can be used to stop and delete a ML model.

REQUEST	DELETE ML MODEL	
Method	DELETE	
Url	https://api.datagora.eu/api	
Endpoint	/analytics/MachineLearning/{{analyticsId}}	
Headers	X-Auth-Token	{{access_token}}

Table 51. Delete Machine Learning model

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

4. AEON

4.1. What is AEON

AEON is a cloud platform to create applications with real time communications channels. The architecture is based on the strongly communication needs that we need to face nowadays, with billions of interconnected devices and short times of response. Thus, the technological solutions used for the implementation are based on strong requirements about performance, response and scalability, making use of the most advanced cutting-edge technologies.

AEON platform offers a shared cloud-based message queuing framework enabling messaging between various entities that wish to communicate with each other seamlessly and reliably using standard vendor neutral protocols

Benefits:

- Communicate applications and services through a real time network
- Easy to use, easy to integrate in developments: AEON provides an SDK to connect your services and devices over a globally scaled real-time network
- Performance, Scalability and Reliability: High performance for message delivery and data exchange between business processes and devices and from device to device. AEON is able to handle multiple types and priorities of messages, whilst at the same time providing the necessary Quality of Service. AEON provides reliable messaging with durability and persistence and needs to scale well for extremely large volumes.
- Big Data: AEON can take care of the cloud messaging of the data capture from M2M environments and big data flows.

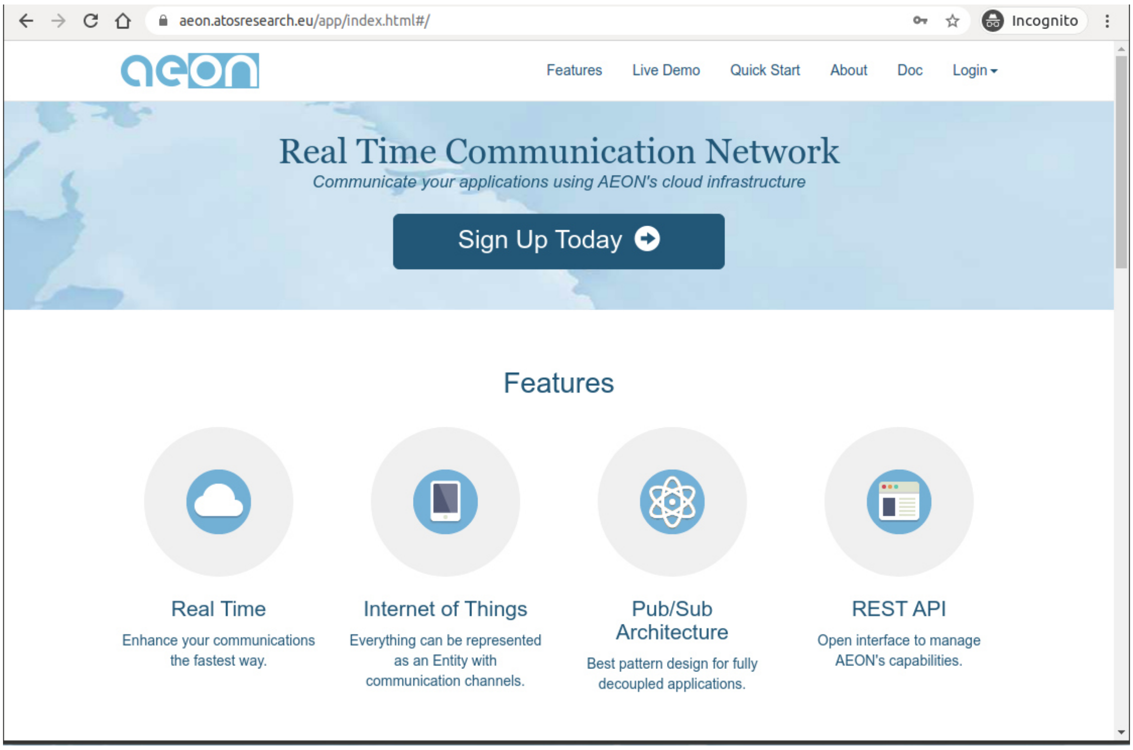


Figure 31. AEON dashboard.

4.2. Configuring AEON channels

AEON users can create AEON channels to publish/subscribe messages through their subscription and publications URLs. The Cross-CPP marketplace uses AEON with its own user to create communication channels for Data Requests (offers), active streaming analytics, Data Views, etc. to get data notifications in real-time.

Data consumers or Service providers will subscribe to the subscriptions URLs to receive real-time data from the Data Requests, analytics, Data Views etc. The configuration is provided in the detailed information view of each one in the Cross-CPP Marketplace frontend and API.

4.3. Subscribing to AEON

AEON provides an SDK (Node.js, JavaScript and Java) that encapsulates the complexity of connecting to a socket server.

You can find an example in section 1.4.2 Push Approach Data Retrieval and visit https://aeon.atosresearch.eu:3000/public/doc/html/quickstart/nodejs.html for further documentation.

5. Context Monitoring and Extraction (CME) guide

The CME module provides one customisation endpoint for Service Providers:

- Customisation of existing or creation of new Reasoning Rules

The customisation explained here and any further customisation of the CME components, e.g. for adding reasoning rules for extraction, can be made by downloading and changing the CME module as provided in the open source code project on GitHub[4] under the EPL 2.0 license or by placing a customisation request to the CME team at context-support@cross-cpp.eu.

A part of the main adaption work to be done when customising the Context monitoring & extraction module is the introduction of new rules and changing the existing ones. For this purpose, the CME framework provides interfaces both within the code as well as in the form of a freely adaptable configuration file. Both will be described within this section.

5.1. Extraction rules configuration file

The configuration file can be found within the main folder of the CME module and is named *extraction_configuration.xml.* Within this file, the following sections can be found which can be adapted by the CME administrator:

- setup of the reasoning rules for the context-sensitive data discovery
- setup of the reasoning rules for the context-sensitive data access control for the data owner (security feature)
- configuration of the reasoning rules

Setup of the reasoning rules for context-sensitive data discovery

The following Figure 32 shows the reasoning rules that can be applied by the context-sensitive filtering of data packages during the data discovery as they are listed on the UI of the Marketplace within the *Additional configuration* tab.

[4] Link to be added

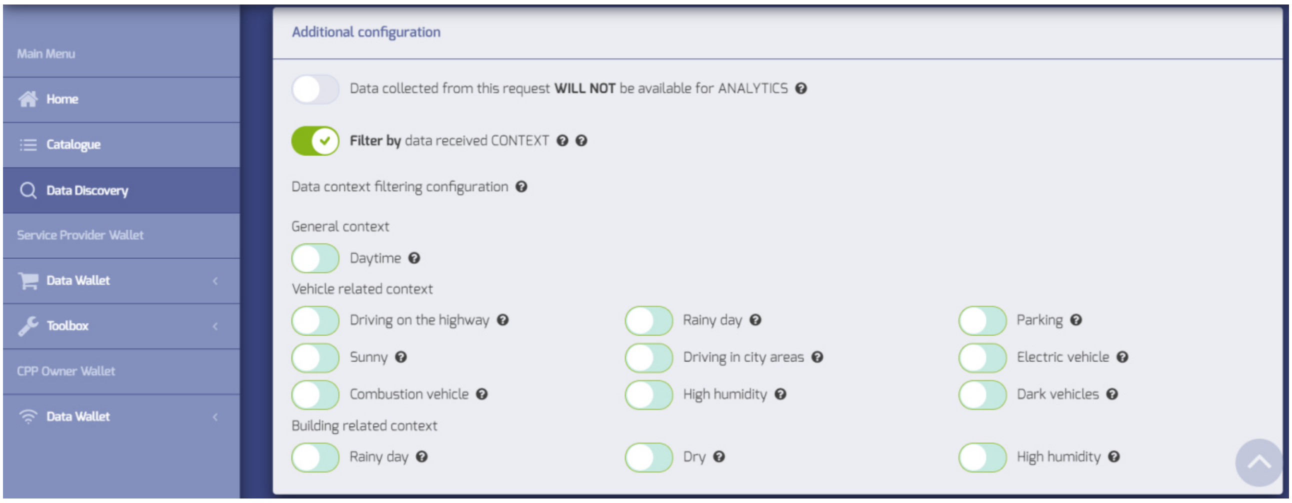


Figure 32. Context-sensitive data filtering in the Cross-CPP Marketplace UI

In the configuration file, the administrator will find the following structure for building this list of filtering options:

```
<rules>
   <rule id={unique_id} name={rule_name} cppType={cpp_type}
        tooltip={tooltip} />
</rules>
```

Each rule for the context-sensitive data filtering has the following attributes:

- *unique_id:* a unique identification number
- *rule_name:* a name indicating the meaning or scope of the rule
- *cpp_type:* the cpp type to which this rule should be applied in the extraction service (by default *vehicle* and/or *building*)
- *tooltip:* a tooltip which gives more information about the semantic of the rule, which will be shown also in the Marketplace UI

A rule can simply be registered by adding a new *<rule>* element to the *<rules>* section.

Hint: Pay special attention to use user-friendly names, descriptions and tooltips within your rule configuration in order to make them easier accessible for the end users. Make sure to define only rules for which exists a valid configuration within a *<ruleConfiguration>* section.

Setup of the reasoning rules for the context-sensitive data access control for data owners

Find in Figure 33 a snippet of the context-sensitive data access control options for the data owner in the Marketplace UI, which allows him to control his data access according to specific context parameters.

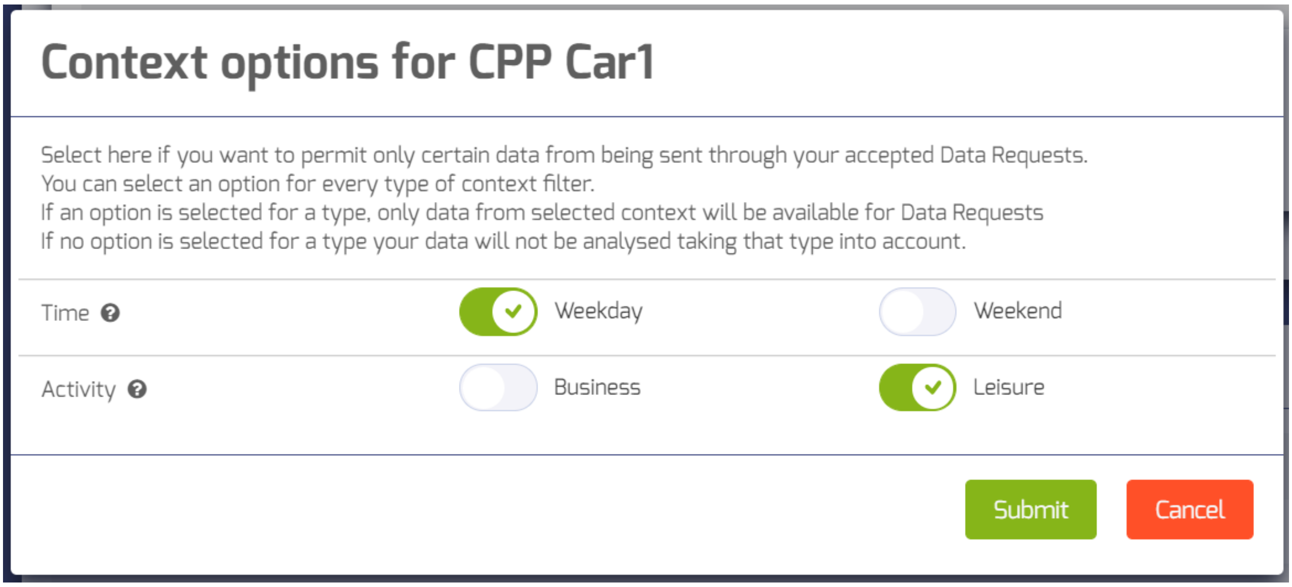


Figure 33. context-sensitive data access control options in the Cross-CPP Marketplace UI

In the configuration file, these options are being defined within the *<dataOwnerContextOptions>* element. Each Option consists of a set of different alternatives for this option and has the following attributes:

- *unique_name:* a unique name for this option
- *tooltip:* a description for this option to be shown as tooltip in the Marketplace UI

Enclosed to an option are different alternatives, specified within the *<alternative>* element as shown below.

```
<dataOwnerContextOptions>
   <option name={unique_name} tooltip={tooltip}>
      <alternative>{alternative1}</alternative>
      <alternative>{alternative2}</alternative>
   </option>
</dataOwnerContextOptions>
```

Hint: in order to function as desired, the choice of options and alternatives has to be aligned with the set of context variables used by the Cross-CPP security module's access control policy. Also make sure to introduce only options and alternatives which correspond to a valid rule configuration (see later in this section).

Configuration of the reasoning rules

Each rule specified within *<rules>* can be configured within a corresponding *<ruleConfiguration>* element. Find the template for the configuration below

```
<ruleConfiguration id={unique_id} internalName={rule_name} cppType={cpp_type}>
   <signalConfiguration id={measurement_channel_id}>
      <value>{value}</value>
```

```
   </signalConfiguration>
   <signalConfiguration id={measurement_channel_id}>
      <max>{maximum_value}</max>
      <min>{minimum_value}</min>
   </signalConfiguration>
</ruleConfiguration>
```

The attributes for a rule configuration are:

- *unique_id:* a unique identifier of the rule corresponding to those specified within *<rules>* element
- *rule_name:* the (module internal) name of the rule
- *cpp_type:* the CPP type this rule applies to (by default *vehicle* or *building*)

Each rule configuration is being accompanied by a set of signal configurations, indicating the measurement channels used within the rule. For each measurement channel involved, a *<signalConfiguration>* element will be introduced. The signal configuration specifies the parameter value operators, to which the rule should be applied to. Examples are *maximum (signal < maximum), minimum (signal > minimum)* and *value (signal = value)*. Each of the value operators can be defined within a corresponding *<max>*, *<min>* or *<value>* element, as can be seen in the template above.

5.2. Source code customisation

In order to make additional defined reasoning rules function correctly or to edit the already existing rules, the administrator has to adapt the existing CME source code at some specific spots, which will be shown and explained in the following section. Changes will have to be made within the following java classes[5]:

- *ExtractionRule:* In order to make the CME recognise the rule this interface has to be implemented
- *ExtractionOptions:* recently created reasoning rules have to be registered here

For each reasoning rule defined in *<rules>* a corresponding class implementing the *ExtractionRule* has to be implemented. The abstract class *ExtractionRule* provides the following methods to be extended:

- *void readConfiguration(string url):* method to parse the *extraction_configuration.xml* and provide information about the attributes and parameters
- *boolean applyRule():* defines the core reasoning logic, which evaluates to *true* if the rule applies to the given set of data

[5] Link to github repository to be added here

The following class attributes are available and have to be set correctly:

- *id:* the unique identifier of the rule
- *keyword:* a name/description of the rule, which can match with the *internalName* specified within the configuration file
- *CPPType:* the CPP type to which the rule should be applied

A template for reading the extraction configuration is available in the already existing rule implementations and can be used for creating new rules.

In order to make CME recognise and use the newly defined rules they first have to registered in *ExtractionOptions*, as seen below. Once the rules are registered here the extraction service will use them at the next module start up.

```
registerRule(new RuleIsDrivingOnHighway(CONFIG_PATH));
registerRule(new RuleIsWeekday(CONFIG_PATH));
registerRule(new RuleIsWeekend(CONFIG_PATH));
registerRule(new RuleIsBusiness(CONFIG_PATH));
registerRule(new RuleIsLeisure(CONFIG_PATH));


... register here the additional rules
```

Further customisation is possible by developers as the CME module is provided as an open source code project on github[6] under the EPL 2.0 license. Furthermore, customisation can also be request via [context-support@cross-cpp.eu.](mailto:context-support@cross-cpp.eu)

[6] Link to be added

FAQ

Cross-CPP data-marketplace

Q: What is Cross-CPP data-marketplace?

A: Cross-CPP data-marketplace connects Data Providers and Data Consumers for selling and acquiring Connected Vehicle and Home Building data under the Common Industrial Data model (CIDM). It offers a secure and privacy preserving experience when selling or buying sharing big data, by having the full control over your data shared, to whom and for what purposes.

Cross-CPP offers to cross-sectorial Data Consumers, the possibility to search for more than 200 sensor signals, display advance visualization representations (such as Histograms, Geo-Histograms, Time Series) and retrieve those datasets in a seamless experience thanks to the open SDK-API created.

Q: How do I, as data consumer, register into CROSS-CPP data-marketplace?

A: You can find the registration form by clicking the "Sign on!" button in the landing page. Select "Service Provider" role and fill the fields to request your registration. Once your registration is validated by a system administrator an email will be sent to you to confirm your access, Then, access the link in your email, login and accept the consent to start using the Cross-CPP Marketplace.

Q: What do I have to do in order to start working with CROSS-CPP data-marketplace?

A: Once registered you must be familiar with the CIDM, as it is the format in which you will receive the data you request. You must also be familiar with AEON, as it is the communication channel used to send the data. You can find information for both in this guide.

Cross-CPP data model

Q: What is the Common Industrial Data Model (CIDM)?

A: The CIDM is a standardized data model for industrial data-driven services.

Q: Which are the benefits and advantages of using the CIDM model for data -driven services:

A: The CIDM constitute a major business and technical advantage for Data Consumers:

- The CIDM provides a brand-independent and transparent data model, which harmonizes proprietary data into generic datasets independently of any cross-sectorial Industry
- It is built on an open and highly scalable automotive big data format (JSON Schema).
- Active community of service providers increasing the number of signals available from vehicles and Smart Buildings to be recorded as well as the type of measurement channels can be modified or extended
- The Data Provider also provides an origin certification as a CIDM feature to support the validation and verification of origin, integrity and completeness of data. The intention is to protect the data inside the Data Package against manipulation.

Q: What is a signal?

A: A signal is the information provider of each CPP. They are the perception organs of CPPs and it is their main duty to detect physical phenomenon and chemical quantities. They observe the environment and generate data in the CIDM format. An example could be "speed" or "latitude"

Q: What is a measurement channel?
A: A measurement channel is the way the physical signals and their sampled measurements are implemented and represented in the CIDM format. Some examples could be "Vehicle Speed" using the signal "Speed" in a time-series or in a histogram format, or "Position" using both "Latitude" and "Longitude" signals.

Q: Can I request a new signal or channel?
A: Cross-CPP data-marketplace offers a wide variety of signals provided by the manufacturers. The catalogue is really extensive and can be filtered in many ways. If even then you can't find the signal that you need and/or think can be provided by any of our data providers, please contact us in: cross-cpp-support@lists.atosresearch.eu.

Cross-CPP marketplace components

Q: What is the Data Discovery component and how does it work?
A: The Data Discovery component is a tool that allows you to find what data you can access through the marketplace. There you can use the filters provided to narrow or enlarge your results and create Data Request based on the configured search. You can think of it like a test of what would you receive if you publish that request.

Q: What is the Context Monitoring and Extraction module and how can help my Organization?
A: The Context Monitoring and Extraction module allows Cross-CPP to suggest you signals to add to your current Data Discovery filters, based on the context data of the signals already selected. This might help you find data of interest that you would miss otherwise.

Data Requests

Q: What is a Data Request?
A: A Data Request is a set of filters that defines which type of data would you like to receive. You would receive data from data owners that have accepted these requests through each request unique AEON channel.

Q: How do I create a Data Request?
A: You have to get to the Data Discovery and define the data you are interested in through the filters given. Once set you give it a descriptive name, so data owners guess in a glance the nature of the request.

Q: How do I receive data from a Data Request?
A: In the very moment a Data Request is created an AEON channel is assigned. You can find the channel configuration in each data request details view. Data from data owners that have accepted the request will be sent through each AEON channel assigned to each eligible data

request, meaning you can receive data from the same user and signal from more than one request.

Q: Can I modify my Data Request one created?
A: No. The acceptance of a request by a data owner implies a consent from its side. Modifying the request would make invalid such consent. Therefore, you can create another Data Request with the new desired configuration.

Q: Can I use the data collected for other purposes not described in my Data Request?
A: No. The acceptance of a request by a data owner implies a consent from its side. That user allows certain usage of the data given and only for the purposes described in the request.

Toolbox

Q: What is the toolbox?
A: The Toolbox is a set of tools that offers Service Providers a way to generate analytics from the data obtained from Data Requests or further filter this data in order to get exact measurements or use the service as a notification system.

Q: How can I request analytics?
A: Any analytics uses data packages received from a Data Request, meaning that first the data Request must have been accepted by CPP Owners and started receiving data. Then, on the Cross-CPP Marketplace a Service Provider can create analytics based on those Data Requests.

Q: How do I get my analytics results?
A: Depending on the analytics type results can be instantly shown on the screen, as a diagram, chart, map and so on, or a new AEON channel is provided in order to subscribe to upcoming analytics results. Every analytics type explains the way to get the results on screen.

Q: How can I consult my analytics?
A: Any created analytics can be consulted on the Cross-CPP Marketplace under Analytics section. There a Service Provider can get the AEON channel for subscription, see results of one-time analytics, or even delete them.

Q: Which Time Series forecasting method should I use for my time series predictions?

A: Prior to attempting to predict future values of a time series, you should verify the assumptions your selected algorithm does on the input data. For example, ARIMA will assume that your data is autoregressive and Regression trees will not be able to detect trends on data (thus not being advisable for a time series in which trend is a significant feature). If you are unsure about the underlying patterns of your data, Neural Networks may work best for you, as it does not rely on any a-priori assumption.

Q: How does the embedding dimension parameter "m" affect the entropy metrics?

A: The time-series are defined by sequences of points. The embedding dimension refers to the number of points use to evaluate. So then, smaller embedding dimensions would yield more detailed information while larger numbers tend to be more general. Bear in mind if the embedding dimension is too low maybe the results may be chaotic.

Q: How do I choose between a Pearson and a Spearman's correlation coefficient?

A: This decision will depend on the nature of your data, as these coefficients measure different types of association between variables (Pearson quantifies linear relationship, while Spearman does so with non-linear patterns). Their use is not exclusive, so even both could possibly be valuable to your analysis.

Q: What are the units of the trajectories module?

A: For distances, the meter, and for times the second. Then the velocity is expressed in m/s.

Q: What is the interpretation of the results of clustering the trajectories?

A: Each trajectory included in the data is assigned to a numerical ID that refers to the cluster that is being assigned. By instance, all trajectories of similar length from east to west are grouped in the cluster number 1, so the results are 1 for each trajectory.

Q: What do the nodes represent in the networks module? And the links?

A: Each node is an object. It could represent a house, building, a car. The links are an abstract representation of some magnitude that is being measured. Links could be based on distances by instance.

Q: What is a Data View?
A: A Data View is a configuration to get data filtered by specific values constraints for one or more Measurement Channels included in a Data Request.

Q: How can I create a Data View?
A: Go to Data Views section under Toolbox in the Cross-CPP Marketplace. There a step by step guide will be offered.

Q: How do I get the data view results?
A: Service Providers can consult their generated Data Views through Data Views under Toolbox section. There the configuration can be consulted, as well as the AEON channel to subscribe to. Also, the options of retrieving the latest data or even deleting the Data View are offered there.

Q: How can I use a created ML model?
A: A machine learning model that was built in a previous step can be applied on new (unseen, unannotated) Data View rows by invoking the Apply function. The service estimates the category on the given data rows and outputs it as the response.

Q: My service processes a lot of data, is the ML component ready for big data application?
A: Yes, even a large neural network model loads in less than 480 ms and it can apply the category prediction with the speed of more than 5000 rows per second on the Cross-CPP testing infrastructure.

Q: What ML methods should I try first?

A: Although the ML components support all new and fancy neural network methods available in Scikit-learn, Google TensorFlow, and Facebook PyTorch libraries, we suggest to start with the simple linear method of Stochastic Gradient Descend or the basic Multi-Layer Perceptron neural network for your initial experiments. Often, the quality of results provided by these models is satisfactory and performance gains can be brought by additional annotations of the data.

AEON

Q: What is AEON?

A: AEON is a cloud platform to create applications with real time communications channels

Q: How can I use AEON to subscribe to my data requests and data analytics?

A: AEON provides an SDK (Node.js, JavaScript and Java) that encapsulates the complexity of connecting to a socket server. Please refer to section 6.4, document examples and online page for extended documentation.

Q: How do I create or configure an AEON channel?

A: You don't have to create or configure any channel. All needed AEON channels, such as for data requests or analytics, are created and assigned by Cross-CPP. You only have to use the channels given. The channels configuration can be found in the details view of each data request or analytics.

Glossary

Administrator: Cross-CPP marketplace system administrator

Autoregressive data: In a time series domain, it refers to data which values depend on prior data points from the same time series.

AEON: AEON application

AEON application: publication/subscription based communication application

AEON channel: set configuration for communication between two actors through AEON application

Analytics Toolbox: set of available analytics functions to be requested by the Service Provider

CIDM: Common Industrial Data Model

CIDM model: standardized data model for industrial data-driven services

CME: Context Monitoring and Extraction

Company Backend: system of an OEM that provides its users data to the Cross-CPP marketplace

Contract: entity that resumes the acceptance of a data request from a CPP owner

CPP: cyber-physical product

CPP Data: data created by a CCP and sent to the system by the Company Backend

CPP owner: CPP owner which CPP is registered in the Cross-CPP data-marketplace

Cross-CPP: System

CSS: Context Sensitive Security

Data Request: set of configurations that define a scope for CPP Data to be received by a SP

Data View: set of configurated filters to receive specific values from a Data Request through a different notification channel

Entropy: It is usually explained as the order of a system. It is more accurate to understand the entropy as the lost information of a system. This definition for data classification problems implies the algorithms search for the variables that reduces the lost information of the system, those are the best classifiers.

Homoscedasticity: property of a multivariate domain in which the variance of each variable's error term is equal.

Machine Learning Analytics: A type of Analytics included in the toolbox

Marketplace: Marketplace Web Application

Measurement Channel: sampler of the data the signals process

Monotonic relationship: A type of association between variables that occurs when two variables tend to increase or decrease in the same direction, but not following a linear pattern (linear relationship).

MP: Marketplace

Multilayer Perceptron: A type of Artificial Neural Network with a varying number of hidden (processing) layers.

Networks Analytics: A type of Analytics included in the toolbox

Network Diameter: Value indicating the shortest distance between the two most distant nodes in a network.

Network Efficiency: Measure of how well information is exchanged between the nodes of a networks.

Node Eccentricity: Value representing the centrality of a network's node, or how close it is to the other nodes in the network.

OEM: Original Equipment Manufacturer

Rank Correlation: A type of correlation measure that quantifies ordinal association between two variables.

Service Provider: actor who receives the data created by owners to use it on the creation or improvement of services

Service Provider Wallet: group of MP functionalities for Service Providers

Signal: information provider of the data the CPP sensors generate

Stationarity: Property of a time series indicating that its statistical properties (e.g. mean, variance…) do not change over time.

System: the whole lot of applications that conforms Cross-CPP, including Marketplace Web Application and Marketplace Server.

Time Series Analytics: A type of Analytics included in the toolbox that analyses drifts in the data flow of time-series type channels.

Time Series Complexity: Measure of the presence of nonlinear patterns that explain the behaviour of a time series' data.

Trajectories Analysis: A type of Analytics included in the toolbox that uses trajectory related signals.

UUID: universally unique identifier. Standardized 16 bytes Id signature formed by 32 hexadecimal digits (example: `90eb04b2-a07c-4835-8618-9c0140f8391a`)

Figures

Tables

Lists

About Cross-CPP

The objective is to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research is to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project will develop: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to pro-vide enhanced local weather forecast and predict and optimise energy consumptions in households. The project will build upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, will be used as a basis for further development aiming to extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

CROSS-CPP

Funded by the Horizon 2020 Framework Programme of the European Union

Every effort has been made to ensure that all statements and information contained herein are accurate, however the Cross-CPP Project Partners accept no liability for any error or omission in the same.

3 Annex 2: Developer Guide Data Provider

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

CROSS-CPP

Cross-CPP MARKETPLACE DEVELOPER GUIDE

CPP DATA PROVIDER GUIDE

(CROSS-SECTORIAL ORIGINAL EQUIPMENT MANUFACTURER)

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 780167

Introduction

Data Provider Developer Guide offers data providers the information needed to connect a Company Backend system to the Cross-CPP Cloud Storage to share CPP data in Data Packages using Common Industrial Data Model (CIDM)

Purpose

This guide aims to help developers from an Original Equipment Manufacturer (OEM from now on) company about how to develop a Company Backend system capable of sharing data with the Cross-CPP solution.

Audience

This guide is meant for and solely for developers of OEM companies that wants to share data from CPP Owners through the Cross-CPP solution.

Scope

The content of this guide is meant to be taken into consideration only when developing a Company Backend system (CB from now on) looking to work with Cross-CPP Marketplace and will only cover functionalities meant to be used by those developers.

Cross-CPP team does not take responsibility on bad use of the application or the data provided when not following the instructions given in this guide.

Troubleshooting

For any questions or inquiries about the use of the Cross-CPP Cloud Storage API or SDK, or the contents of it or this guide, or if you find there is no content in this guide for some functionality please forward it to: marketplace-support@cross-cpp.eu.

Contact

Cross-CPP Project website: https://cross-cpp.eu

Cross-CPP Marketplace: https://datagora.eu

Marketplace support: marketplace-support@cross-cpp.eu

Context Monitoring and Extraction Module (CME): [context-support@cross-cpp.eu.](mailto:context-support@cross-cpp.eu)

Contents

Guide

Cross-CPP Marketplace Company Backend (CB) developers guide offers five distinct sections for Data Providers Company Backend (CB):

- Sending Data Process: complete workflow of data sharing
- Common Industrial Data Model (CIDM): Cross-CPP data model specification
- Cloud Storage API: REST API through which the data is sent to be stored in the Cloud Storage (CS) and made available for the Cross-CPP Marketplace.
- SDK
- Context Monitoring and Extraction (CME)

1. Sending Data Process

This process describes the required steps a CB needs to perform in order to send data to the CS.

1. Register Company Backend into the Cloud Storage
    a. CB must be CIDM compliant
    b. CB receives API key to operate with the CS
2. CPP Owners registration
    a. CPP Owners must register on their own
    b. Registered CPP owners must give write access
3. Data Storage

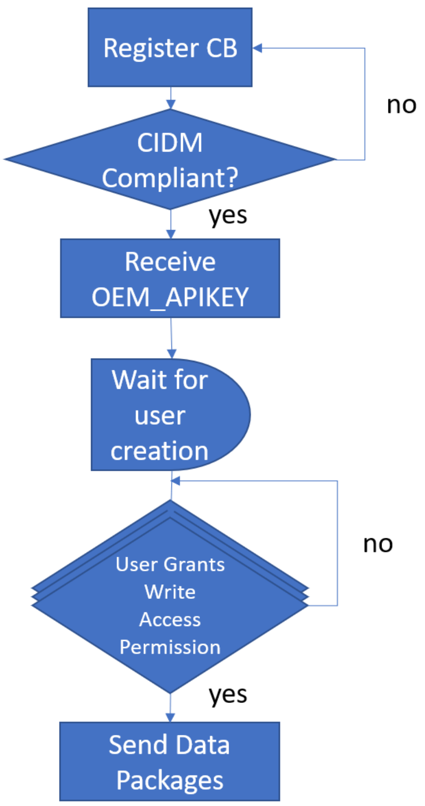


Figure 1. Sending Data process

1.1. Register the Company Backend into the Cloud Storage

The first step for the OEM is to register the Company Backend (CB) into the Cloud Storage (CS).

For this, the OEM have to provide the company name and a URL, that also will be used to receive notifications,

The administrator of the Cloud Storage will register a CIDM compliant Company Backend[1] by means of the Admin panel of the Cloud Storage. After the registration, the CB administrator will receive the **APIKEY** to configurate CB-CS communication.

1.2. CPP Owners registration

CPP Owners (vehicle or building owners) have the right to grant access to write their own data coming from the Company Backend into the Cloud Storage.

CPP Owners can grant write permission to the CB through the CS frontend: The screen capture shows the CPP Owner interface to grant write access permission. Once the permission is granted the name of the OEM Company is green. CB would receive the **vault-id** to start sending data to the CS. The vault id is the identification of the secure owner data space related to a vehicle or building.

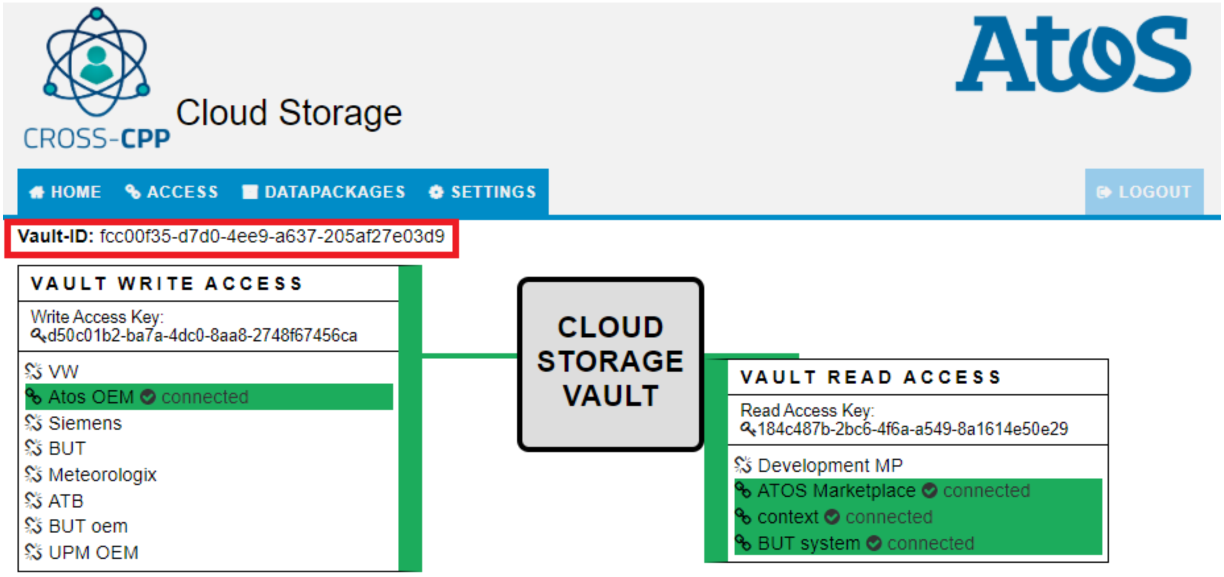


Figure 2. Cloud Storage Permissions View.

[1] A CIDM compliant Company Backend, is a Company Backend that implements the APIs provided by the Cloud Storage (CIDM compliant) and can therefore store CPP stream data in the Cloud Storage in the CIDM format.

1.3. Data Storage

Once the data owner grants the CB write permission, the CB can send data to the CS. The CB uses the Data Storage API endpoint to send data in the CIDM format. The request must provide the OEM_APIKEY in order to authenticate the CB and to check the authorization of the vault-id of the data-package.

This is the main request for Data Providers as is the entry point of data collected from their CPPs.

Note that the request body is always an array, and every package is treated separately, meaning the array can contain any amount of data from different signals and data types.

REQUEST	PUSH DATA PACKAGE	
Method	POST	
Url	https://cloudstorage-api.datagora.eu/api	
Endpoint	/datapackages	
Headers	Authentication	{{oem_apikey}}
	Content-type	application/json

Body

```
[
    {
        "cvim-version": "1.2.0"                                  // string
        "vault-id": "---",                                       // uuid
        "measurement-channel-id": "151",                         // string
        "type": "histogram",                                    // enumeration string
        "timestamp-start": "2019-05-20T17:01:32.402330Z", // date-time
        "timestamp-stop": "2019-05-20T17:18:00.402330Z", // date-time
        "data": [ * ],                                           // array of values
        "cpp-type": "vehicle",                                  // enumeration string
        "trip-id": "---",                                        // uuid
        "sumbmit-time": "2019-05-20T17:19:00.402330Z ",  // date-time
        "expiration-date": "2018-12-31T23:59:00Z",       // date-time
        "mileage-start": xx,                                     // number
        "mileage-stop": xx,                                      // number
        "room-id": "---",                                        // string
        "geo-bounding-box": {
            "latitude-max": 51.517008,                           // number
            "longitude-min": 7.4256,                             // number
            "longitude-max": 7.48278,                            // number
            "latitude-min": 51.501453                            // number
        },
        "location": {
            "latitud": 51.517008,                                // number
            "longitude": 7.4256                                  // number
        },
        "data-masiking-active": false                            // boolean
        "signatures": [     // array of objects about security signatures
            {
                "signatory": "OEM",                              // string
                "signature": "Rr---------4Zw==",                 // encrypted string
                "checksum": "4f3-------d71"                      // string
            }
```

	``], "data-ownership-information": { "data-privacy-level": "public", // enumerated string ["public", "shared", "private"] "data-stakeholders": [{ "status": "Creator", // string "name": "CPP owner name" // string }, { ... }], "copyright-stakeholders": [{ "status": "Creator", // string "name": "CPP owner name" // string }, { ... }], "privacy-veto-rights": { "consent-level": "public", // enumerated string ["public", "shared", "private"] "data-format": "time-series", // enumerated string ["time-series", "histogram"] "jurisdiction": "Europe", // enumerated string ["Europe", "any"] "storage-constraint": "OEM storage" // enumerated string ["OEM storage", "Personal storage"] } } }] ```

Table 1. Push Data Package request

Body explanation (for each data package):

- **vault-id**: identificatory of the cloud storage vault belonging to the data owner
- **measurement-channel-id:** channel to sample the data received
- **type**: type of data. Must be the same as the type of channel (time-series, histogram, geo-histogram, event-based, general-purpose, basic-cpp-information)
- **data**: array of data. The entities expected depend on the type of data
- **geo-bounding-box / location**: squared boundaries of the location, or exact location, in which the data was generated
- **signatures**: encrypted security signatures from the OEM
- **data-ownership-information**: optional object including the information relative of the nature of the data, including privacy levels and rights, involved stakeholders and copyright entities.

The body of the request is an array of CIDM data-packages. Even though the CIDM specification requires the "submit-time" and "datapackage-id" you don't need to provide them as the CS will provide those values automatically when the data-package is saved. The CS will validate that the data-packages are CIDM compliant and response OK.

Every data-package must provide the "vault-id" in order to assign the data-package to the proper storage space of the user. The "type" field specifies the type of data package, the main types are "time-series", "basic-cpp-information", "event-based", etc (see CIDM section 2 for further details). The "measurement-channel-id" specifies the signals that the data-package collects (you can check the list of measurement-channels available on the Marketplace Catalogue or ask the administrator to create a new one that fits to your needs). The format of the "data" object will depend on the "type" and the "measurement-channel-id" of the data-package.

1.3.1. Time-series

```
{
    "type": "time-series",
    "number-of-samples": 3                          // integer (required)
    "data": [                                       // object array (required)
        {
            "timestamp": "2019-05-20T17:08:29.607343Z",   // date-time
            "value": [                              // values array
                *                                   // value per channel signal
            ]
        },
        { … },
    ],
    "statistic-properties": {                       // object (optional)
        "min": 0,                                   // number
        "max": 3,                                   // number
        "average": 2,                               // number
        "histogram": {                              // object
            "measurement-channel-id": "2"           // string
            "data": [                               // array
                2,                                  // number
                ...
            ]
        }
    }
}
```

Listing 1. Time-series type data package

```
* "data" length must be equal to "number-of-samples" value
```

1.3.2. Histogram

```
{
    "type": "histogram",
    "data": [                       // number or number arrays array (required)
        1,                          // number
        …
    ]
}
```

Listing 2. Histogram type data package

1.3.3. Geo-histogram

```
{
    "type": "geo-histogram",
    "data": [                                               // object array (required)
        {
            "tileY": 21790,                                 // number
            "tileX": 34123,                                 // number
            "histogram": [                                  // number array
                44,                                         // number
                ...
            ]
        },
        { ... },
    ]
}
```

Listing 3. Geo-histogram type data package

1.3.4. Event based

```
{
    "type": "event-based",
    "data": [                                               // object array (required)
        "event-type": "real-time-event"                     // enumeration string *
        "event-data": {
            "event-time-stamp": "2019-05-20T17:08:29.607343Z", // date-time
            "event-datapackages": [ DataPackage model ]     // DataPackage object array
            "value": ""                                     // string
        }
    ]
}
```

Listing 4. Event-based type data package

```
* "real-time-event", "trigger-event", "threshold-event"
```

1.3.5. General purpose

```
{
    "type": "general-purpose",
    "data": {                                               // object (required)
        "key": "value"                                      // configured key: value
    }
}
```

Listing 5. General purpose type data package

1.3.6. Basic CPP information

```
{
    "type": "basic-cpp-information",
    "data": {                                               // object (required)
        "VehicleColor": "red"                               // configured key: value
    }
}
```

Listing 6. Basic CPP information type data package

Responses:

Code		Description
200	OK	Request successful
400	Bad Request	Query was malformed or incorrect
401	Unauthorized	Missing authorization token Unauthorized role Unauthorized user
404	Not found	Something requested does not exist
500	Internal Server Error	Something else went wrong

2. Common Industrial Data Model (CIDM)

Cross-CPP uses the CIDM as its data model. All data pushed into Cross-CPP Marketplace must follow this model.

2.1. Model Architecture

The CIDM architecture consist of three layers:

- The Signal layer consisting of the information provided by the CPP devices like vehicles or smart buildings. Signals are generated by sensors that observe the environment and produce data, as they detect physical and chemical phenomenon, for example, speed, temperature, charge state level, etc.
- The Measurement Channel layer providing signals data aggregation. The data needs to be pre-processed since raw sensor data exceeds the available storage and transferring capacity, to reduce the size of data down-sampling and histograms methods are provided.
- The data layer aggregating data inside data packages to store and transfer. One data package contains data from exactly one signal measured with one Measurement Channel. In addition to the actual data, Data Packages contain header information ("meta data"). This header information provides ownership of the data and gives quality of signal indications by OEM signatures or describes parameters of the measurement (e.g. time, rough position estimate, etc.).

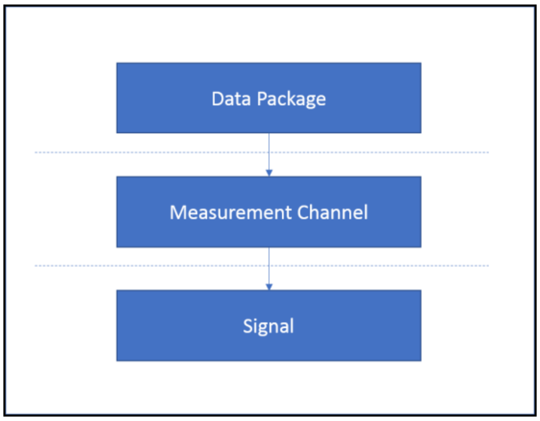


Figure 3: Layered High-level View of the Common Industrial Data Model (CIDM)

2.2. Signal Layer Specification

Sensors are the perception organs of CPP devices like vehicles and buildings. It is their main duty to detect physical phenomenon and chemical quantities by transferring them into electrical signals. The signal layers describe different types of signals and formats represented in the system. A new property is needed to group signals regarding the signal source type, cpp-type. Figure 4 shows the UML modelling of the signals for CIDM.

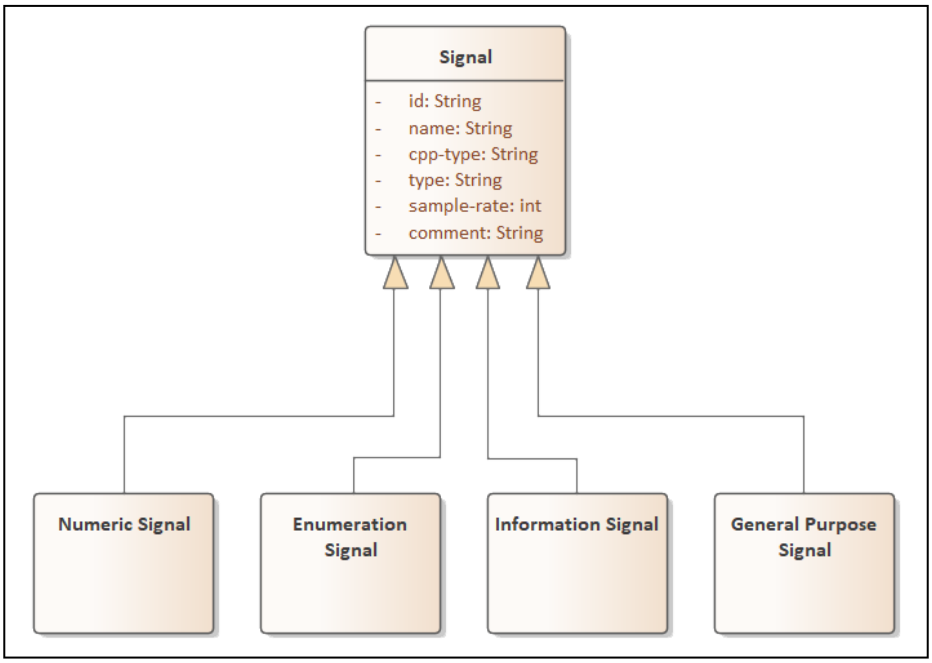


Figure 4. Signal UML Model.

The cpp-type is a required property that must be one of the two values, “vehicle” or “building”. The table below shows the complete definition of the Signal.

Property	Occurrence	Type	Format	Description
Common Properties				
id	Required	String		Unique Identifier of the Signal
name	Required	String		Name of the Signal
cpp-type	Required	String	one of: - vehicle - building	Type of the CPP
type	Required	String	one of: - numeric - enumeration - information - general-purpose	Type of the Signal
format	Optional	String		Signals representation format
sample-rate	Required	Numeric	double	Sample rate in Hz (Samples per Second). Must be larger than or equal to zero.
comment	Optional	String		Description of the signal
Numeric Signal				

Property	Occurrence	Type	Format	Description
type	Required	String	"numeric"	Type of the Signal needs to be numeric
format	Required	String	*<numeric formats>*	Signal's numeric representation (e.g. uint8, double, etc.)
min	Required	Number	<according to format>	Minimum Signal value
max	Required	Number	<according to format>	Maximal Signal value
resolution	Required	Number	<according to format>	Signals resolution
Unit	Required	String		Unit of the Signal (e.g. ºC)
Enumeration Signal				
type	Required	String	"enumeration"	Signal's type attribute needs to be "enumeration"
items	Required	Array	String	String array with possible Signal values
Information Signal				
type	Required	String	"information"	Signal's type attribute needs to be "information"
format	Required	String		Signals representation format (e.g. VIN, etc.)
General Purpose Signal				
type	Required	String	"general-purpose"	Signal's type attribute needs to be "general-purpose"
*	Optional	Any	No	May be extended with further attributes

Table 2. Signal property definition.

2.3. Measurement Channel Layer Specification

The measurement layer defines how sensor signals are captured and processed. One Measurement Channel describes how samples from one (or more - in the case of multidimensional histograms) sensor signal are aggregated and measured. Figure 5 shows the Measurement Channel UML model.

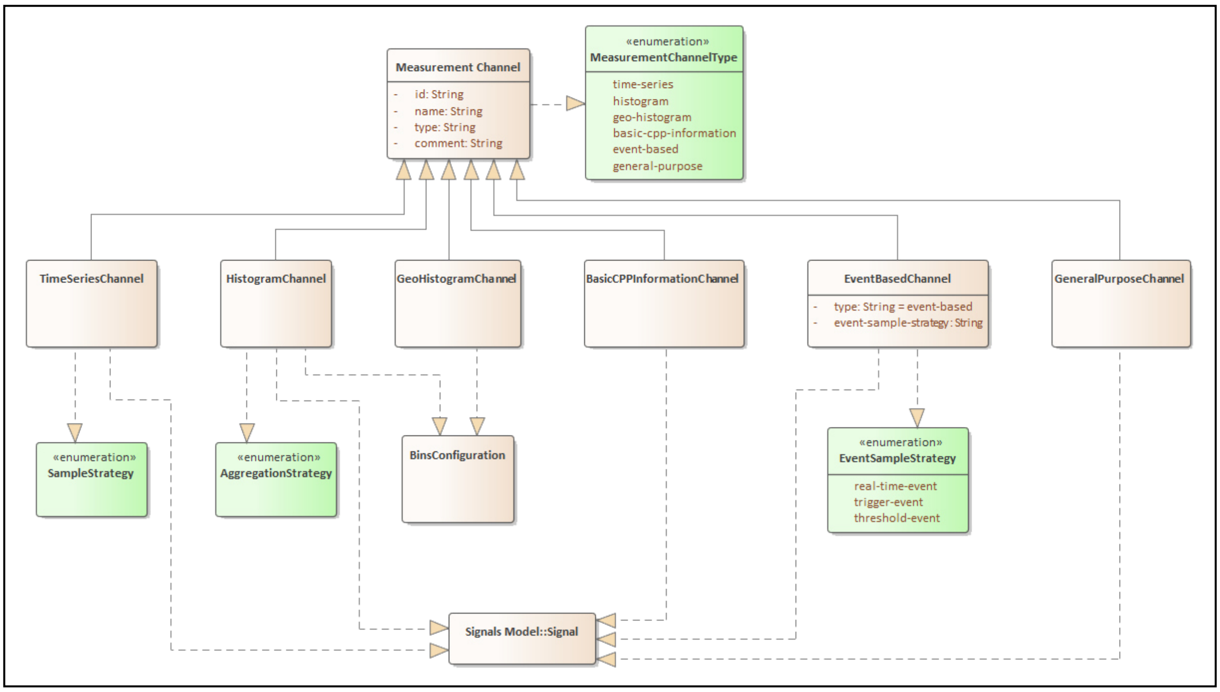


Figure 5. Measurement Channel UML Model.

The basic CPP information channel provides static information that is not measured by sensors but provides information about the CPP device, like the colour of a vehicle, the number of floors of a building, the identification number of a car, etc.

The event-based measurement channel provides information of events that occurs when the value of a measurement gets to a specific value (real-time-event) or when the value passes a specific threshold (threshold) and the last one that provides information about the event and the data-packages that have triggered the event. The table below details the measurement channel specification.

Property	Occurrence	Type	Format	Description
Common Properties				
id	Required	String	No	Unique Identifier of the Measurement Channel
name	Required	String	No	Name of the Measurement Channel
type	Required	String	one of: - time-series - histogram - geo-histogram - general-purpose - event-based - basic-cpp-information	Type of Measurement Channel
Comment	Optional	String	No	Description of the signal

Property	Occurrence	Type	Format	Description
Time Series Measurement Channel				
type	Required	String	"time-series"	Type of the Measurement Channel needs to be time-series
format	Required	String		Data type format of the samples
dimension	Optional	Number	uint32	Dimension of the Time-series. If dimension is not given, one-dimensional is assumed
capture-interval	Required when on-change is false	Number	double	Capture interval between two samples in seconds. Only required, when on-change is false.
on-change	Required	Boolean		Does Measurement-Channel only record changes in signal
sample-strategy	Required	String	one of: - min - max - average - last-known-value	Signal sampling strategy
signal	Required	Object	Array of Signal Object	See section 6.1.2 for Signal object definition
Histogram AND geo-Histogram Measurement Channel				
type	Required	String	one of: - histogram - geo-histogram	Type of the Measurement Channel needs to be histogram or geo-histogram
aggregation-strategy	Required	String	one of: - time - count - min - max	Histogram values aggregation strategy
capture-interval	Required	Number	double	Capture Interval of one Histogram. Needs to be larger than zero. +Infinity is valid (see IEEE 754).
dimensions	Required	Number	uint32	Dimensions of the Histogram
bins	Required	Array	Bin-Configuration Object	Array of bin configurations. Array

Property	Occurrence	Type	Format	Description
				needs to contain exactly one configuration for every dimension/axes of the histogram
Geo-Histogram Measurement Channel				
type	Required	String	"geo-histogram"	Type of the Measurement Channel needs to be geo-histogram
geo-resolution	Required	Numeric	double	Zoom level of the geo-histogram
Basic CPP Information Measurement Channel				
type	Required	String	"basic-cpp-information"	Type of the Measurement Channel needs to be basic-cpp-information
signal	Required	Object	Signal Object	See section for Signal object definition
Event Based Measurement Channel				
type	Required	String	"event-based"	Type of the Measurement Channel needs to be event-based
format	Required	String		Data type format of the samples
event-sample-strategy	Required	Event Sample	one of: - real-time-event - trigger-event - threshold-event	Event sampling strategy
comment	Optional	String	No	Description of Event Strategy
General Purpose Measurement Channel				
type	Required	String	"general-purpose"	Type of the Measurement Channel needs to be general-purpose
signal	Required	Any		See section 6.1.2 for Signal definition

Table 3: Measurement channel definition

2.4. Data Package Layer Specification

Data Packages contain the actual data of Signal measurements. As Signals are the information providers and Measurement Channels define the process of data acquisition from those Signals, Data Packages provide a structure for storing the data. In addition, they provide meta / header information containing time of recording, data ownership information, etc. Data Packages contain data from exactly one Measurement Channel. This leads to six different types of Data Packages that are defined similar as the Measurement Channels:

- Time Series Data Package
- Histogram Data Package
- Geo-Histogram Data Package
- Event based data Package
- Basics CPP information Data Package
- General Purpose Data Package

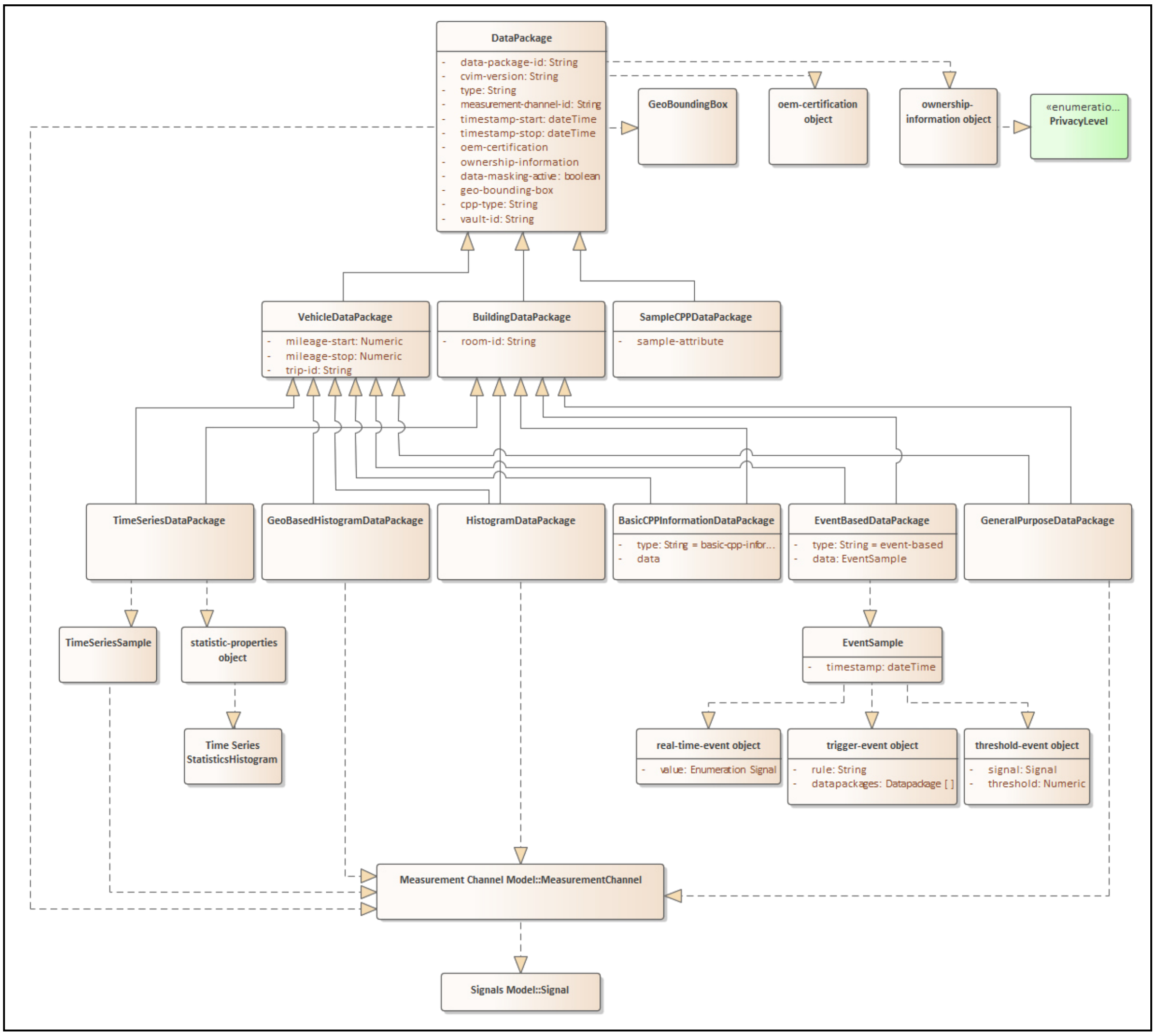


Figure 6. Data Package UML Model.

The data-package definition includes cpp-type to indicate is the data-package belongs to a "building" or a "vehicle" cpp and definition of data-package type, such as basics-cpp-information data-package and the event-based data-package.

The data of basics-cpp-information data-package depends of the signal type definition of the measurement channel - numeric, enumeration, information or general-purpose.

Event-based data-package has an additional property, "event-sample-strategy", to indicate three different type of event, real-time, trigger and threshold. According to the type of event, the "data" object has two mandatory properties, "timestamp" and "value", and an optional property named "datapackages" that is an array of the data-packages that triggers the "trigger-event".

The "building" CPP devices provides a set of sensors distributed along the rooms of the building, in order to identify the devices of the same room a new property has been included, "room-id".

Property	Occurrence	Type	Format	Description
Common Properties				
data-package-id	Required	String	UUID	Identifier of the Data Package. Unique per Cloud Storage Vault, Set by Cloud Storage Provider
cvim-version	Required	String	version	The name of the property is for backward compatibility with CVIM[2]. Must be set to 1.2.1
type	Required	String	one of: - time-series - histogram - geo-histogram - general-purpose - event-based - basic-cpp-information	Type of the Data Package
vault-id	Required	String	UUID	ID of the Cloud Storage Vault, where the data is stored in.
cpp-id	Optional	String	any	ID of the CPP
cpp-type	Required	String	one of: - vehicle - building	Type of the CPP
trip-id	Optional	String	any	Trip-ID of the User

[2] Common Vehicle Information Model: the basis for the current CIDM

Property	Occurrence	Type	Format	Description
room-id	Optional	String	any	ID of the room in a building where the measurement data was collected
measurement-channel-id	Required	String		Identifier of the Measurement Channel whose data is inside this data package
mileage-start	Optional	Number	double	Mileage at the start of the measurement in kilometres (km)
mileage-stop	Optional	Number	double	Mileage at end of measurement (km)
geo-bounding-box	Optional	Object	Geo-Bounding-Object	Geographic bounding box
location	Optional	Object	Location-Object	Single location including latitude and longitude
oem-certification	Optional	Object	OEM-Certification-Object	OEM Certification
data-ownership-information	Optional	Object	Ownership-Information -Object	Data Ownership Information
expiration-date	Optional	String	date-time	Data expiration date
data-masking-active	Optional	Boolean		Indicates status of data-masking (true = active)
Time Series Data Package				
type	Required	String	"time-series"	Type of the Measurement Channel needs to be time-series
timestamp-start	Required	String	date-time	Measurement start time
timestamp-stop	Required	String	date-time	Measurement stop time
number-of-samples	Required	Number	uint32	Number of samples that are stored in data
statistic-properties	Optional	Object	statistic-properties-object	Provides statistic properties about the data
data	Required	Array	time-series key-value-pair -object	Array of time-series-data Objects. The size of the array mist equal number of samples
Histogram Data Package				

Property	Occurrence	Type	Format	Description
type	Required	String	histogram	Type of the Measurement Channel needs to be histogram
timestamp-start	Required	String	date-time	Measurement start time
timestamp-stop	Required	String	date-time	Measurement stop time
data	Required	(Multi-dimensional) Array	Number	Array containing he bin counts. Size of array must match the dimension and bin configuration of the related Measurement Channel. Number format depends on Histogram aggregation-strategy
Geo-Histogram Data Package				
type	Required	String	geo-histogram	Type of the Measurement Channel needs to be geo-histogram
timestamp-start	Required	String	date-time	Measurement start time
timestamp-stop	Required	String	date-time	Measurement stop time
data	Required	(Multi-dimensional) Array	Number	Array containing the bin counts. Size of array must match the dimension and bin configuration of the related Measurement Channel. Number format depends on Histogram aggregation-strategy. The outer most dimension is the geo-dimension. It must match in its size the size of the geo-tiles array.
geo-tiles	Required	Array	Geo-Tile Object	Array of geo-tile objects. Only visited tiles are included.
Basic CPP Information Data Package				
type	Required	String	"basic-cpp-information"	Type of the Measurement Channel needs to be basic-cpp-information

Property	Occurrence	Type	Format	Description
Timestamp	Required	String	date-time	Measurement date time
data	Required	Any		Data depends on the type of signal of the measurement channel
Event Based Data Package				
type	Required	String	"event-based"	Type of the Measurement Channel needs to be event-based
Timestamp	Required	String	date-time	Measurement date time
event-sample-strategy	Required		one of: - real-time-event - trigger-event - threshold-event	Event sampling strategy
data	Required	Object	Event Sample Object	event-based data Object indicating an event
General Purpose Data Package				
type	Required	String	"general-purpose"	Type of the Measurement Channel needs to be general-purpose
Timestamp	Required	String	date-time	Measurement date time
data	Required	Any	time	Datatype depends on Measurement Channel

Table 4. Data Package definition.

Property	Occurrence	Type	Format	Description
Timestamp	Required	String	date-time	Timestamp of the event
value	Required	String		e.g. "*Ignition On", "Wipers Off"*
datapackages	Optional	Array of data-package		

Table 5. Event Sample Object

2.5. Measurement Channel Catalogue

The complete list of available Measurement Channels in the Cross-CPP Marketplace can be found at https://ng8.datagora.eu/pages/management/channels

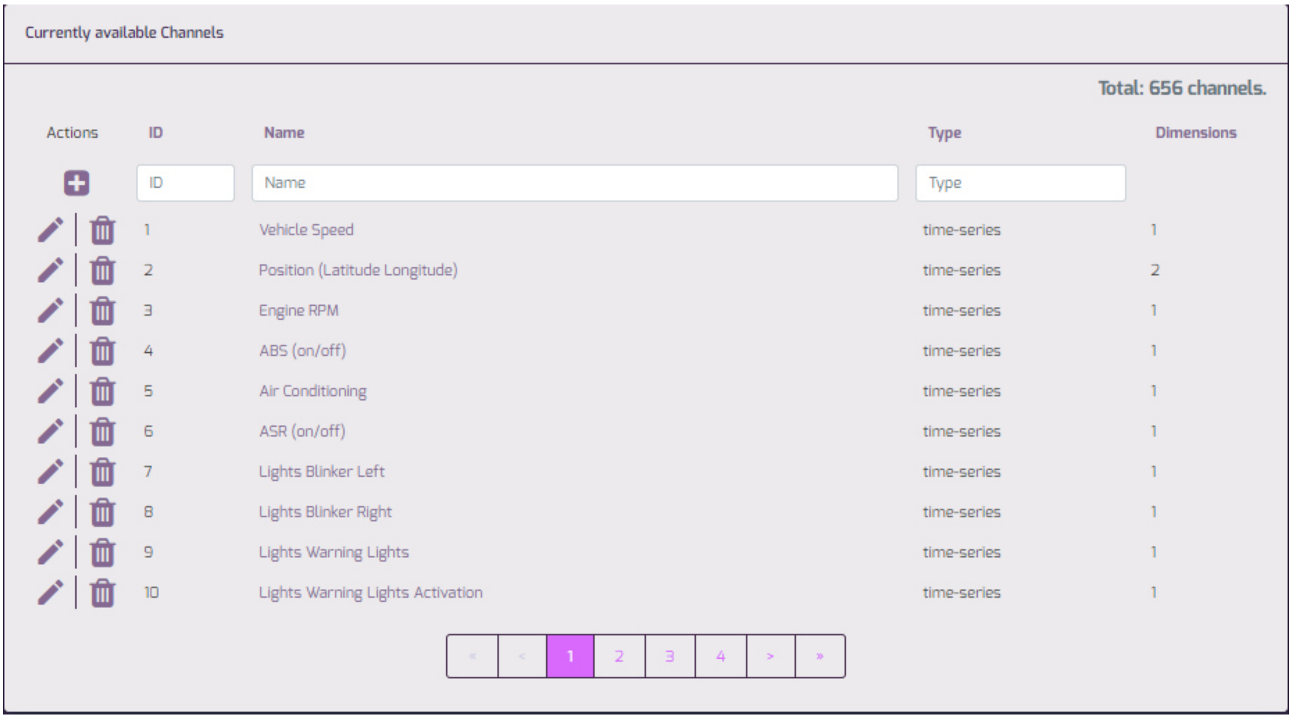


Figure 7. Measurement Channel Catalogue

3. Cloud Storage API Specification

Data sent by Data Providers is stored in the Cloud Storage (CS) to make it available for the Cross-CPP Marketplace functionalities such as the data discovery. When Data is received in the CS, the MP will be notified so that it can be collected and sent to those Data Consumers subscribed for that kind of data.

Data received in the Cloud Storage **must follow the Common Industrial Data Model (CIDM).**

The API url (from now on *api_url*) is: **https://cloudstorage-api.datagora.eu/**

The specification of the CS API is provided in the OpenApi specification that allows to describe and visualize RESTful web services. An online reference of version 3 of the API can be found in: https://cloudstorage-swagger.datagora.eu/docs/ under sections:

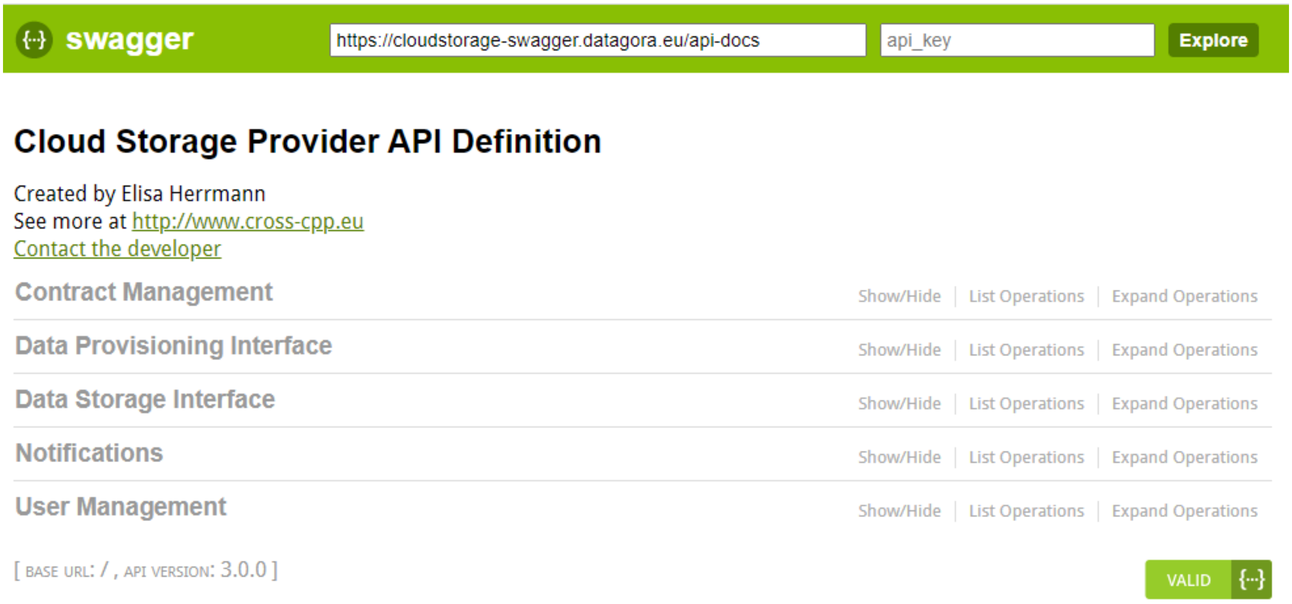


Figure 8. Cloud Storage API Specification.

See the complete OpenAPI specification in the Annex OpenAPI Specification of the Company Backend REST API (yaml).

4. SDK

In order to facilitate the integration of the Company Backends with the Cloud Storage there are several SDK provided. The list of the languages supported can be increased by means of tools that automatize the creation from the OpenAPI specification.

- typescript-node-client-generated.zip
- typescript-angularjs-client-generated.zip
- java-client-generated.zip
- python-client-generated.zip
- javascript-client-generated.zip
- go-client-generated.zip
- csharp-client-generated.zip

5. Context Monitoring and Extraction (CME)

The CME module provides 2 main customisation endpoints:

- Context models (section 5.1)
- Customisation of existing or creation of new Reasoning Rules (section 5.2)

The customisation explained here and any further customisation of the CME components, either for adding further monitors or reasoning rules for extraction, can be made by downloading and changing the CME module as provided in the open source code project on GitHub[3] under the EPL 2.0 license or by placing a customisation request to the CME team at context-support@cross-cpp.eu.

5.1. Context Models

The Context Models in Cross-CPP are not a software component to be implemented, but models that describes the static and dynamic aspects of use of CPPs.

The Context Models are modelled in the form of ontologies and for this, the context modelling software component is used to create and update the needed models.

As context models describe the situations under which signal value is measured, or a CPP is used, they may vary from CPP to CPP. This means that in the setup phase of the Cross CPP ecosystem, the context model(s) for the specific CPP have to be defined and implemented. The Cross-CPP context model bundle includes one Generic Context Model, which is the basis for the following extensions:

- CPP specific vehicle context model: a specific context model that extends the Generic Context Model with vehicle specific measured and basic signals.
- CPP specific smart infrastructure context model: a specific context model that extends the Generic Context Model with smart infrastructure specific measured and basic signals.
- Vehicle and building discovery extended relational context model: a context model

The chosen context modelling tool was Protégé[4], a free, open-source platform that provides the needed functionality for the Cross-CPP context modelling tool, but any tool of your choice that provides the same functionality to create and update ontologies (*.owl files) could be used. The following txt uses Protégé though when an ontology edit is needed.

5.1.1. Basic principles for context modelling

Some basic principles for context modelling were identified, followed within Cross-CPP and should be taken into account when further developing the provided context models:

[3] Link to be added
[4] https://protege.stanford.edu/products.php

1. Support description of main context:
   In practices, we cannot model all context information, and it is also not realistic. The context model should consider those most related factors according to the requirement of context sensitive adoption.
2. Model the context that is easy acquirable:
   Those context factors considered should be identifiable and acquirable, whether provided through context monitoring services automatically, or by user input explicitly.
3. Trade-off between investment of context modelling/extracting and effects of context sensitive adoption:
   Intuitively, if we could model as much context factors in as much details, the accuracy of context will be higher. However, this does not come for free. On the one hand, more time and efforts are need on context modelling; on the other hand, more computing resources are needed to handle the context, which will bring deficiency to the adoption process.

For each CPP, one has to define which concepts are relevant for the description of the situations (context), under which the CPP signals are generated and measured. Once of the concepts relevant to the description of context of CPP data streams generation are defined, the next step is to define the concepts which are relevant for the application (Cross-CPP module, filtering to be applied, discovery process) where it will be used. As a first approach, and because there are so many CPP measured signals, some general situations are considered (situations that could be interesting for a wide range of cross-sectorial services) and that could bring the most benefit in terms of filtering capabilities that could be offered on the extracted context basis.

The process for defining the Context Model(s) for a CPP is as follows:

- Define CPP specific Context Model (starting from the Generic context model)
- Select functionality of the application for which the context model has to be specified.
- From the existing ontologies and the CPP specific Context Model select a concept relevant for adaptation of the functionality.
- Check whether or not exist CPP basic or measured data in the CPP Cloud Storage
  - If it exists, the concept may be adopted.
  - If it does not exist, check if there are sensor signals from the OEM Backends that are still not yet configured but could be also collected.
  - If this also does not exist, check if is it reasonable to introduce new CPP data concept in the CIDM to provide sensor measured data for it.
- If the concept is selected – define the relative weighing of this concept in the extraction of the current context.
- Repeat the process for each functionality where the extracted context is to be applied

The process is iterative, i.e. based on analysis of the 'needs' of each functionality and service, the initial model can be updated, and the process repeated.

5.1.2. Generic context model

The main entities in the generic context model are:

- CPP: comprising all CPPs that are possible
- Activity: the type of activities that can be identified
- Information:
  - Basic CPP Information, comprising all CPP information that are intrinsic for the CPP and do not change (at least not often) such as vehicle colour, sensor height, etc.
  - Sensor Measurement Data, comprising of all sensor signals that can possibly be collected in a certain CPP
- Stakeholder, the actors that are involved in the CPP information value chain

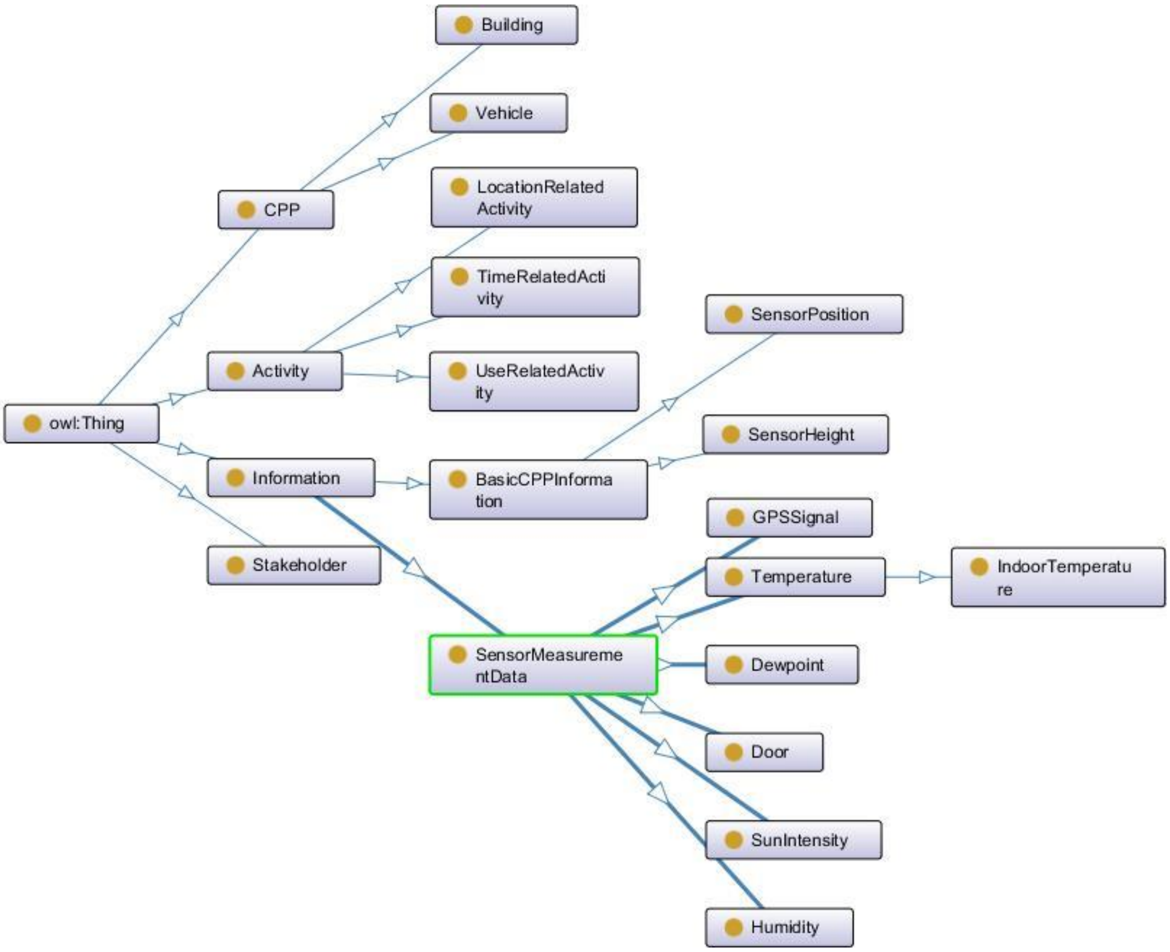


Figure 9: Generic context model

This context model is the basis for the CPP specific context models that exist or will be modelled when that Cross-CPP Marketplace is opened for more OEM data providers.

5.1.3. Creating/Editing of context models

In order to edit any of the currently available CPP specific context models, you need to open the respective ontology in Protégé (*Cross-CPP_Context_Model_v042_vehicle.owl* or *Cross-CPP_Context_Model_v042_building.owl*) and make the necessary edit such as:

1. Add new Sensor Measurement Data
2. Add new Basic CPP Information
3. Add new Stakeholders
4. Save, close and upload in the Context Model repository of the CME module (see more details in section 5.3).

To create a context model for a new CPP you need to:

1. create a new ontology file in Protégé
2. import the generic context model ontology (copy the *genericOntology_v42.owl* file to your working folder and make a direct import in the ontology project that you have created).
3. Follow the same steps as described above for edition of existing ontologies.

The above mentioned models are at the moment available on request (please direct your emails to context-support@cross-cpp.eu) but they will be available in a public GitHub repository free for download under a license to be defined.

The addition of new context models needs to be customised on the Context Monitor component. The CME module is provided as an open source code project on GitHub[5] under the EPL 2.0 license and can be customised. Customisation can also be request via the context-support@cross-cpp.eu.

5.2. Reasoning Rules Configuration

A part of the main adaption work to be done when customising the Context monitoring & extraction module is the introduction of new rules and changing the existing ones. For this purpose, the CME framework provides interfaces both within the code as well as in the form of a freely adaptable configuration file. Both will be described within this section.

5.2.1. Extraction rules configuration file

The configuration file can be found within the main folder of the CME module and is named *extraction_configuration.xml*. Within this file, the following sections can be found which can be adapted by the administrator:

- setup of the reasoning rules for the context-sensitive data discovery

[5] Link to be added

- setup of the reasoning rules for the context-sensitive data access control for the data owner (security feature)
- configuration of the reasoning rules

Setup of the reasoning rules for context-sensitive data discovery

The following

Figure 10 shows the reasoning rules that can be applied by the context-sensitive filtering of data packages during the data discovery as they are listed on the UI of the Marketplace within the *Additional configuration* tab.

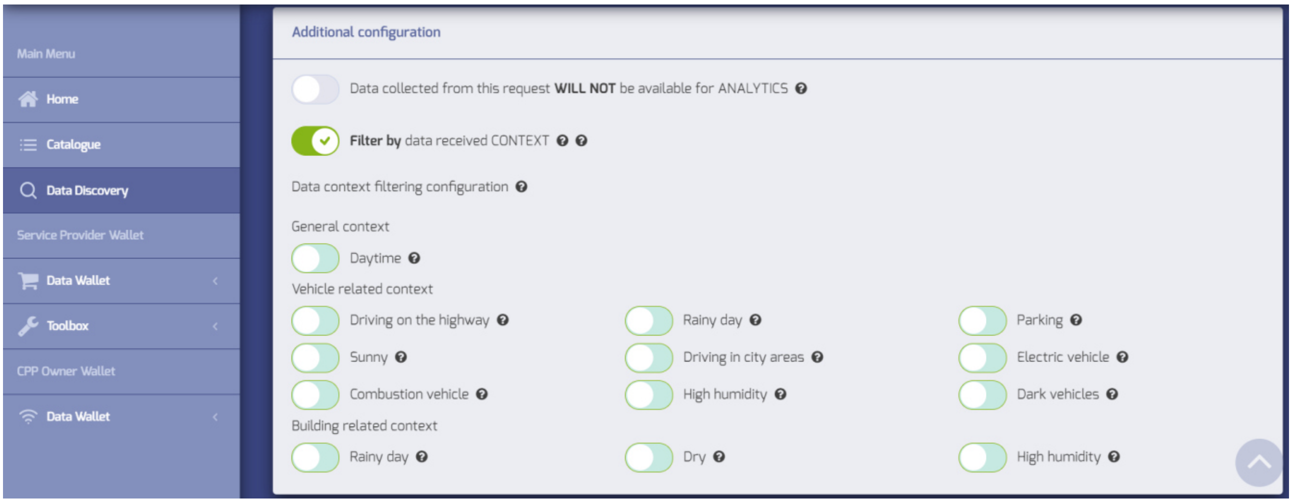


Figure 10. Context-sensitive data filtering in the Cross-CPP Marketplace UI

In the configuration file, the administrator will find the following structure for building this list of filtering options:

```
<rules>
   <rule id={unique_id} name={rule_name} cppType={cpp_type}
        tooltip={tooltip} />
</rules>
```

Each rule for the context-sensitive data filtering has the following attributes:

- *unique_id:* a unique identification number
- *rule_name:* a name indicating the meaning or scope of the rule
- *cpp_type:* the cpp type to which this rule should be applied in the extraction service (by default *vehicle* and/or *building*)
- *tooltip:* a tooltip which gives more information about the semantic of the rule, which will be shown also in the Marketplace UI

A rule can simply be registered by adding a new *<rule>* element to the *<rules>* section.

Hint: Pay special attention to use user-friendly names, descriptions and tooltips within your rule configuration in order to make them easier accessible for the end users. Make sure to define only rules for which exists a valid configuration within a *<ruleConfiguration>* section.

Setup of the reasoning rules for the context-sensitive data access control for data owners

Find in

Figure 11 a snippet of the context-sensitive data access control options for the data owner in the Marketplace UI, which allows him to control his data access according to specific context parameters.

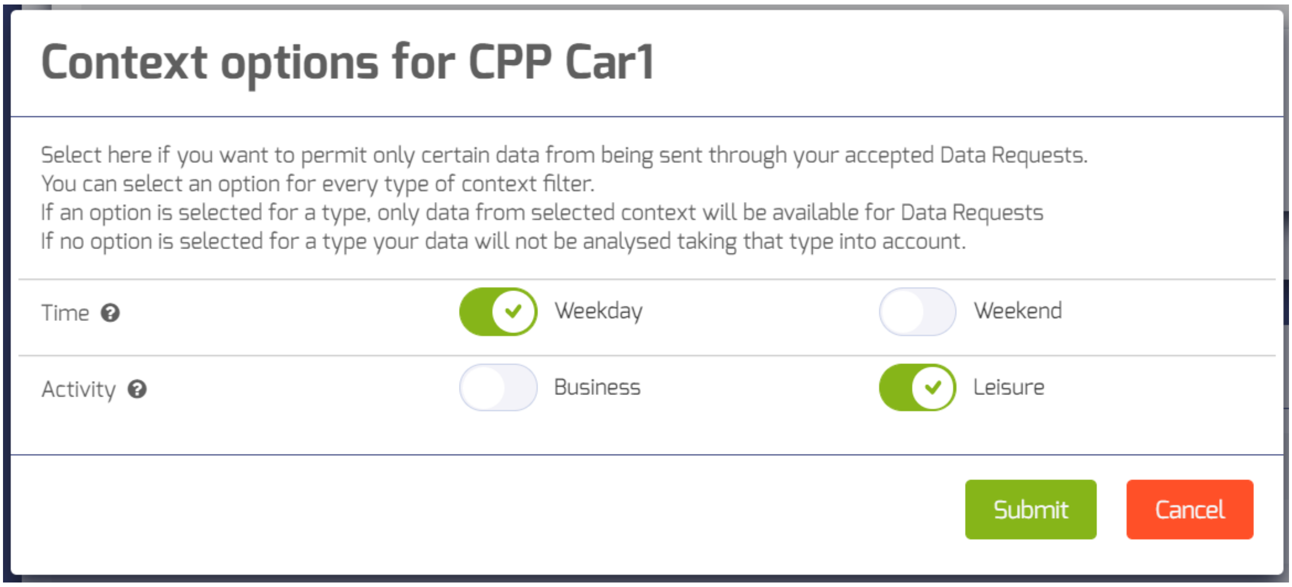


Figure 11. context-sensitive data access control options in the Cross-CPP Marketplace UI

In the configuration file, these options are being defined within the *<dataOwnerContextOptions>* element. Each Option consists of a set of different alternatives for this option and has the following attributes:

- *unique_name:* a unique name for this option
- *tooltip:* a description for this option to be shown as tooltip in the Marketplace UI

Enclosed to an option are different alternatives, specified within the *<alternative>* element as shown below.

```
<dataOwnerContextOptions>
   <option name={unique_name} tooltip={tooltip}>
      <alternative>{alternative1}</alternative>
      <alternative>{alternative2}</alternative>
   </option>
</dataOwnerContextOptions>
```

Hint: in order to function as desired, the choice of options and alternatives has to be aligned with the set of context variables used by the Cross-CPP security module's access control policy. Also make sure to introduce only options and alternatives which correspond to a valid rule configuration (see later in this section).

Configuration of the reasoning rules

Each rule specified within *<rules>* can be configured within a corresponding *<ruleConfiguration>* element. Find the template for the configuration below

```
<ruleConfiguration id={unique_id} internalName={rule_name} cppType={cpp_type}>
   <signalConfiguration id={measurement_channel_id}>
      <value>{value}</value>
   </signalConfiguration>
   <signalConfiguration id={measurement_channel_id}>
      <max>{maximum_value}</max>
      <min>{minimum_value}</min>
   </signalConfiguration>
</ruleConfiguration>
```

The attributes for a rule configuration are:

- *unique_id:* a unique identifier of the rule corresponding to those specified within *<rules>* element
- *rule_name:* the (module internal) name of the rule
- *cpp_type:* the CPP type this rule applies to (by default *vehicle* or *building*)

Each rule configuration is being accompanied by a set of signal configurations, indicating the measurement channels used within the rule. For each measurement channel involved, a *<signalConfiguration>* element will be introduced. The signal configuration specifies the parameter value operators, to which the rule should be applied to. Examples are *maximum (signal < maximum)*, *minimum (signal > minimum)* and *value (signal = value)*. Each of the value operators can be defined within a corresponding *<max>*, *<min>* or *<value>* element, as can be seen in the template above.

5.2.2. Source code customisation

In order to make additional defined reasoning rules function correctly or to edit the already existing rules, the administrator has to adapt the existing CME source code at some specific spots, which will be shown and explained in the following section. Changes will have to be made within the following java classes[6]:

- *ExtractionRule:* In order to make the CME recognise the rule this interface has to be implemented
- *ExtractionOptions:* recently created reasoning rules have to be registered here

For each reasoning rule defined in *<rules>* a corresponding class implementing the *ExtractionRule* has to be implemented. The abstract class *ExtractionRule* provides the following methods to be extended:

- *void readConfiguration(string url):* method to parse the *extraction_configuration.xml* and provide information about the attributes and parameters

[6] Link to GitHub repository to be added here

- *boolean applyRule():* defines the core reasoning logic, which evaluates to *true* if the rule applies to the given set of data

The following class attributes are available and have to be set correctly:

- *id:* the unique identifier of the rule
- *keyword:* a name/description of the rule, which can match with the *internalName* specified within the configuration file
- *CPPType:* the CPP type to which the rule should be applied

A template for reading the extraction configuration is available in the already existing rule implementations and can be used for creating new rules.

In order to make CME recognise and use the newly defined rules they first have to registered in *ExtractionOptions*, as seen below. Once the rules are registered here the extraction service will use them at the next module start up.

```
registerRule(new RuleIsDrivingOnHighway(CONFIG_PATH));
registerRule(new RuleIsWeekday(CONFIG_PATH));
registerRule(new RuleIsWeekend(CONFIG_PATH));
registerRule(new RuleIsBusiness(CONFIG_PATH));
registerRule(new RuleIsLeisure(CONFIG_PATH));

... register here the additional rules
```

5.3. Updating the context model and measurement channel list

Before uploading a new context model into CME it has to be ensured that the signal names match those provided within the Cross-CPP Marketplace, otherwise the respective signal may not be recognized by the monitoring service. If the preconditions are met, just copy the new context model (see section 5.1.3 on how to edit and create a context model) into folder *context-monitoring-extraction/resources* and overwrite the existing one (make a backup before). The CME service has to be restarted to recognize the changed context model.

The measurement channel list with the mapping of the measurement channel IDs to the measurement channel names can be updated in the class *MeasurementChannelIdsEnum*. This mapping is being loaded by the context monitoring service during runtime in order to recognize the valid measurement channels to work with. Here also has to be ensured that the names of the measurement channels match with those provided in the context model, otherwise the respective measurement channel may not be recognized by the monitoring and extraction service.

Further customisation is possible by developers as the CME module is provided as an open source code project on GitHub[7] under the EPL 2.0 license. Furthermore, customisation can also be request via context-support@cross-cpp.eu.

[7] Link to be added

F.A.Q.

Cross-CPP data-marketplace

Q: What is Cross-CPP data-marketplace?

A: Cross-CPP data-marketplace connects Data Providers and Data Consumers for selling and acquiring Connected Vehicle and Home Building data under the Common Industrial Data model (CIDM). It offers a secure and privacy preserving experience when selling or buying sharing big data, by having the full control over your data shared, to whom and for what purposes.

Cross-CPP offers to cross-sectorial Data Consumers, the possibility to search for more than 200 sensor signals, display advance visualization representations (such as Histograms, Geo-Histograms or Time Series) and retrieve those datasets in a seamless experience thanks to the open SDK-API created.

Q: How do I, as data provider, register into Cross-CPP data-marketplace?

A: You can find the registration form by clicking the "Sign on!" button in the landing page. Select "Original Equipment Manufacturer" role and fill the fields to request your registration. Once your registration is validated by a system administrator an email will be sent to you to confirm your access,

Q: What do I have to do in order to start working with CROSS-CPP data-marketplace?

A: Once registered you must be familiar with the CIDM, as it is the format in which you will receive the data you request.

Cross-CPP data model

Q: What is the Common Industrial Data Model (CIDM)?

A: The CIDM is a standardized data model for industrial data-driven services.

Q: Which are the benefits and advantages of using the CIDM model for data -driven services:

A: -The CIDM constitute a major business and technical advantage for Data Consumers:

- The CIDM provides a brand-independent and transparent data model, which harmonizes proprietary data into generic datasets independently of any cross-sectorial Industry
- It is built on an open and highly scalable automotive big data format (JSON Schema).
- Active community of service providers increasing the number of signals available from vehicles and Smart Buildings to be recorded as well as the type of measurement channels can be modified or extended
- The Data Provider also provides an origin certification as a CIDM feature to support the validation and verification of origin, integrity and completeness of data. The intention is to protect the data inside the Data Package against manipulation.

Q: What is a signal?

A: A signal is the information provider of each CPP. They are the perception organs of CPPs and it is their main duty to detect physical phenomenon and chemical quantities. They observe the environment and generate data in the CIDM format. An example could be "speed" or "latitude"

Q: What is a channel?

A: A channel is the way the physical signals and their sampled measurements are implemented and represented in the CIDM format. Some examples could be "Vehicle Speed" using the signal "Speed" in a time-series or in a histogram format, or "Position" using both "Latitude" and "Longitude" signals.

Q: Can I request a new signal or channel?

A: Cross-CPP data-marketplace offers a wide variety of signals provided by the manufacturers. The catalogue is really extensive and can be filtered in many ways. If even then you can't find the signal that you need and/or think can be provided by any of our data providers, please contact us in: cross-cpp-support@lists.atosresearch.eu.

Cross-CPP marketplace components

Q: What is the Context Monitoring and Extraction module and how can it help me?

A: The Context Monitoring and Extraction module allows Cross-CPP to suggest signals to add to your current Data Discovery filters, based on the context model of the signals already selected. This might help you find data of interest that you would miss otherwise.

Q: Can I add a new context sensitive filter to the discovery process?

A: This is possible by following the steps described in section 5 as the CME module is provided as an open source code project on GitHub[8] under the EPL 2.0 license. Furthermore, customisation can also be request via context-support@cross-cpp.eu.

Q: Can I get more information on how the context is being calculated?

A: In the context sensitive filter within the discovery process each context option is accompanied by a tooltip in the UI. Hovering over it will show the logic behind each context filtering option.

[8] Link to be added

Glossary

Administrator: Cross-CPP marketplace system administrator

AEON: AEON application

AEON application: publication/subscription based communication application

AEON channel: set configuration for communication between two actors through AEON application

CB: Company Backend

CIDM: Common Industrial Data Model

CIDM model: standardized data model for industrial data-driven services

Cloud Storage: Storage system deployed by Cross-CPP solution to store CPP owners data coming from Data Providers

CME: Context Monitoring and Extraction

Company Backend: Data Provider backend system to connect to the Cloud Storage

CPP: cyber-physical product

CPP Data: data created by a CPP and sent to the system by the Data Provider

CPP Owner: Data Owner which CPP is registered in the Cross-CPP data-marketplace

Cross-CPP: System

CS: Cloud Storage

Data Consumer: actor who receives the data created by owners to use it on the creation or improvement of services

Data Owner: owner of the CPP that sends data to the system

Data Provider: OEM that provides its users data to the Cross-CPP marketplace

Data Request: set of configurations that define a scope for CPP Data to be received by a Data Consumer

Id: generic document id string (example: `5cd96b65ff89151c002d16b3`)

Marketplace: Marketplace Web Application

Measurement Channel: sampler of the data the signals process

MP: Marketplace

OEM: Original Equipment Manufacturer

Provider: Data Provider

Service Provider: Data Consumer

Signal: information provider of the data the CPP sensors generate

System: the whole lot of applications that conforms CROSS-CPP, including Marketplace Web Application and Marketplace Server.

UUID: universally unique identifier. Standardized 16 bytes Id signature formed by 32 hexadecimal digits (example: `90eb04b2-a07c-4835-8618-9c0140f8391a`)

Figures

Tables

Listings

Annex

OpenAPI Specification of the Company Backend REST API (yaml)


```yaml
---
swagger: "2.0"
info:
  version: 3.0.0
  title: Cloud Storage Provider API Definition
  contact:
    name: Elisa Herrmann
    url: http://www.cross-cpp.eu
    email: elisa.herrmann@atos.net
host: cloudstorage-api.datagora.eu
basePath: /
schemes:
- https
consumes:
- application/json
produces:
- application/json
security:
- api_key: []
paths:
  /users:
    get:
      tags:
      - User Management
      summary: Retrieve all users
      description: This functionalities provides a list of all users within this Cloud Storage Provider. This API call is not specified and may only be used by the Cloud Storage Provider for internal user management. Returns an array of `User` objects.
      operationId: usersGET
      parameters: []
      responses:
        "200":
          description: OK
          schema:
            type: array
            items:
              $ref: '#/definitions/FullUser'
        "401":
          description: Unauthorized
        "403":
```

```
        description: Forbidden
      "404":
        description: Not found
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
  post:
    tags:
    - User Management
    summary: Create new user
    description: API call registers a new user and creates new Cloud Storage Vault at the Provider. This API call is not specified and is intended to be used for internal user management.
    operationId: usersPOST
    parameters:
    - in: body
      name: User
      description: Defines full name, login and password of the user.
      required: true
      schema:
        $ref: '#/definitions/User'
    responses:
      "200":
        description: OK
        schema:
          $ref: '#/definitions/FullUser'
      "400":
        description: Bad request
      "401":
        description: Unauthorized
      "403":
        description: Forbidden
      "409":
        description: Conflict - User already exists
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
 /users/access:
  post:
    tags:
    - Contract Management
    summary: Grant access permission
    description: Acquires access permissions to the `api_key` that is used in this request. If `key`is a `vault-write-key` `write permission`is granted, if `key`is `vault-read-key` `read permission`is granted.
    operationId: usersAccessPOST
    parameters:
```

```
    - in: body
      name: key
      description: (Read or write) Access key to one Cloud Storage Vault.
      required: true
      schema:
        $ref: '#/definitions/key'
    responses:
      "200":
        description: OK
        schema:
          $ref: '#/definitions/inline_response_200'
      "400":
        description: Bad request
      "401":
        description: Unauthorized
      "403":
        description: Forbidden
      "404":
        description: Not found
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
/users/access/{key}:
  get:
    tags:
    - Contract Management
    summary: Validate access token
    description: tbd
    operationId: usersAccessGET
    parameters:
    - name: key
      in: path
      description: Access key for Cloud Storage Vault
      required: true
      type: string
      format: uuid
    responses:
      "200":
        description: OK
        schema:
          $ref: '#/definitions/inline_response_200_1'
    x-swagger-router-controller: Default
  delete:
    tags:
    - Contract Management
    summary: Release access permissions
```

```
      description: Releases access permissions of the `api_key` that is used in this request. If `key`is a `vault-write-key` `write permission`is released, if `key`is `vault-read-key` `read permission`is released.
      operationId: usersAccessKeyDELETE
      parameters:
      - name: key
        in: path
        description: Access key for Cloud Storage Vault
        required: true
        type: string
        format: uuid
      responses:
        "200":
          description: OK
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /datapackages:
    post:
      tags:
      - Data Storage Interface
      summary: Push data packages into Cloud Storage
      description: This API call enables OEMs to write data packages into the cloud. Data packages are stored within a CIDM container structure. Every data package needs to contain the correct vault-id. The OEM needs write permission to the user's Cloud Storage Vault. The Cloud Storage Provider assignes an unique datapackage-id to every delivered data package.
      operationId: datapackagesPOST
      parameters:
      - in: body
        name: Data Package Container
        description: CIDM container structure containing valid and json encoded CIDM data packages
        required: true
        schema:
          type: array
          items:
            $ref: '#/definitions/DataPackage'
      responses:
```

```yaml
      "200":
        description: OK
      "400":
        description: Bad request
      "401":
        description: Unauthorized
      "403":
        description: Forbidden
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
/datapackages/{datapackageid}:
  get:
    tags:
    - Data Provisioning Interface
    summary: Get data package
    description: Retrieve one data package with `datapackage-id`.
    operationId: datapackagesIdGET
    parameters:
    - name: datapackageid
      in: path
      description: Unique data package identifier
      required: true
      type: string
    - name: metadata
      in: query
      description: Retrieve *only* metadata, default=false
      required: false
      type: boolean
      default: false
    responses:
      "200":
        description: OK
        schema:
          $ref: '#/definitions/DataPackage'
      "400":
        description: Bad request
      "401":
        description: Unauthorized
      "403":
        description: Forbidden
      "404":
        description: Not found
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
```

```
/datapackages/query:
  post:
    tags:
    - Data Provisioning Interface
    summary: Query data packages
    description: Allows searching for data packages. Cloud Storage Provider will only include Cloud Storage Vaults into search where `api_key` has read access.
    operationId: datapackagesQueryPOST
    parameters:
    - in: body
      name: query
      description: Query for data packages
      required: true
      schema:
        $ref: '#/definitions/Query'
    responses:
      "200":
        description: OK
        schema:
          type: array
          items:
            $ref: '#/definitions/DataPackage'
      "400":
        description: Bad request
      "401":
        description: Unauthorized
      "403":
        description: Forbidden
      "404":
        description: Not found
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
/datapackages/query_stream:
  post:
    tags:
    - Data Provisioning Interface
    summary: Query data packages
    description: Allows searching for data packages. Cloud Storage Provider will only include Cloud Storage Vaults into search where `api_key` has read access.
    operationId: datapackagesQueryStreamPOST
    parameters:
    - in: body
      name: query
      description: Query for data packages
      required: true
```

```yaml
        schema:
          $ref: '#/definitions/Query'
      responses:
        "200":
          description: OK
          schema:
            type: array
            items:
              $ref: '#/definitions/DataPackage'
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /datapackages/basicCppQuery:
    post:
      tags:
      - Data Provisioning Interface
      summary: Query basic CPP information data packages
      description: Allows searching for data packages. Cloud Storage Provider will only include
Cloud Storage Vaults into search where `api_key` has read access.
      operationId: basicCppQueryPOST
      parameters:
      - in: body
        name: query
        description: Query for data packages
        required: true
        schema:
          $ref: '#/definitions/Query'
      responses:
        "200":
          description: OK
          schema:
            type: array
            items:
              $ref: '#/definitions/DataPackage'
        "400":
          description: Bad request
        "401":
          description: Unauthorized
```

```yaml
      "403":
        description: Forbidden
      "404":
        description: Not found
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
  /datapackages/basicCppQuery_stream:
    post:
    tags:
    - Data Provisioning Interface
    summary: Query basic CPP information data packages
    description: Allows searching for data packages. Cloud Storage Provider will only include
Cloud Storage Vaults into search where `api_key` has read access.
    operationId: basicCppQueryStreamPOST
    parameters:
    - in: body
      name: query
      description: Query for data packages
      required: true
      schema:
        $ref: '#/definitions/Query'
    responses:
      "200":
        description: OK
        schema:
          type: array
          items:
            $ref: '#/definitions/DataPackage'
      "400":
        description: Bad request
      "401":
        description: Unauthorized
      "403":
        description: Forbidden
      "404":
        description: Not found
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
  /notifications:
    post:
    tags:
    - Notifications
    summary: Subscribe Push Notification
```

```yaml
      description: Functionality of the marketplace to subscribe to push notification events. Push notifications are sent, whenever users put data into their Cloud Storage Vaults.
      operationId: notificationsPOST
      parameters:
      - in: body
        name: config
        description: Push Notification URL
        required: true
        schema:
          $ref: '#/definitions/config'
      responses:
        "200":
          description: OK
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
    delete:
      tags:
      - Notifications
      summary: Unsubscribe Push Notification
      description: No more push notifications will be sent from the Cloud Storage Provider to the marketplace.
      operationId: notificationsDELETE
      parameters: []
      responses:
        "200":
          description: OK
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /access:
    get:
      tags:
      - Contract Management
```

```
    summary: Validate authentication token
    description: t.b.d.
    operationId: accessGET
    parameters: []
    responses:
      "200":
        description: OK
        schema:
          $ref: '#/definitions/inline_response_200_2'
      "403":
        description: Forbidden / No authorization token in header
      "404":
        description: Authorization token not found!
      "500":
        description: Internal Error
    x-swagger-router-controller: Default
securityDefinitions:
  api_key:
    description: Provides authentification for OEM and Marketplace. Must be sent in HTTP header.
    type: apiKey
    name: Authentication
    in: header
definitions:
  FullUser:
    allOf:
    - $ref: '#/definitions/User'
    - {}
  User:
    type: object
    required:
    - full-name
    - login-name
    - password
    properties:
      full-name:
        type: string
        description: Full name of the user
      login-name:
        type: string
        format: email
        description: Login name of the user (e-mail)
      password:
        type: string
        description: Password of the user
    example:
```

```
      password: password
      full-name: full-name
      login-name: login-name
  DataPackage:
    type: object
    required:
    - cpp-type
    - cvim-version
    - data
    - measurement-channel-id
    - type
    - vault-id
    properties:
      vault-id:
        type: string
        description: Cloud Storage Vault ID, where the data packages are pushed into.
      datapackage-id:
        type: string
        description: Unique identifier of the data package. Property is set by Cloud Storage Provider.
        readOnly: true
      data:
        type: object
        description: CIDM data
        properties: {}
      cvim-version:
        type: string
        description: Version of the CIDM protocol >=1.1.2
      type:
        type: string
        description: Type of the Data Package
      cpp-type:
        type: string
        description: Type of the CPP device, "vehicle" or "builging"
      cpp-id:
        type: string
        description: CPP ID for identifying vehicles or bulding of the same owner and vault-id
      measurement-channel-id:
        type: string
        description: Identifier of the Measurement Channel whose data is inside this data package
      timestamp:
        type: string
        description: Measurement timestamp (basic-cpp-information and event-based)
      timestamp-start:
        type: string
        description: Measurement start time
```

```
    timestamp-stop:
      type: string
      description: Measurement stop time
    mileage-start:
      type: number
      description: Mileage at the start of measurement in kilometres (km)
    mileage-stop:
      type: number
      description: Mileage at the end of measurement in kilometres (km)
    geo-bounding-box:
      type: object
      description: geographic bounding box (see reference manual section 6.5.1.1)
      properties: {}
    room-id:
      type: string
      description: Identifier of the room in a building cpp-type
    oem-certification:
      type: object
      description: OEM certification (see reference manual section 6.5.1.2)
      properties: {}
    ownership-information:
      type: object
      description: Data Ownership Information (see reference manual section 6.5.1.3)
      properties: {}
    expiration-date:
      type: string
      description: Data expiration date
    data-masking-active:
      type: boolean
      description: Indicates status of data-masking (true = active)
  example:
    datapackage-id: datapackage-id
    vault-id: vault-id
    data: '{}'
    data-masking-active: true
    cvim-version: cvim-version
    type: type
    mileage-stop: 6.027456183070403
    measurement-channel-id: measurement-channel-id
    timestamp-start: timestamp-start
    geo-bounding-box: '{}'
    mileage-start: 0.8008281904610115
    oem-certification: '{}'
    ownership-information: '{}'
    timestamp-stop: timestamp-stop
    expiration-date: expiration-date
```

```
Query:
  type: object
  properties:
    datapackage-id:
      type: array
      description: Array of Data Package IDs
      items:
        type: string
        description: Data Package ID
    measurement-channel-id:
      type: array
      description: Array of Measurement Channel IDs
      items:
        type: string
        description: Measurement Channel ID
    vault-id:
      type: array
      description: Array of Cloud Storage Vault IDs
      items:
        type: string
    submit-time:
      $ref: '#/definitions/Query_submittime'
    metadata:
      type: boolean
      description: Request only metadata, default=off
      default: false
  example:
    datapackage-id:
    - datapackage-id
    - datapackage-id
    submit-time:
      min: 2000-01-23T04:56:07.000+00:00
      max: 2000-01-23T04:56:07.000+00:00
    metadata: false
    vault-id:
    - vault-id
    - vault-id
    measurement-channel-id:
    - measurement-channel-id
    - measurement-channel-id
key:
  type: object
  properties:
    vault-access-key:
      type: string
      format: uuid
```

```
      description: Access key for Cloud Storage Vault
config:
  type: object
  properties:
    handler-url:
      type: string
      format: uuid
      description: URL for push notifications
    level:
      type: string
      description: Defines the level of the notification ('id-only', 'metadata' or 'full')
Query_submittime:
  properties:
    min:
      type: string
      format: date-time
      description: Earliest Data Package submission time
    max:
      type: string
      format: date-time
      description: Latest Data Package submission time
  description: Data Package submission time
  example:
    min: 2000-01-23T04:56:07.000+00:00
    max: 2000-01-23T04:56:07.000+00:00
inline_response_200:
  type: object
  properties:
    full-name:
      type: string
    vault-id:
      type: string
      description: ID of the user's Cloud Storage Vault.
inline_response_200_1:
  type: object
  properties:
    full-name:
      type: string
    vault-id:
      type: string
      format: uuid
      description: ID of the user's Cloud Storage Vault.
    type:
      type: string
      description: '`read` or `write` access key'
    in-use:
```

```
      type: boolean
      description: '`true` when key is already in use, otherwise `false`'
inline_response_200_2:
  type: object
  properties:
    name:
      type: string
    type:
      type: string
```

CIDM v1.2.1 jsonSchema


```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "description": "Common Industrial Data Model",
    "type": "object",
    "properties": {
        "Signal": {
            "$ref": "#/definitions/Signal"
        },
        "MeasurementChannel": {
            "$ref": "#/definitions/MeasurementChannel"
        },
        "DataPackage": {
            "$ref": "#/definitions/DataPackage"
        }
    },
    "additionalProperties": false,
    "definitions": {
        "TimeSeriesMeasurementChannel": {
            "title": "TimeSeriesChannel",
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "time-series"
                    ]
                },
                "capture-interval": {
                    "type": "number"
                },
                "on-change": {
                    "type": "boolean"
                },
                "sample-strategy": {
                    "type": "string",
                    "enum": [
                        "min",
                        "max",
                        "average",
                        "last-known-value"
                    ]
                },
```

```
            "signal": {
                "$ref": "#/definitions/Signal"
            },
            "format": {
                "type": "string"
            },
            "dimension": {
                "type": "number"
            }
        },
        "required": [
            "type",
            "capture-interval",
            "on-change",
            "sample-strategy",
            "signal"
        ]
    },
    "MeasurementChannel": {
        "type": "object",
        "properties": {
            "id": {
                "type": "string"
            },
            "name": {
                "type": "string"
            },
            "type": {
                "type": "string",
                "enum": [
                    "time-series",
                    "histogram",
                    "geo-histogram",
                    "general-purpose",
                    "event-based",
                    "basic-cpp-information"
                ]
            },
            "comment": {
                "type": "string"
            }
        },
        "required": [
            "id",
            "name",
            "type"
        ],
        "oneOf": [
            {
                "$ref": "#/definitions/TimeSeriesMeasurementChannel"
            },
            {
                "$ref": "#/definitions/HistogramMeasurementChannel"
            },
            {
                "$ref": "#/definitions/GeoBasedHistogramMeasurementChannel"
            },
```

```
            {
                "$ref": "#/definitions/GeneralPurposeMeasurementChannel"
            },
            {
                "$ref": "#/definitions/BasicCppInformationMeasurementChannel"
            },
            {
                "$ref": "#/definitions/EventBasedMeasurementChannel"
            }
        ]
    },
    "HistogramMeasurementChannel": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "histogram",
                    "geo-histogram"
                ]
            },
            "aggregation-strategy": {
                "type": "string",
                "enum": [
                    "time",
                    "count",
                    "min",
                    "max"
                ]
            },
            "capture-interval": {
                "type": "number"
            },
            "dimensions": {
                "type": "integer",
                "minimum": 1
            },
            "bins": {
                "type": "array",
                "minItems": 1,
                "items": {
                    "type": "object",
                    "properties": {
                        "type": {
                            "type": "string",
                            "enum": [
                                "linear",
                                "logarithmic",
                                "custom"
                            ]
                        },
                        "lower-bound": {
                            "type": "number"
                        },
                        "upper-bound": {
                            "type": "number"
                        },
```

```
        "signal": {
            "$ref": "#/definitions/Signal"
        },
        "number-of-bins": {
            "type": "integer",
            "minimum": 0
        },
        "alternative-bin-labels": {
            "type": "array",
            "items": {
                "type": "string"
            }
        }
    },
    "required": [
        "type",
        "lower-bound",
        "upper-bound",
        "signal",
        "number-of-bins"
    ],
    "oneOf": [
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "linear",
                        "logarithmic"
                    ]
                }
            },
            "required": [
                "type"
            ]
        },
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "custom"
                    ]
                },
                "custom-bounds": {
                    "type": "array",
                    "items": {
                        "type": "number"
                    }
                }
            },
            "required": [
                "type",
                "custom-bounds"
            ]
```

```
                        }
                    ]
                }
            }
        },
        "required": [
            "type",
            "aggregation-strategy",
            "capture-interval",
            "dimensions",
            "bins"
        ],
        "additionalProperties": false
    },
    "GeoBasedHistogramMeasurementChannel": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "geo-histogram"
                ]
            },
            "geo-resolution": {
                "type": "number"
            }
        },
        "required": [
            "type",
            "geo-resolution"
        ],
        "additionalProperties": false,
        "allOf": [
            {
                "$ref": "#/definitions/HistogramMeasurementChannel"
            }
        ]
    },
    "GeneralPurposeMeasurementChannel": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "general-purpose"
                ]
            },
            "signal": {
                "$ref": "#/definitions/Signal"
            }
        },
        "required": [
            "type",
            "signal"
        ],
        "additionalProperties": false
    },
```

```json
"BasicCppInformationMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "basic-cpp-information"
            ]
        },
        "signal": {
            "$ref": "#/definitions/Signal"
        }
    },
    "required": [
        "type",
        "signal"
    ],
    "additionalProperties": false
},
"EventBasedMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "event-based"
            ]
        },
        "format": {
            "type": "string"
        },
        "event-sample-strategy": {
            "type": "string",
            "enum": [
                "real-time-event",
                "trigger-event",
                "threshold-event"
            ]
        },
        "comment": {
            "type": "string"
        }
    },
    "required": [
        "type",
        "signal",
        "event-sample-strategy"
    ],
    "additionalProperties": false
},
"DataPackage": {
    "type": "object",
    "properties": {
        "datapackage-id": {
            "type": "string"
        },
        "vault-id": {
```

```
        "type": "string"
    },
    "trip-id": {
        "type": "string"
    },
    "cpp-id": {
        "type": "string"
    },
    "cvim-version": {
        "type": "string",
        "enum": [
            "1.0.0",
            "1.0.1",
            "1.2.0",
            "1.2.1"
        ]
    },
    "cpp-type": {
        "type": "string",
        "enum": [
            "vehicle",
            "building"
        ]
    },
    "type": {
        "type": "string",
        "enum": [
            "time-series",
            "histogram",
            "geo-histogram",
            "general-purpose",
            "event-based",
            "basic-cpp-information"
        ]
    },
    "measurement-channel-id": {
        "type": "string"
    },
    "submit-time": {
        "type": "string",
        "format": "date-time"
    },
    "mileage-start": {
        "type": "number"
    },
    "room-id": {
        "type": "string"
    },
    "mileage-stop": {
        "type": "number"
    },
    "geo-bounding-box": {
        "type": "object",
        "properties": {
            "latitude-min": {
                "type": "number"
            },
```

```
            "latitude-max": {
                "type": "number"
            },
            "longitude-min": {
                "type": "number"
            },
            "longitude-max": {
                "type": "number"
            },
            "altitude-min": {
                "type": "number"
            },
            "altitude-max": {
                "type": "number"
            }
        },
        "additionalProperties": false
    },
    "location": {
        "type": "object",
        "properties": {
            "latitude": {
                "type": "number"
            },
            "longitude": {
                "type": "number"
            }
        },
        "additionalProperties": false
    },
    "oem-certification": {
        "type": "object",
        "properties": {
            "signature": {},
            "checksum": {},
            "sequence-number": {}
        },
        "additionalProperties": false
    },
    "expiration-date": {
        "type": "string",
        "format": "date-time"
    },
    "data-ownership-information": {
        "type": "object",
        "properties": {
            "privacy-veto-rights": {
                "type": "object",
                "properties": {
                    "consent-level": {
                        "type": "string",
                        "enum": [
                            "public",
                            "shared",
                            "private"
                        ]
                    },
```

```
                "data-format": {
                    "type": "string",
                    "enum": [
                        "time-series",
                        "histogram"
                    ]
                },
                "jurisdiction": {
                    "type": "string",
                    "enum": [
                        "Europe",
                        "any"
                    ]
                },
                "storage-constraint": {
                    "type": "string",
                    "enum": [
                        "OEM storage",
                        "Personal storage"
                    ]
                }
            },
            "required": [
                "consent-level"
            ]
        },
        "copyright-stakeholders": {
            "type": "array",
            "items": [
                {
                    "type": "object",
                    "properties": {
                        "name": {
                            "type": "string"
                        },
                        "status": {
                            "type": "string"
                        }
                    },
                    "required": [
                        "name",
                        "status"
                    ],
                    "additionalProperties": false
                }
            ]
        },
        "data-stakeholders": {
            "type": "array",
            "items": [
                {
                    "type": "object",
                    "properties": {
                        "name": {
                            "type": "string"
                        },
                        "status": {
```

```
                                "type": "string"
                            }
                        },
                        "required": [
                            "name",
                            "status"
                        ],
                        "additionalProperties": false
                    }
                ]
            },
            "data-privacy-level": {
                "type": "string",
                "enum": [
                    "public",
                    "shared",
                    "private"
                ]
            }
        },
        "additionalProperties": false
    },
    "data-masking-active": {
        "type": "boolean"
    },
    "signatures": {
        "type": "array",
        "items": [
            {
                "type": "object",
                "properties": {
                    "signatory": {
                        "type": "string"
                    },
                    "checksum": {
                        "type": "string"
                    },
                    "signature": {
                        "type": "string"
                    }
                },
                "required": [
                    "signatory",
                    "checksum",
                    "signature"
                ],
                "additionalProperties": false
            }
        ]
    }
},
"required": [
    "cvim-version",
    "type",
    "measurement-channel-id",
    "vault-id",
    "cpp-type"
```

```
],
"oneOf": [
    {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "time-series"
                ]
            },
            "number-of-samples": {
                "type": "integer",
                "minimum": 1
            },
            "statistic-properties": {
                "type": "object",
                "properties": {
                    "min": {
                        "type": "number"
                    },
                    "max": {
                        "type": "number"
                    },
                    "average": {
                        "type": "number"
                    },
                    "histogram": {
                        "type": "object",
                        "properties": {
                            "measurement-channel-id": {
                                "type": "string"
                            },
                            "data": {
                                "type": "array",
                                "minItems": 0,
                                "maxItems": 10,
                                "items": {
                                    "type": "number"
                                }
                            }
                        },
                        "required": [
                            "measurement-channel-id",
                            "data"
                        ]
                    }
                }
            },
            "data": {
                "type": "array",
                "minItems": 1,
                "items": {
                    "type": "object",
                    "properties": {
                        "timestamp": {
                            "type": "string",
```

```json
                        "format": "date-time"
                    },
                    "value": {
                        "type": [
                            "array",
                            "string",
                            "number",
                            "boolean"
                        ],
                        "items": {
                            "type": [
                                "string",
                                "number",
                                "boolean"
                            ],
                            "minLength": 1
                        }
                    }
                }
            }
        },
        "timestamp-start": {
            "type": "string",
            "format": "date-time"
        },
        "timestamp-stop": {
            "type": "string",
            "format": "date-time"
        }
    },
    "required": [
        "type",
        "number-of-samples",
        "data",
        "timestamp-start",
        "timestamp-stop"
    ]
},
{
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "histogram",
                "geo-histogram"
            ]
        },
        "data": {
            "type": "array",
            "items": {
                "type": [
                    "number",
                    "array"
                ],
                "minItems": 1,
                "items": {
```

```
                        "type": "number"
                    }
                }
            },
            "timestamp-start": {
                "type": "string",
                "format": "date-time"
            },
            "timestamp-stop": {
                "type": "string",
                "format": "date-time"
            }
        },
        "required": [
            "type",
            "data",
            "timestamp-start",
            "timestamp-stop"
        ]
    },
    {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "general-purpose",
                    "basic-cpp-information"
                ]
            },
            "data": {},
            "timestamp": {
                "type": "string",
                "format": "date-time"
            }
        },
        "required": [
            "type",
            "data",
            "timestamp"
        ]
    },
    {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "event-based"
                ]
            },
            "data": {
                "type": "object",
                "properties": {
                    "event-type": {
                        "type": "string",
                        "enum": [
```

```
                                        "real-time-event",
                                        "trigger-event",
                                        "threshold-event"
                                    ]
                                },
                                "event-data": {
                                    "type": "object",
                                    "properties": {
                                        "value": {
                                            "type": "string"
                                        },
                                        "event-datapackages": {
                                            "type": "array",
                                            "minItems": 1,
                                            "items": {
                                                "$ref": "#/definitions/DataPackage"
                                            }
                                        }
                                    },
                                    "required": [
                                        "value"
                                    ]
                                }
                            },
                            "required": [
                                "event-type",
                                "event-data"
                            ]
                        },
                        "timestamp": {
                            "type": "string",
                            "format": "date-time"
                        }
                    },
                    "required": [
                        "type",
                        "data",
                        "timestamp"
                    ]
                }
            ]
        },
        "Signal": {
            "title": "CIDM Signal",
            "description": "Signals are the perception organs of vehicles. It is their ma
in duty to detect physical phenomenons and chemical quantities by transferring them into
electrical signals. They observe the environment and gener-
ate the data that is exchangeable at AutoMat's marketplace. They are one of the core comp
onents of the AutoMat project. Figure 11 shows the UML modelling of the signals. Within A
utoMat all information pro-
viders are modelled as Signal. They can be classified as static signals or changing/non-
static signals, having a sample rate larger than zero. ",
            "type": "object",
            "properties": {
                "id": {
                    "type": "string"
                },
```

```
        "name": {
            "type": "string"
        },
        "cpp-type": {
            "type": "string",
            "enum": [
                "vehicle",
                "building"
            ]
        },
        "type": {
            "type": "string",
            "enum": [
                "numeric",
                "information",
                "enumeration",
                "general-purpose"
            ]
        },
        "format": {
            "type": "string"
        },
        "sample-rate": {
            "type": "number",
            "minimum": 0
        },
        "comment": {
            "type": "string"
        }
    },
    "required": [
        "id",
        "name",
        "type",
        "sample-rate",
        "cpp-type"
    ],
    "oneOf": [
        {
            "$ref": "#/definitions/NumericSignal"
        },
        {
            "$ref": "#/definitions/EnumerationSignal"
        },
        {
            "$ref": "#/definitions/InformationSignal"
        },
        {
            "$ref": "#/definitions/GeneralPurposeSignal"
        }
    ]
},
"GeneralPurposeSignal": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
```

```
                "enum": [
                    "general-purpose"
                ]
            }
        },
        "required": [
            "type"
        ]
    },
    "InformationSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "information"
                ]
            },
            "format": {
                "type": "string"
            }
        },
        "required": [
            "type",
            "format"
        ]
    },
    "EnumerationSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "enumeration"
                ]
            },
            "items": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            }
        },
        "required": [
            "type",
            "items"
        ]
    },
    "NumericSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "numeric"
                ]
            },
```

```
                "format": {
                    "type": "string",
                    "enum": [
                        "int",
                        "uint",
                        "float",
                        "double"
                    ]
                },
                "min": {
                    "type": "number"
                },
                "max": {
                    "type": "number"
                },
                "resolution": {
                    "type": "number"
                },
                "unit": {
                    "type": "string"
                }
            },
            "required": [
                "type",
                "format",
                "min",
                "max",
                "resolution",
                "unit"
            ]
        }
    }
}
```

About Cross-CPP

The objective is to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research is to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project will develop: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to pro-vide enhanced local weather forecast and predict and optimise energy consumptions in households. The project will build upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, will be used as a basis for further development aiming to extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

Funded by the Horizon 2020 Framework Programme of the European Union

4 Annex 3: User Guide Data Owner

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

CROSS-CPP

Cross-CPP MARKETPLACE

USER GUIDE

CPP DATA OWNER GUIDE

(VEHICLE / BUILDING OWNER)

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 780167

Introduction

CPP owner User Guide describes CPP owner functionalities within Cross-CPP Marketplace, such as discovering available data, browsing available data requests, managing accepted requests and browsing data sent to the marketplace and to the Service Providers.

Purpose

This guide aims to help users from a CPP owner role on how to use the platform and gives knowledge about the different functionalities available.

Audience

This guide is meant for and solely for users of Cross-CPP Marketplace with CPP owner role. Other roles can find their own user guides under cross-cpp.eu.

Scope

The contents of this guide are meant to be taken into consideration only when using the Cross-CPP Marketplace and will only cover functionalities meant to be used by the role stated above.

Cross-CPP team does not take responsibility on bad use of the application or the data provided when not following the instructions given in this guide.

Troubleshooting

For any questions or inquiries about the use of the Cross-CPP Marketplace web application or the contents of it or this guide, or if you find there is no content in this guide for some functionality please forward it to: marketplace-support@cross-cpp.eu.

Contact

Cross-CPP Project website: https://cross-cpp.eu

Cross-CPP Marketplace: https://datagora.eu

Marketplace support: marketplace-support@cross-cpp.eu

Context Monitoring and Extraction Module (CME): context-support@cross-cpp.eu.

Contents

Guide

There are two main functionalities in the Cross-CPP Marketplace (Figure 1):

- **Catalogue**: list of all available registered signals in the Cross-CPP Marketplace and their assigned measurement channels. (2.1 Catalogue)
- **Data Discovery**: discover the data from your registered CPPs available in the Cross-CPP Marketplace through a set of filters. (2.2 Data Discovery)

Cross-CPP Marketplace offers a section specifically for CPP owners apart from the main functionalities (Figure 2):

- **My Data Wallet**: is considered as the owner central point where they can take the control of their user profile options, data requests acceptances and data transactions information. (3 My Data Wallet)

At a glance, Cross-CPP Marketplace main functionalities can be accessed directly through the side menu (Figure 1).

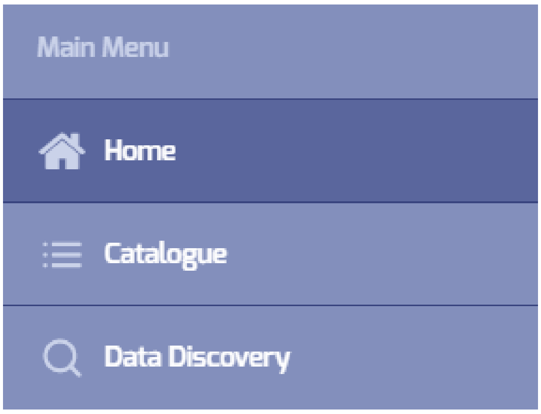


Figure 1. Main functionalities menu

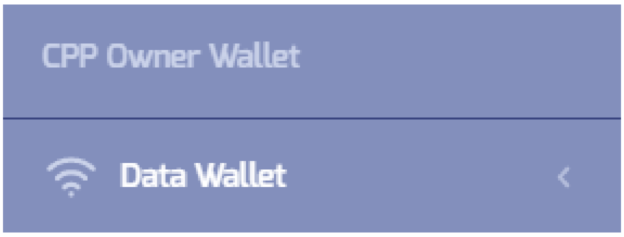


Figure 2. Service Provider Wallet menu

Other user related functions can be found in the top right corner of the toolbar which shows the F.A.Q. and a menu including Marketplace *Profile*, User *Settings* and *Log out* (Figure 3).

The CPP shown in the toolbar is the one for which all data retrieved will belong to. In case a CPP owner has more than one CPP registered it can be changed through the toolbar or in the User Profile.

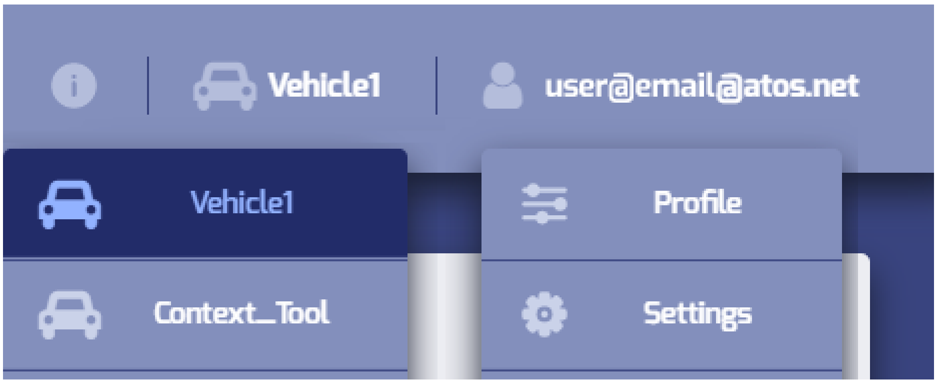


Figure 3. User menu

1. How does Cross-CPP Marketplace work?

1.1. Basic knowledge

Cross-CPP Marketplace works under the premise of letting Service Providers decide which data they want to receive and subscribe to, while giving CPP Owners control over their data at all times.

This leads to the following premises:

- Service Providers request specific data through a set of filters given, creating Data Requests
- CPP Owners decide which Data Requests they want to accept based on the filters decided by the Service Provider
- Any number of CPP Owners can accept any number of Data Requests

Data Requests are further explained in section 3.1 What is a Data Request?

1.2. Step by step

In a nutshell, these are the steps on how to work within Cross-CPP Marketplace and the possible interactions:

1. User creation (see section 1.4)
    a. CPP Owner has a cloud storage provided by the OEM, or has to create a user on Cross-CPP Cloud Storage (section 1.3)
    b. CPP Owner registration at the Marketplace (section 1.4)
        i. CPP Owner provides Cloud Storage identification and Cloud Vault Id in which his/her data is stored, along with the access key to read it.
2. Discovering data to know which data is available in the system. (see section 2.2)
3. Sharing data (section 3)

a. List available Data Request
b. Open Data Request details
c. Accept Data Request
d. Active contracts can be terminated at any time
e. Consult shared data

1.3. Registration at Cloud Storage

The first thing the CPP Owner need is a place to store the data generated by his/her CPPs.

In case the Data Provider (OEM) doesn't provide that storage Cross-CPP offers its own Cloud Storage.

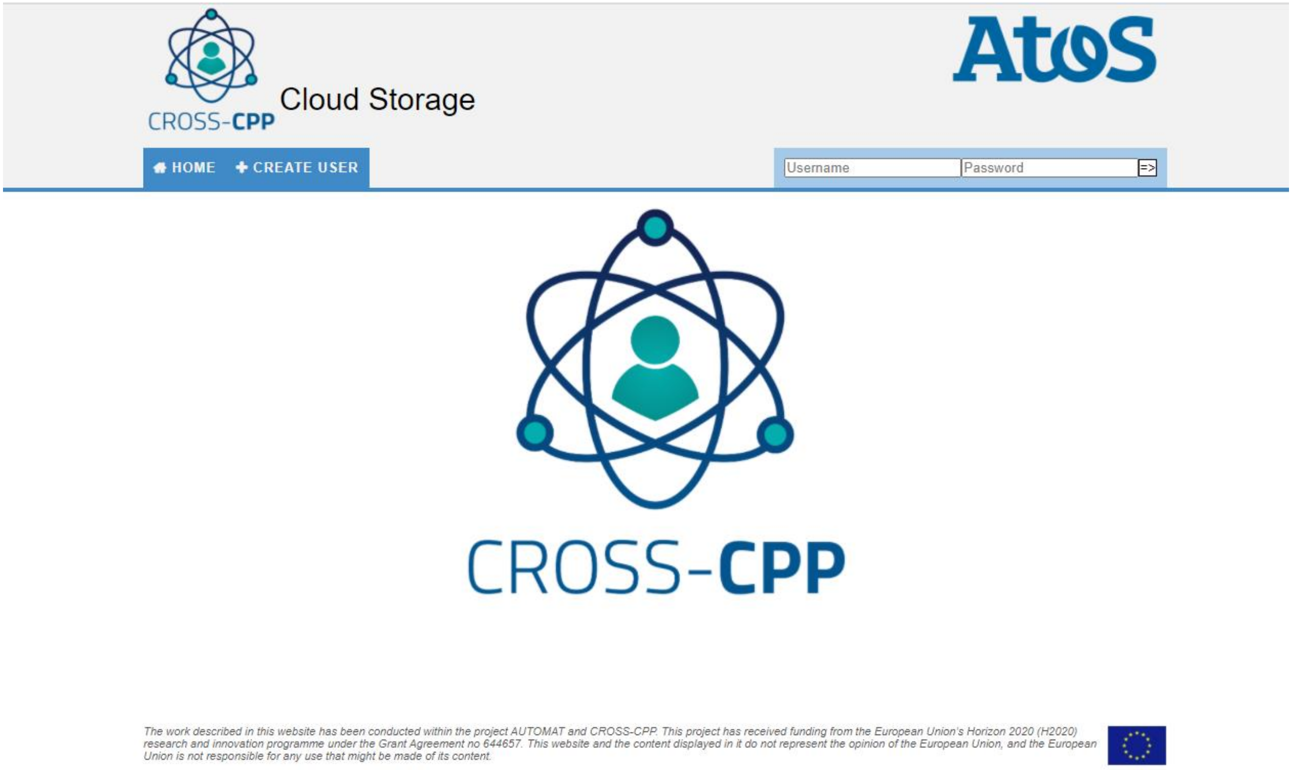
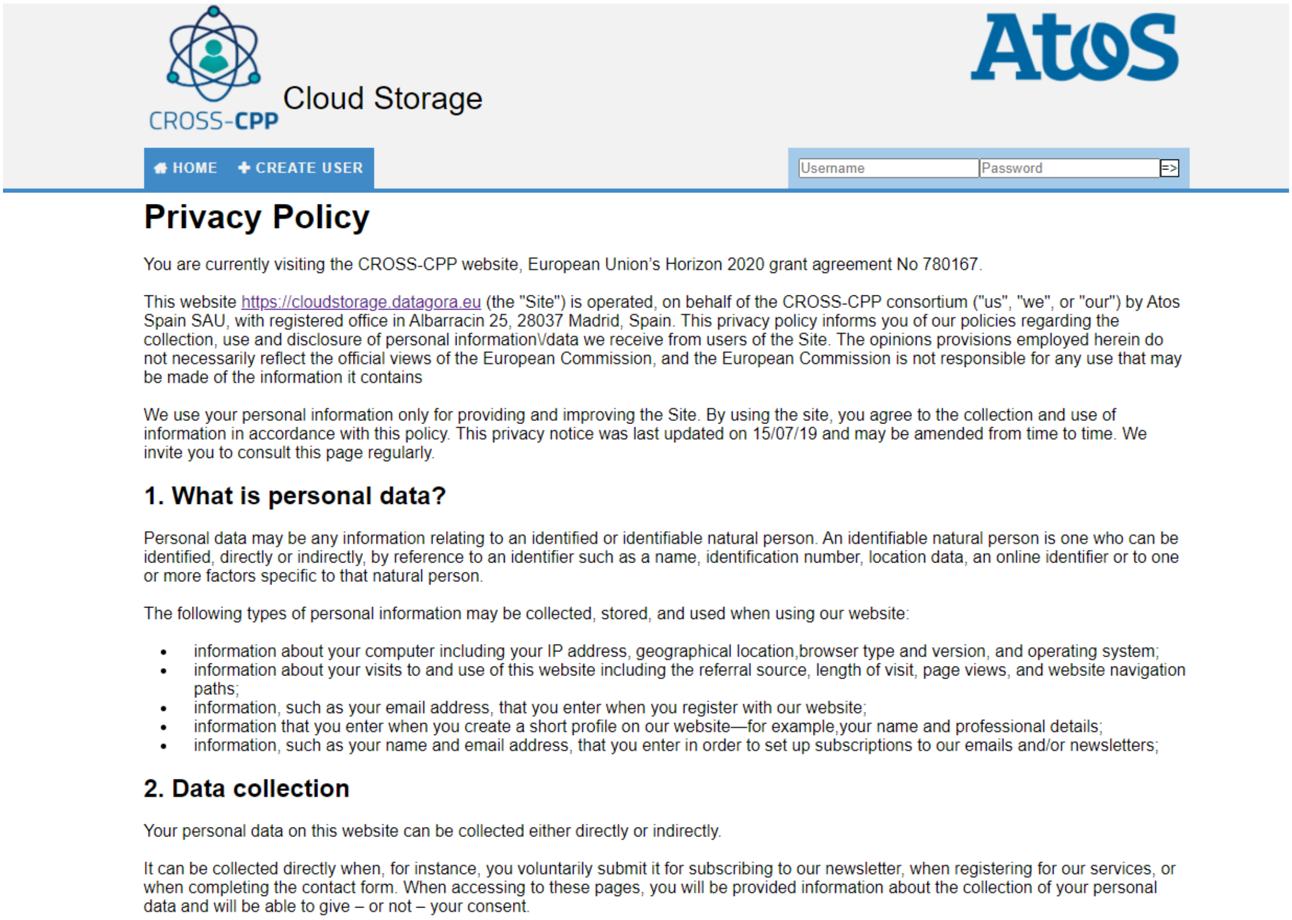


Figure 4. Cross-CPP Cloud Storage - Landing page

After clicking on the "+ CREATE USER" button the Privacy Policy is displayed. Once read, at the bottom of the page, the user registration form can be found.



Privacy Policy

You are currently visiting the CROSS-CPP website, European Union's Horizon 2020 grant agreement No 780167.

This website https://cloudstorage.datagora.eu (the "Site") is operated, on behalf of the CROSS-CPP consortium ("us", "we", or "our") by Atos Spain SAU, with registered office in Albarracin 25, 28037 Madrid, Spain. This privacy policy informs you of our policies regarding the collection, use and disclosure of personal information\/data we receive from users of the Site. The opinions provisions employed herein do not necessarily reflect the official views of the European Commission, and the European Commission is not responsible for any use that may be made of the information it contains

We use your personal information only for providing and improving the Site. By using the site, you agree to the collection and use of information in accordance with this policy. This privacy notice was last updated on 15/07/19 and may be amended from time to time. We invite you to consult this page regularly.

1. What is personal data?

Personal data may be any information relating to an identified or identifiable natural person. An identifiable natural person is one who can be identified, directly or indirectly, by reference to an identifier such as a name, identification number, location data, an online identifier or to one or more factors specific to that natural person.

The following types of personal information may be collected, stored, and used when using our website:

- information about your computer including your IP address, geographical location,browser type and version, and operating system;
- information about your visits to and use of this website including the referral source, length of visit, page views, and website navigation paths;
- information, such as your email address, that you enter when you register with our website;
- information that you enter when you create a short profile on our website—for example,your name and professional details;
- information, such as your name and email address, that you enter in order to set up subscriptions to our emails and/or newsletters;

2. Data collection

Your personal data on this website can be collected either directly or indirectly.

It can be collected directly when, for instance, you voluntarily submit it for subscribing to our newsletter, when registering for our services, or when completing the contact form. When accessing to these pages, you will be provided information about the collection of your personal data and will be able to give – or not – your consent.

Figure 5. Cross-CPP Cloud Storage - Privacy Policy

12. Contact Us

If you have any questions about this Privacy Policy, please contact us: support@datagora.eu.

Create User

Full Name:

Login Name:

Password:

Repeat Password:

I agree with the privacy policy

Create User

The work described in this website has been conducted within the project AUTOMAT and CROSS-CPP. This project has received funding from the European Union's Horizon 2020 (H2020) research and innovation programme under the Grant Agreement no 644657. This website and the content displayed in it do not represent the opinion of the European Union, and the European Union is not responsible for any use that might be made of its content.

Figure 6. Cross-CPP Cloud Storage – Create user form

If created correctly, user will be notified and automatically logged in.

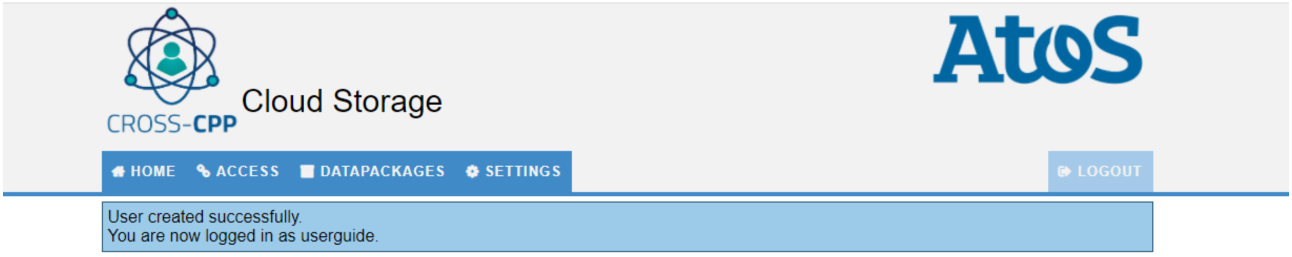


Figure 7. Cross-CPP Cloud Storage – Create user confirmation

In the "ACCESS" page the user can find the information related to his/her created data Vault. Here can be found the following information:

- **Vault ID** (red rectangle): identifier of the vault. Need to be handed to the Marketplace upon registration and to the OEMs to be able to push data into the vault.
- **Write Access Key** (yellow rectangle): token that has to be handed to the OEM for them to manually ask access to write in the vault.
- **Read Access Key** (green rectangle): token that has to be handed to the Marketplace and its components to be able to read the data stored in the vault.

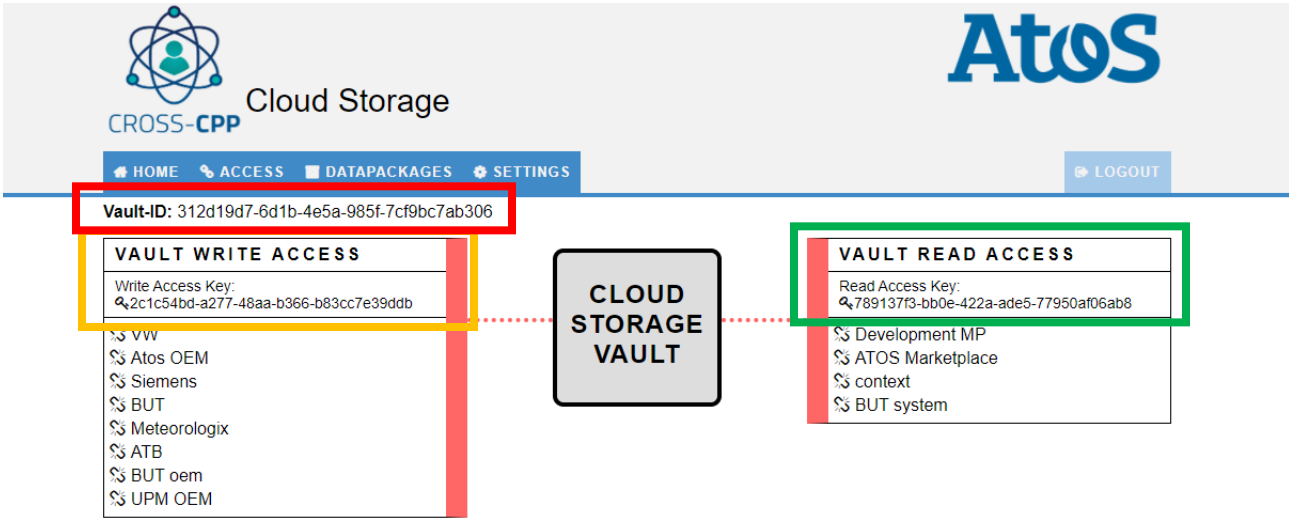


Figure 8. Cross-CPP Cloud Storage – Vault information

As the vault owner you can, and it is recommended, use the tool to manually give read or write permission to the different actors by simply clicking on them.

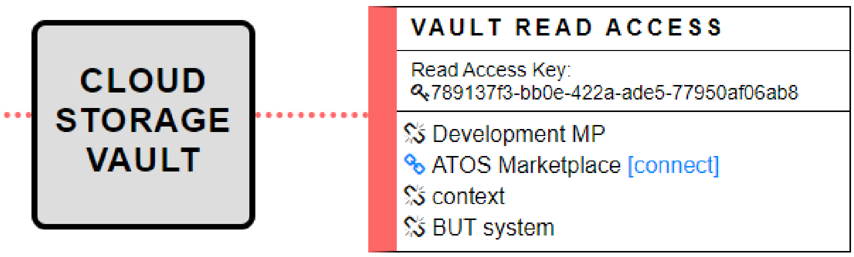


Figure 9. Cross-CPP Cloud Storage – Give access

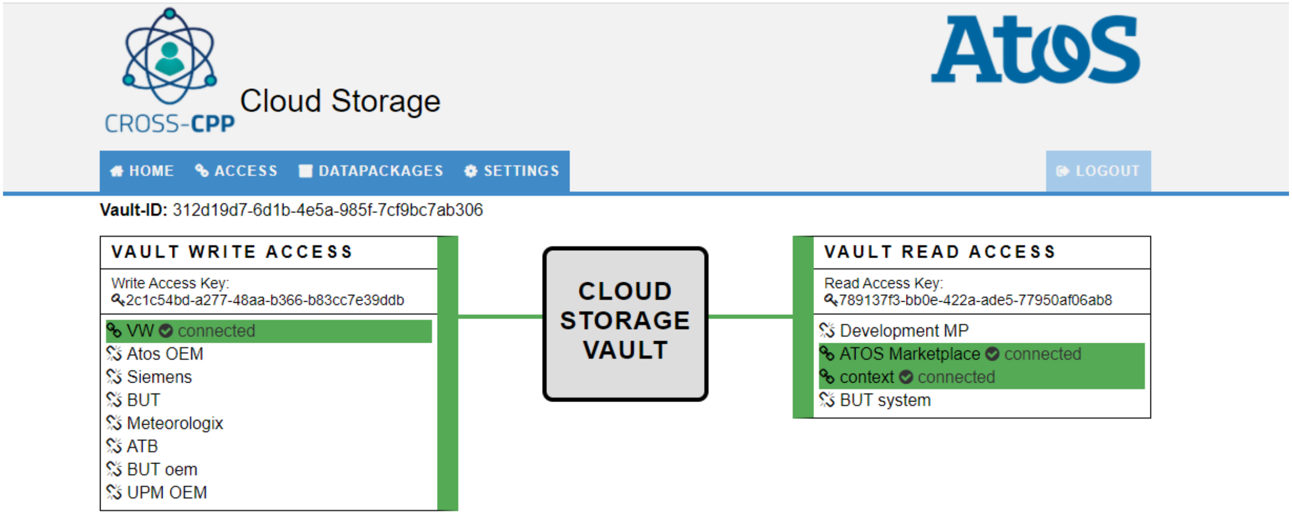


Figure 10. Cross-CPP Cloud Storage – Vault status

The user can always revoke the given permissions by clicking on the same actor and confirming the decision.

1.4. Registration at Marketplace

In order to register into the Marketplace, CPP Owners need to create an account. In the Sign up! button to create a CPP Owner account. After filling the sign-up form, the user receives the credentials for log in into the Marketplace.

As seen in 1.3 Registration at Cloud Storage above the necessary information can be found in the Cloud Storage Vault page.

Once the form is filled the user must accept the Privacy Policy.

Figure 11. Cross-CPP Marketplace sign up form

Once the form is filled the user must accept the Privacy Policy.

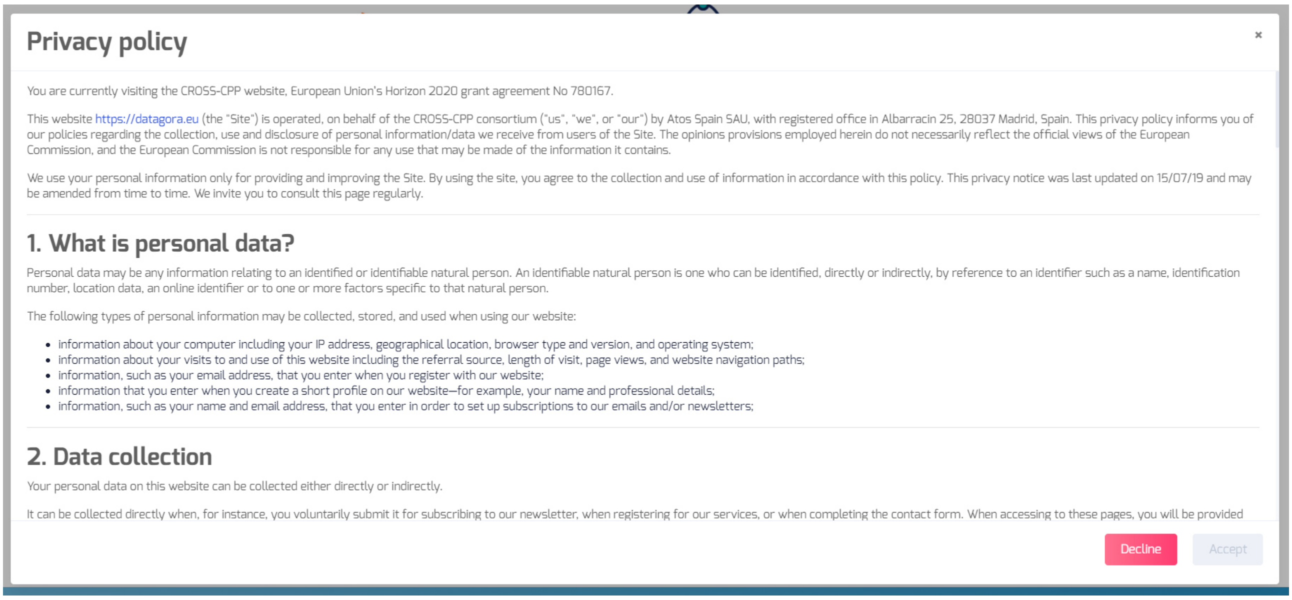


Figure 12. Cross-CPP Marketplace Privacy Policy

Once accepted and the account is successfully created a confirmation email will be sent to the provided email with the steps to activate the account.

Dear Device Owner,

Your Data owner account has been created.

Device: my vehicle

Password: a1m9XDVm63.

Now you can login in the AGORA Data Marketplace,
with your email: user@owner.com

Best Regards,
Agora team

Figure 13. User creation confirmation email

Once activated the user can log in. Clicking in the "Sign in" button will lead to the Identity Manager sign in page.

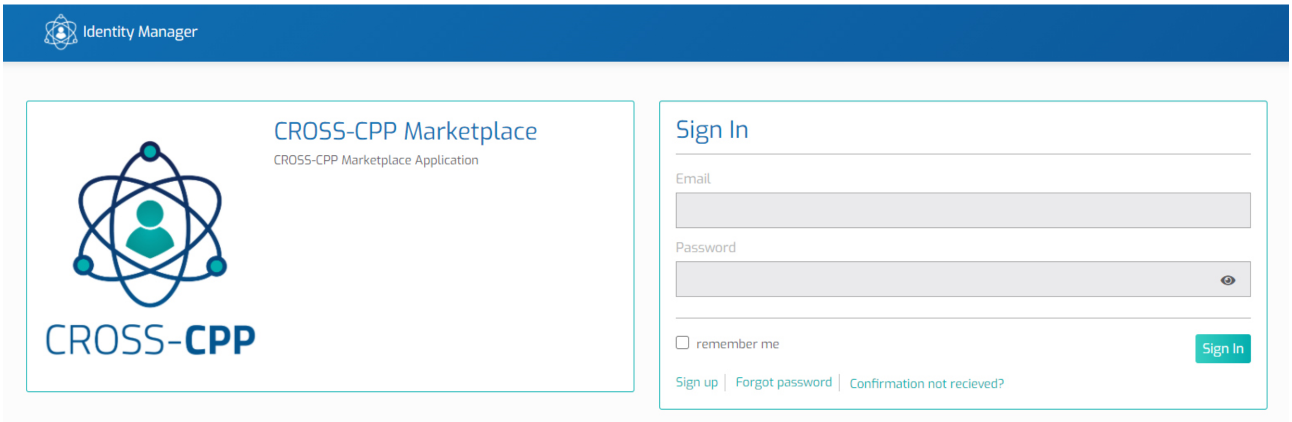


Figure 14. Identity Manager - Sign In page

In the case the user has not received any confirmation email, he/she can request it again by clicking on "Confirmation not received?" and providing the account email.

A CPP owner can have more than one CPP registered to the same account. In this case just provide the same email of the previously registered account and the CPP/device will be added and ready to be used on his/her next session.

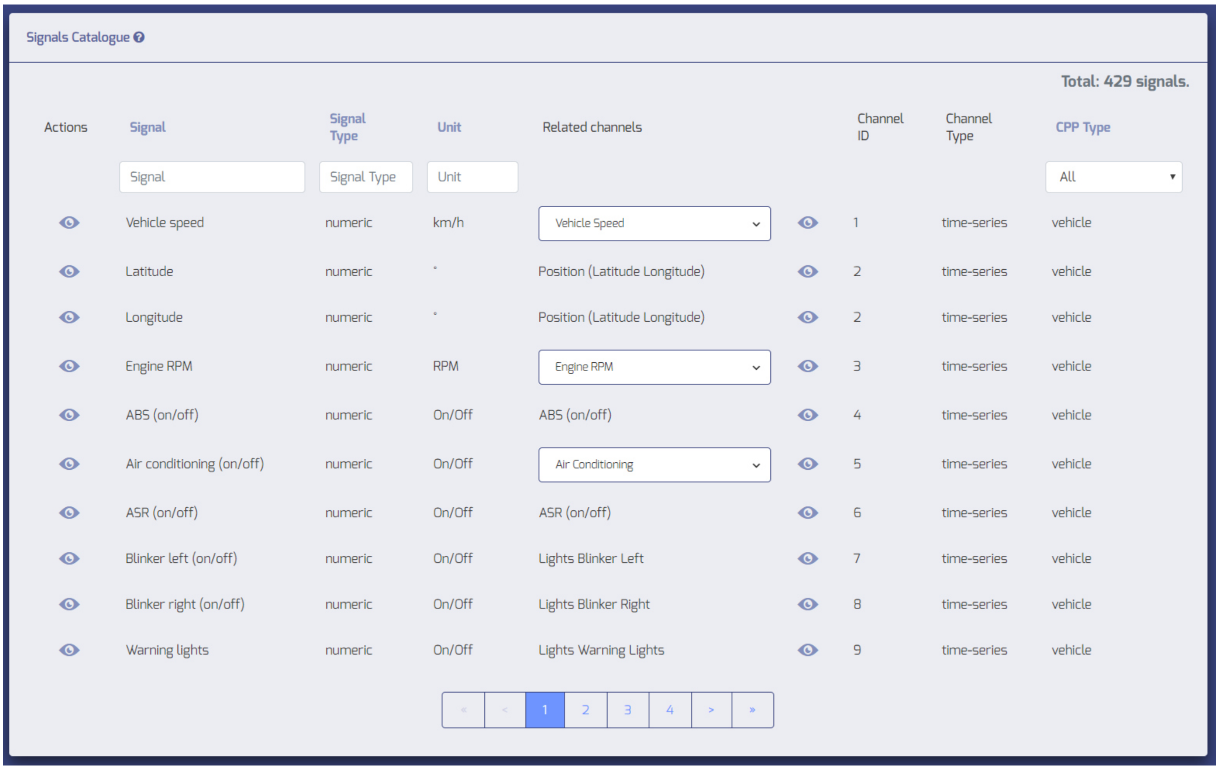
2. Main Marketplace Functionalities

2.1. Catalogue

The catalogue (see Figure 15) shows a list of all available signals under a harmonized data model called Common Industrial Data Model (from now on CIDM – see F.A.Q in section 8) which enables to retrieve data requests in our services. The data in the Cloud Storage is stored following the CIDM which harmonizes the proprietary OEM formats into brand independent data formats (made then available in the catalogue of the marketplace).

There a CPP owner can search and filter by different attributes and see the channels that use each signal. Below, the filters which appears in Signals Catalogue are explained (See Figure 16):

- **Signal**: the name of the signal describing what is measured.
- **Signal type**: type of the signal data. Following the CIDM this can be “numeric”, “enumeration”, “information” or “general-purpose”.
- **Unit**: units of the data
- **Related Channels**: data channels that uses the data received from this signal. In case there is more than one that use the signal, a list is available to select which one is of interest.
- **Channel ID**: identification of the selected channel that receives data from the signal
- **Channel Type**: selected channel type of data receiving. Following the CIDM this can be “time-series”, “histogram”, “geo-histogram” or “general-purpose”.
- **CPP Type**: from which type of CPP this signal receives data from. At the moment this can be from vehicles or buildings.



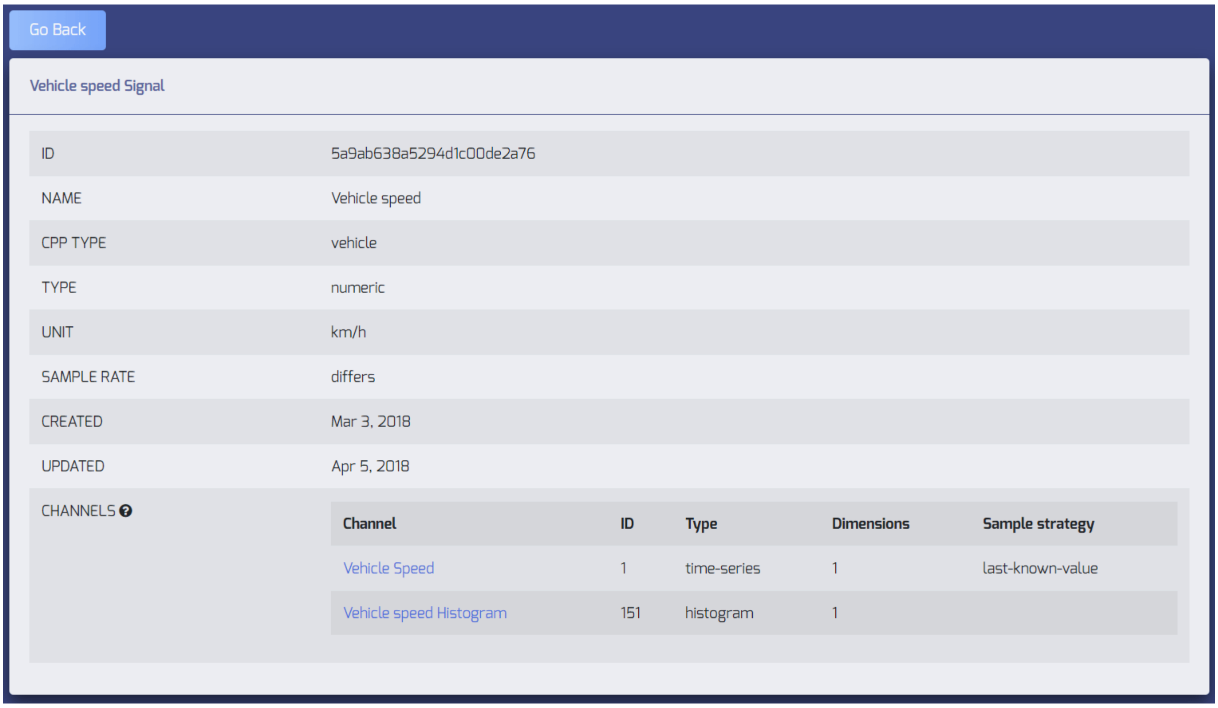
Signals Catalogue

Total: 429 signals.

Actions	Signal	Signal Type	Unit	Related channels		Channel ID	Channel Type	CPP Type
	Signal	Signal Type	Unit					All
	Vehicle speed	numeric	km/h	Vehicle Speed		1	time-series	vehicle
	Latitude	numeric	°	Position (Latitude Longitude)		2	time-series	vehicle
	Longitude	numeric	°	Position (Latitude Longitude)		2	time-series	vehicle
	Engine RPM	numeric	RPM	Engine RPM		3	time-series	vehicle
	ABS (on/off)	numeric	On/Off	ABS (on/off)		4	time-series	vehicle
	Air conditioning (on/off)	numeric	On/Off	Air Conditioning		5	time-series	vehicle
	ASR (on/off)	numeric	On/Off	ASR (on/off)		6	time-series	vehicle
	Blinker left (on/off)	numeric	On/Off	Lights Blinker Left		7	time-series	vehicle
	Blinker right (on/off)	numeric	On/Off	Lights Blinker Right		8	time-series	vehicle
	Warning lights	numeric	On/Off	Lights Warning Lights		9	time-series	vehicle

Figure 15. Signals catalogue

Also, each signal and channel have its own detailed view with all available information, including direct links to each other.



ID	5a9ab638a5294d1c00de2a76
NAME	Vehicle speed
CPP TYPE	vehicle
TYPE	numeric
UNIT	km/h
SAMPLE RATE	differs
CREATED	Mar 3, 2018
UPDATED	Apr 5, 2018
CHANNELS	

Channel	ID	Type	Dimensions	Sample strategy
Vehicle Speed	1	time-series	1	last-known-value
Vehicle speed Histogram	151	histogram	1	

Figure 16. Signal details

2.2. Data Discovery

The Data Discovery offers CPP owners a tool to browse the data from his/her CPPs that is currently available in the Cross-CPP Marketplace. A wide variety of filters are provided in order to help CPP owners to narrow the results.

In a first step, the CPP owner has to select the signals for which data is interested in. If none is selected, a search for all channels will be requested. Once a signal is selected the channel that uses that signal will be added to the selection list. Selected channels can be removed from the list at any time.

If one or more channels are selected suggestions can be requested. This is possible thanks to the implementation of the Context Monitoring and Extraction module (CME) (check additional information in Context Sensitive Security in section 4.1) within the Data Discovery process. In case suggestions are requested, the CME will provide a list of channels related to those already selected[1]. Channels of interest can be added directly from the suggestion list. Also, suggestions can be requested again each time a new channel is added.

Once the channels are selected, the user may apply several filters to the data packages (See Figure 17):

- **Date and duration** section filter the results by data package submission date, measurement recording date and measurement time duration.
- The **geographical selection** offers a way to narrow results in some geographical area. This can be set selecting a country in the list, or drawing a customized area entering the area bounds coordinates.

[1] Suggestions are provided based on a context model that is the result of analysis of the physical relation between signals (e.g. in case temperature is a selected signal, the CME would suggest the humidity and sun intensity).

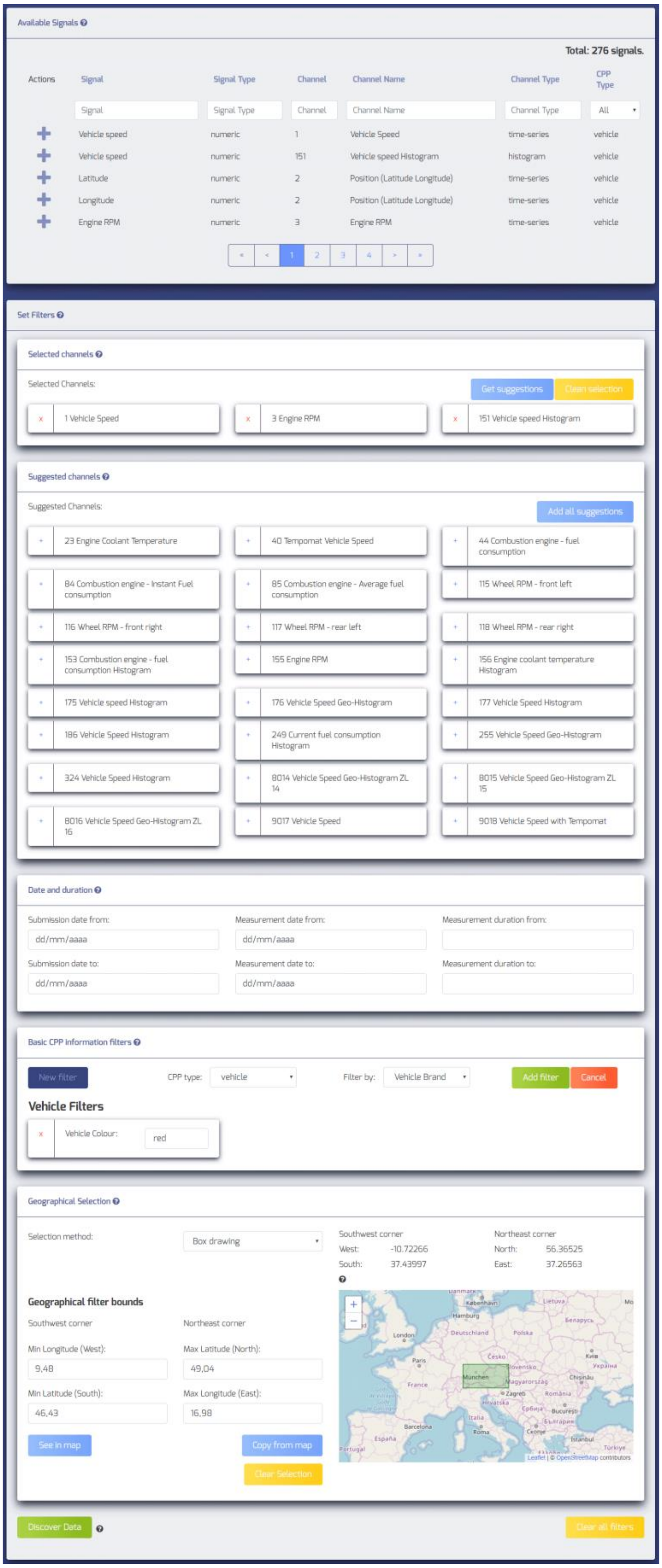


Figure 17. Data Discovery filters

Once the filters are set, the CPP Owner can see the data from his/her CPPs available in the Cross-CPP Marketplace (See Figure 18). Only metadata is used in this process and no real data can be shown or retrieved through this functionality:

- A box with some **general statistics** shows the total of CPP owners and the number of selected channels that contains any data, and the number of data packages retrieved in the discovery.
- For each type of CPP selected with data a box is displayed showing the number of entities and channels with data, and the minimum, maximum and average duration of its measurements.
- The **heatmap** offers and overview of the geographical areas that provides data, in case it is desired to narrow the discovery to a specific area.
- The **pie charts** show how the data is distributed across the different channels. Sectors can be hidden if needed.
- The **line and bar charts** display the amount of data packages retrieved depending on its submission date and time duration.

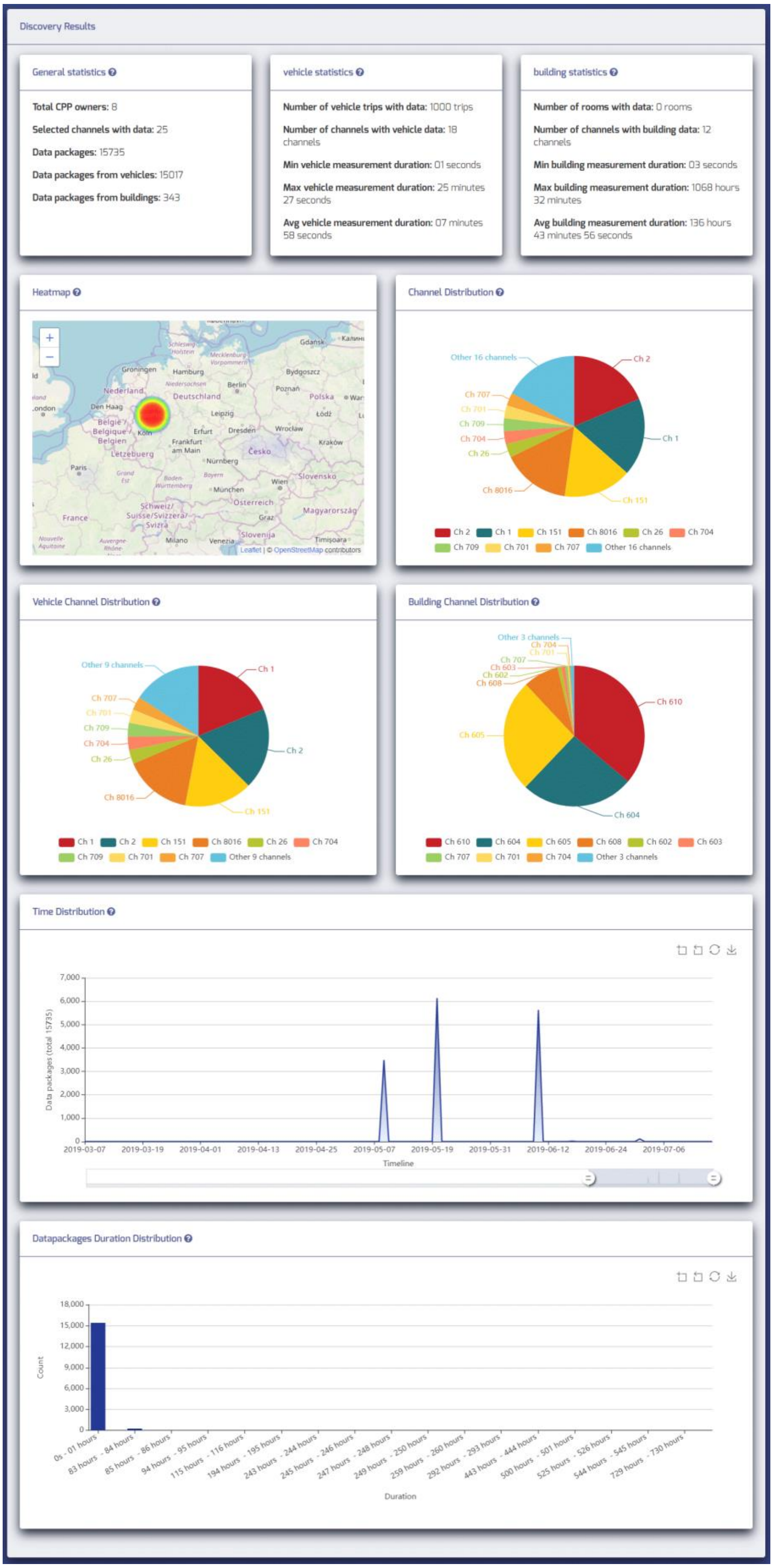


Figure 18. Data Discovery results

3. My Data Wallet

The CPP owner Wallet (from now on Data Wallet) (see Figure 19) groups different views for data review in which the user can:

- List available data requests
- Review each data request details
- Accept a data request
- List accepted data requests
- Review each data request acceptance details
- Terminate data request acceptance
- See the data packages sent through each data request
- Get a summary of each data request package transactions

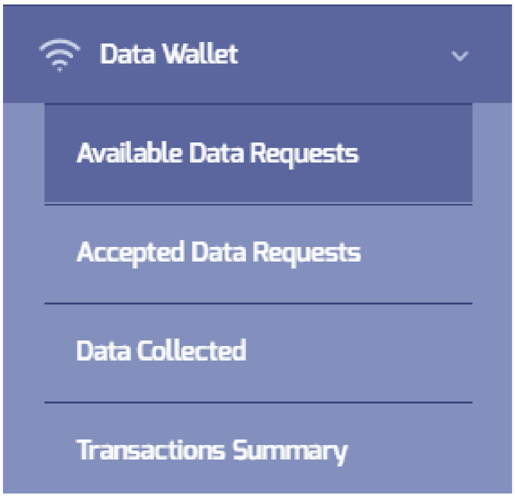


Figure 19. Data Wallet menu

3.1. What is a Data Request?

Data Request stands for a configuration set by a group of filters in order to receive concrete data within a desired scope. This scope is set by Service Providers themselves.

Once the Data Request is published by Service Providers, CPP owners can accept and get the consent to deliver their data through that request. Service Providers will receive all data from CPP owners that accepted that request and that is exclusively for the scope of the Data Request published.

3.2. How to accept a Data Request?

In the Available Data Requests view a table with published Data Request is shown (Figure 20). Selecting one will lead to a detailed page in which the full configuration and intended use of the data received is displayed (Figure 21).

There a CPP owner can decide to accept that Data Request.

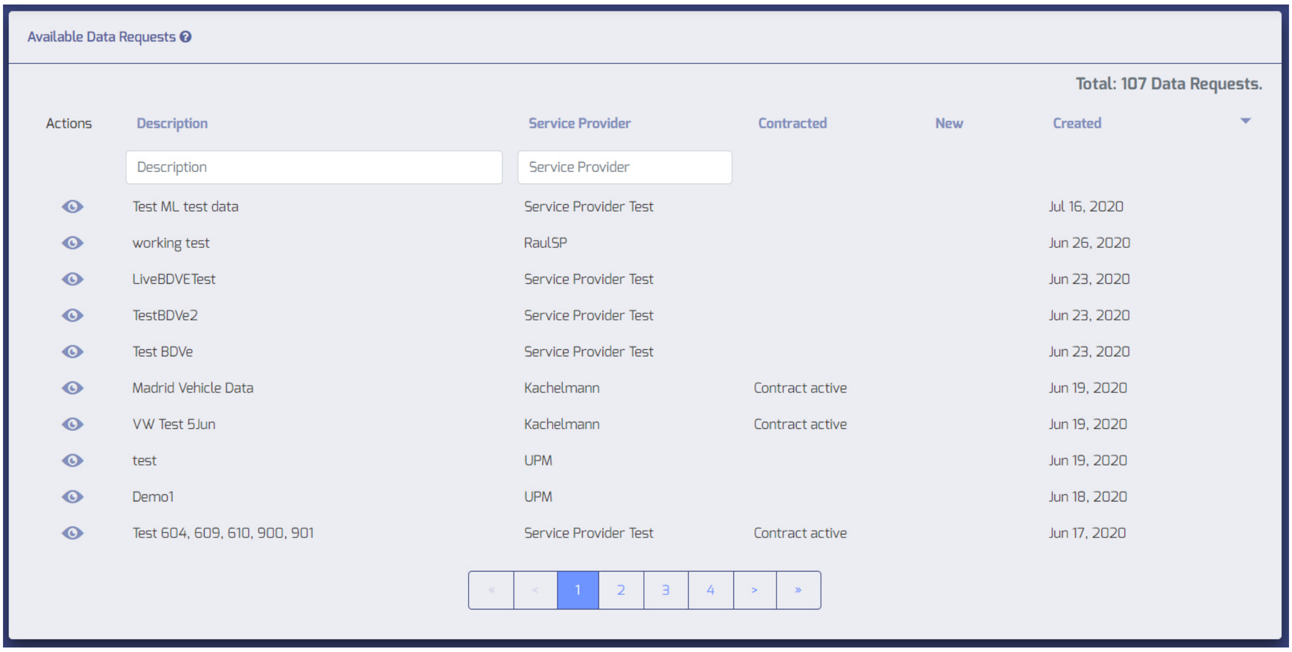


Figure 20. Available offers

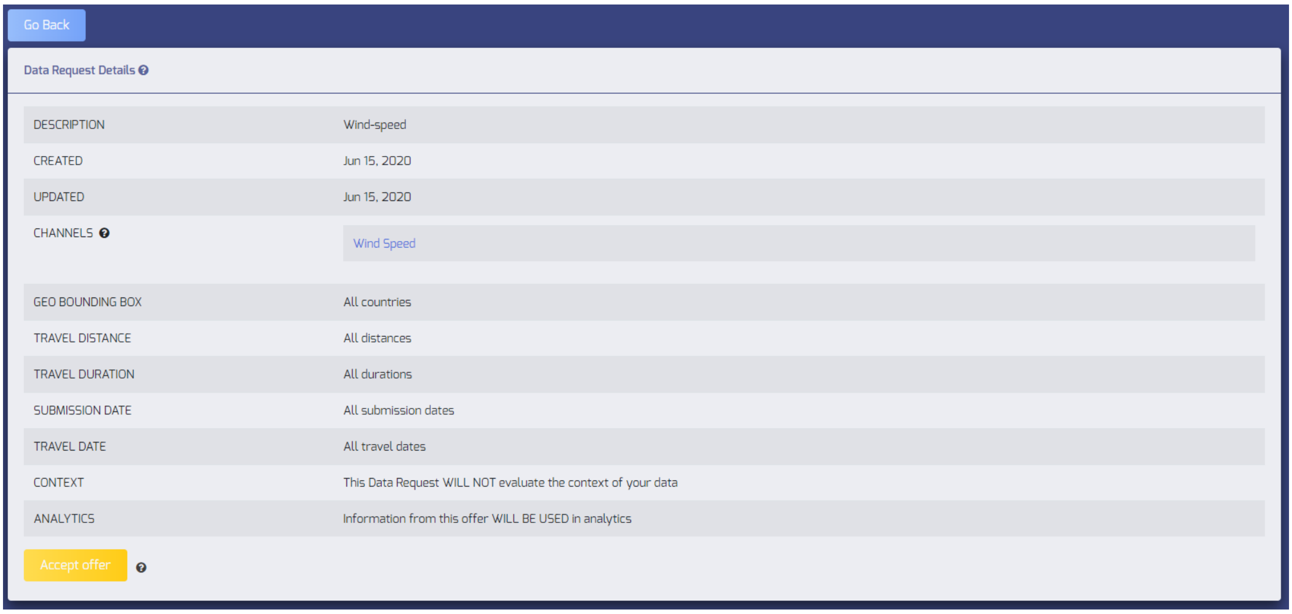


Figure 21. Published Data Request details

3.3. Data Request validity

Once a Data Request is accepted CPP owner's data will be sent to that Service Provider in case the data passes through the data request filters set by the Service Provider when making the Data Request.

This agreement also represents the reference for the data access control of Service Provider by the marketplace (in connection with the Context Sensitive Security module, see section 4) during runtime.

CPP owners can review their accepted offers (also called contracts) in the list displayed (see Figure 22).

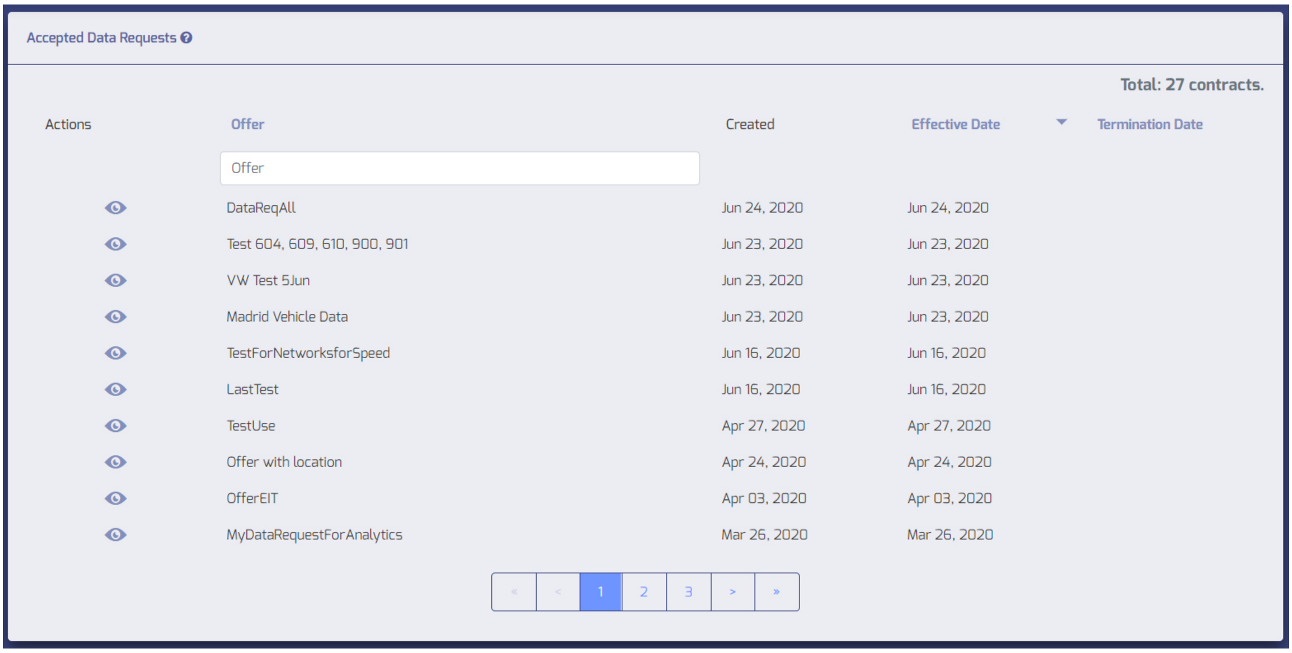


Figure 22. Accepted Data Requests

Data Requests has no expiration time, and his validity would end in one of two scenarios explained in the following sections.

3.3.1. Terminate contract

From the Accepted Data Requests list (Figure 22) the CPP owner can get to the details of each one, which contains a link to the data request itself, and, if the contract is active, the option to terminate it (see Figure 23).

If the CPP owner chooses to terminate a contract, data will no longer be sent to the Service Provider through that data request.

CPP owners can always accept the offer again, generating a new contract.

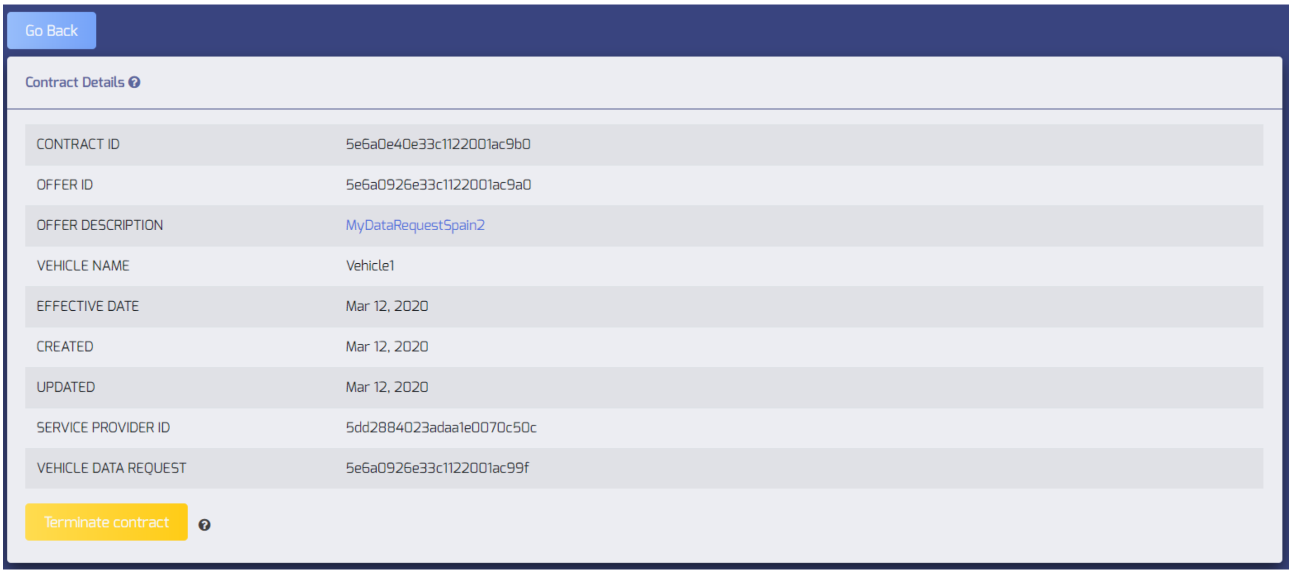


Figure 23. Contract details

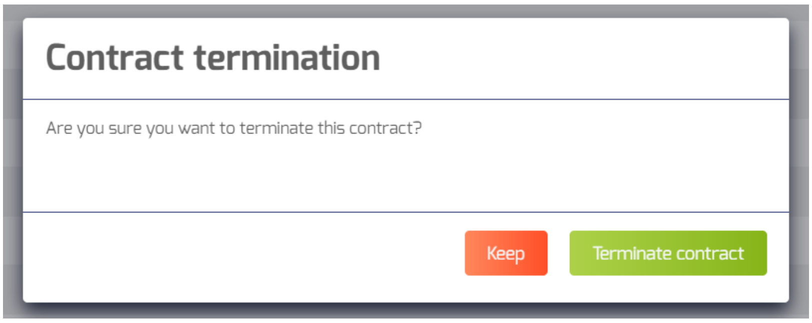


Figure 24. Contract termination

3.3.2. Data Request modification

Service Providers cannot modify their Data Requests. Any change would generate a new Data Request that can be accepted (or not) by all CPP owners.

On the other end, Service Providers can delete their old Data Requests, terminating all active contracts. This will also stop the flow of data from all CPP owners with contracts.

3.4. Data Review

CPP owners has two different views to get a summary of their collected data.

3.4.1. Transactions Summary

Transactions Summary section view displays a summary of data collected from the published Data Requests, including the following information:

- total transactions

- amount of pull type transactions
- amount of push type transactions
- total size of available data
- first package received
- last package received

The details view leads to the Data Requests detail view (Figure 21). Data Transactions view is depicted as follows including the following fields:

- **Data Request**: The name of the Data Request the transactions belongs to
- **Contract transactions**: total number of transactions from users which accepted this data request.
- **Pulls**: number of times the data has been pulled
- **Pushes**: number of times the data has been sent through the AEON channel
- **Total size**: number of bytes between all transactions

Data Transactions

Total: 7 contracts

Actions	Contract ID	Data Request	Contract transactions	Pulls	Pushes	Total size	First	Last
	Contract ID	Data Request						
	5cbdd5134008d31e008b06f5	All	10857	9893	964	398.36 Mb	Apr 26, 2019	Mar 21, 2020
	5cdbe600ff89151c002d172e	trayectories	346	330	16	22.14 Mb	May 15, 2019	Jul 03, 2019
	5cf6709da6bd912300bfa79a	Wiper data VW	1912	1896	16	62.38 Mb	Jun 04, 2019	Mar 11, 2020
	5d02d3c93a2b491e0018761d	x	91	75	16	4.93 Mb	Jun 17, 2019	Mar 21, 2020
	5d1c78c00258331e00ed8839	all for all	20	20	0	1.41 Mb	Jul 03, 2019	Jul 03, 2019
	5d1dacf1f4cd991e00d0a310	all with all includes	478	478	0	31.64 Mb	Jul 04, 2019	Mar 19, 2020
	5e591a3a003fa222001a3f62	Meteologix Smart Building Forecast from Siemens Buildings	10	10	0	20.21 Kb	Mar 03, 2020	Mar 17, 2020

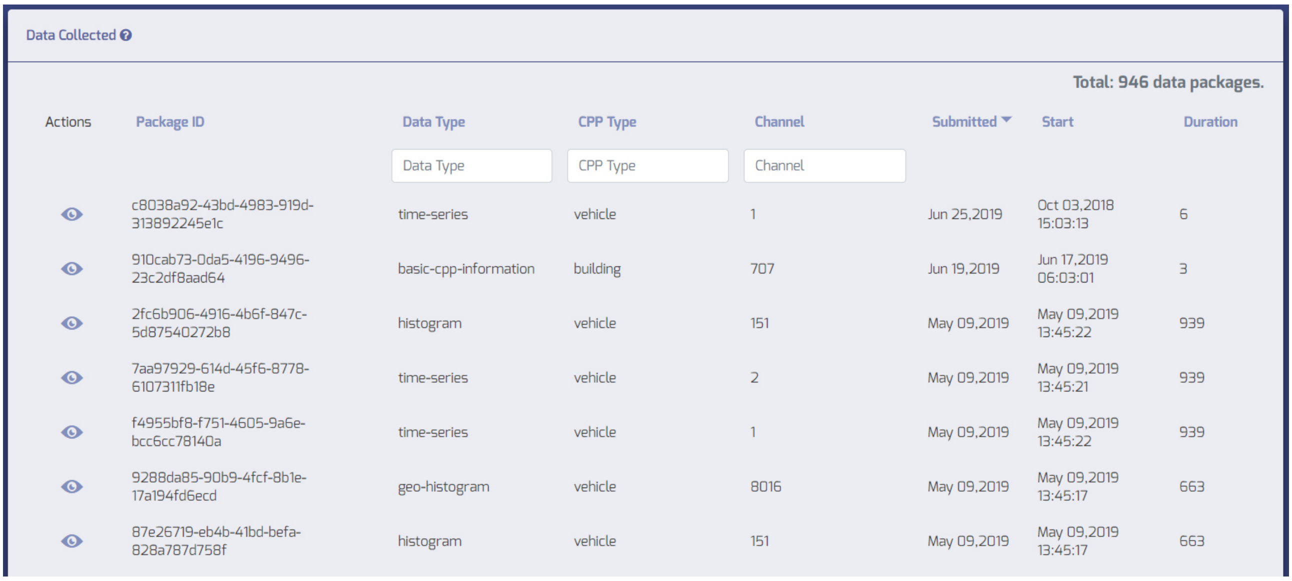
Figure 25. Data Transactions summary list

3.4.2. Data Collected

Data Packages view displays a list of data packages collected from the CPP owner for all the accepted offers.

The list can be filtered by:

- Data type: type of data generated from the data collected between time-series, histogram, geo-histogram or basic-cpp-information
- CPP Type: type of the CPP that generated the data between vehicle and building
- Channel: measurement channel through which the data was collected



Data Collected

Total: 946 data packages.

Actions	Package ID	Data Type	CPP Type	Channel	Submitted	Start	Duration
		Data Type	CPP Type	Channel			
	c8038a92-43bd-4983-919d-313892245e1c	time-series	vehicle	1	Jun 25,2019	Oct 03,2018 15:03:13	6
	910cab73-0da5-4196-9496-23c2df8aad64	basic-cpp-information	building	707	Jun 19,2019	Jun 17,2019 06:03:01	3
	2fc6b906-4916-4b6f-847c-5d87540272b8	histogram	vehicle	151	May 09,2019	May 09,2019 13:45:22	939
	7aa97929-614d-45f6-8778-6107311fb18e	time-series	vehicle	2	May 09,2019	May 09,2019 13:45:21	939
	f4955bf8-f751-4605-9a6e-bcc6cc78140a	time-series	vehicle	1	May 09,2019	May 09,2019 13:45:22	939
	9288da85-90b9-4fcf-8b1e-17a194fd6ecd	geo-histogram	vehicle	8016	May 09,2019	May 09,2019 13:45:17	663
	87e26719-eb4b-41bd-befa-828a787d758f	histogram	vehicle	151	May 09,2019	May 09,2019 13:45:17	663

Figure 26. Data collected list

4. Security and authorization

4.1. Context Sensitive Security

The Cross-CPP project aims to provide a cross-sectorial Ecosystem, where unified quality of data, commercial confidentiality, privacy, IPR and ethical issues may arise in dynamically changing and varied combinations, requiring the services provided to be adapted to the specific needs of the users and of the providers of CPP data streams under the prevailing contextual conditions.

More detailed explanation of the Context Sensitive Security module can be read in the Cross-CPP Ecosystem guide (to be found under www.cross-cpp.eu).

4.2. Configure my context preferences

The user has the possibility to select two possibilities related to the context. The first is to select whether the context can be extracted from its CPP data. In the case that the answer is positive then he/she may select in which way the extracted context may influence the Context Sensitive Security module. This means CPP owners can restrict access to some of their data depending on the context configuration.

In the case that the answer is no, then no Context data will be extracted from the CPP in question and therefore the flexibility advantages offered by the semantic enrichment of the Security policies (see previous section) will not be taken into account.

To configure the access to the context of the data CPP owners must access their Profile view (Figure 27), where they can find a list of their available CPPs registered in the Marketplace (Figure 28). Each CPP has a mark whether it has some kind of context filtering or not.

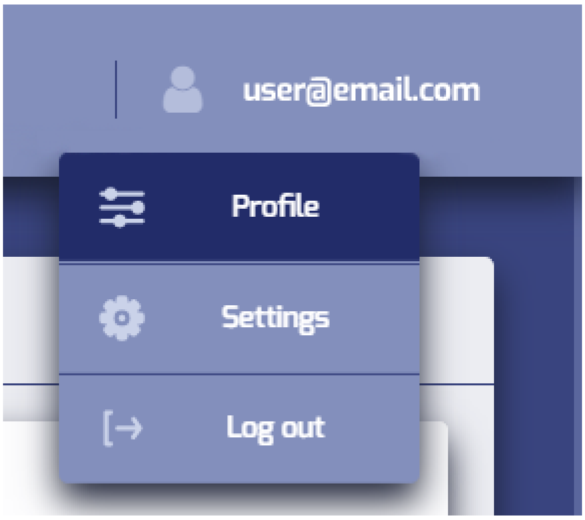


Figure 27. Access user profile

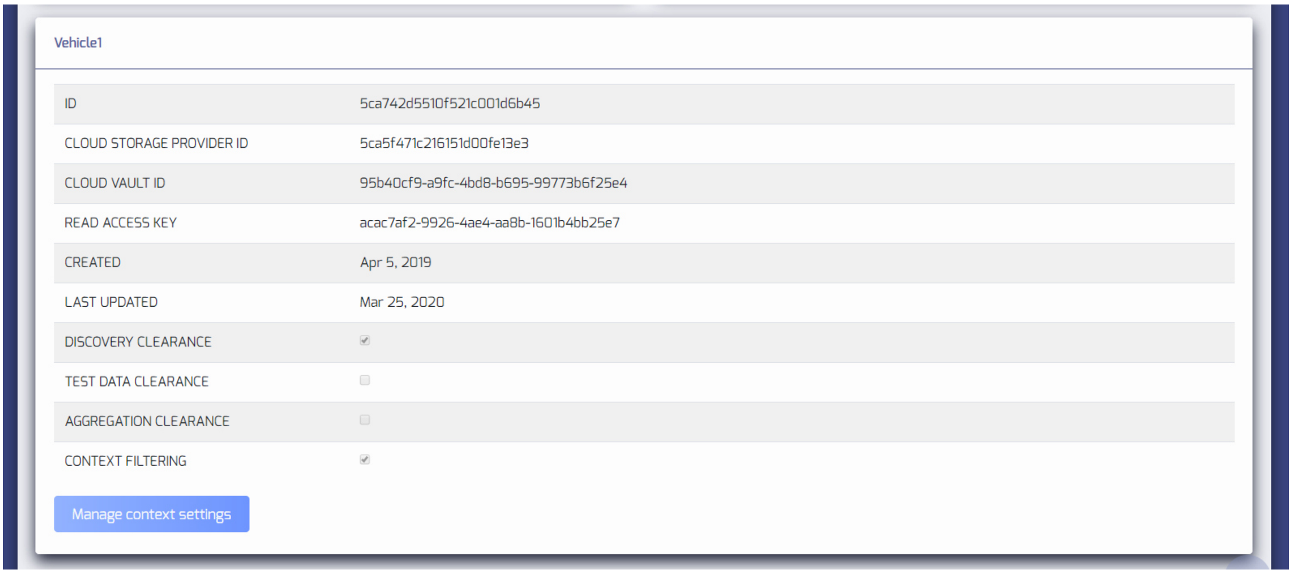


Figure 28. CPP details

Each CPP context can be configured through the form window that opens by clicking the "Manage context settings" button below each one details (see Figure 29).

The form consists on a set of options and alternatives for each option. In case one or more options are selected, then any Data Request for this particular CPP will be filtered, giving Service Providers access only to data fitting selected options. If no option is selected for a type data will not be analysed taking that context into account.

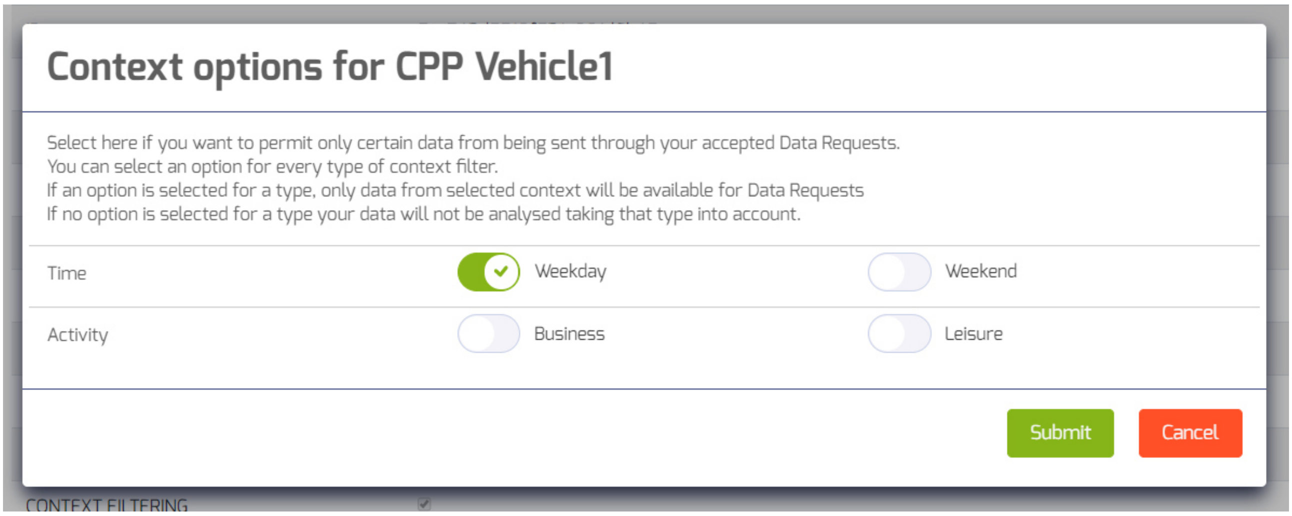


Figure 29. CPP context options

In the screenshot example data will be available for any kind of activity but only if taken on weekdays. Any data taken during weekends will not be available to any Service Provider.

5. Account management

CPP Owners can find their account options through the toolbar menu.

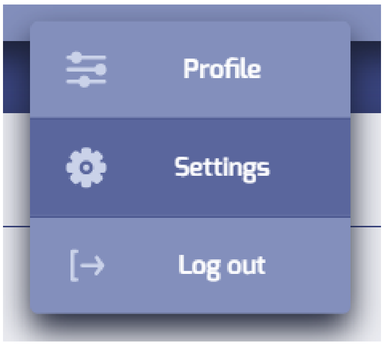


Figure 30. CPP Owner toolbar account menu

5.1. Account profile

On the profile page the CPP Owner can find a list of its connected CPPs and their summary.

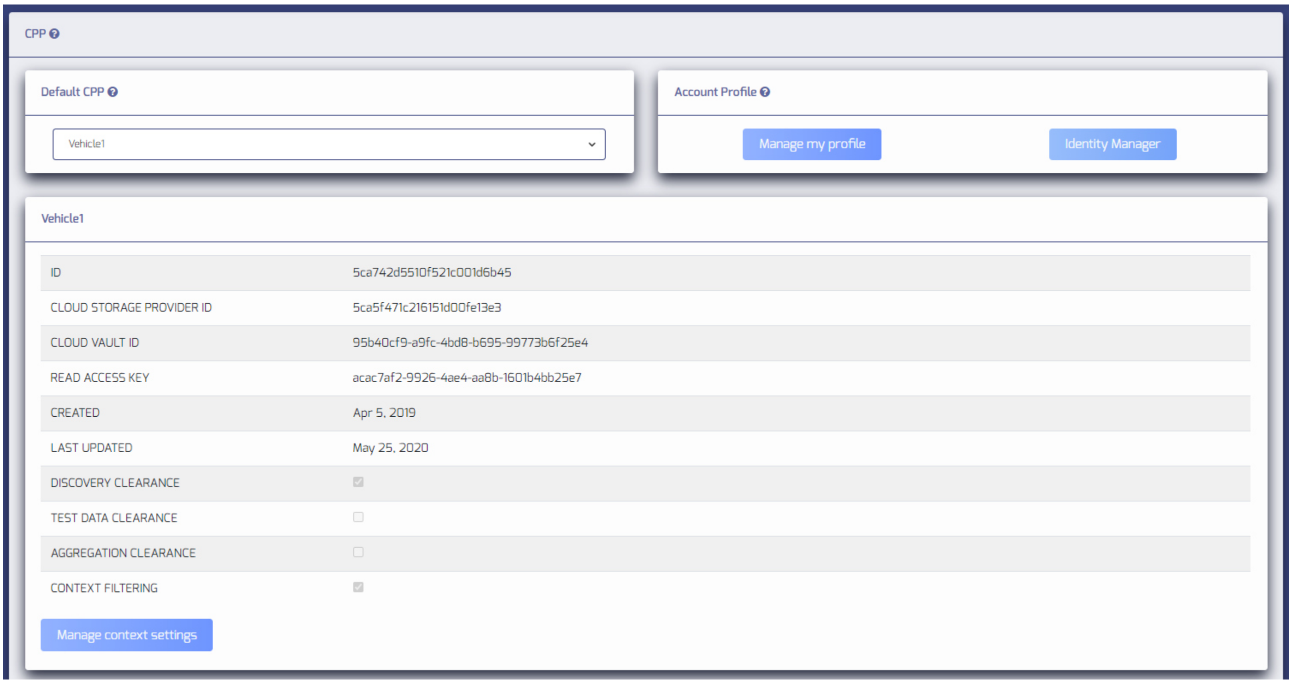


Figure 31. CPP Owner account summary

5.1.1. Working with multiple CPPs

When having multiple CPPs connected to the CPP Owner account the user can only work with one at the same time. CPP Owners can select the CPP that wants to be used to work with through the profile page, or through the dropdown list on the toolbar.

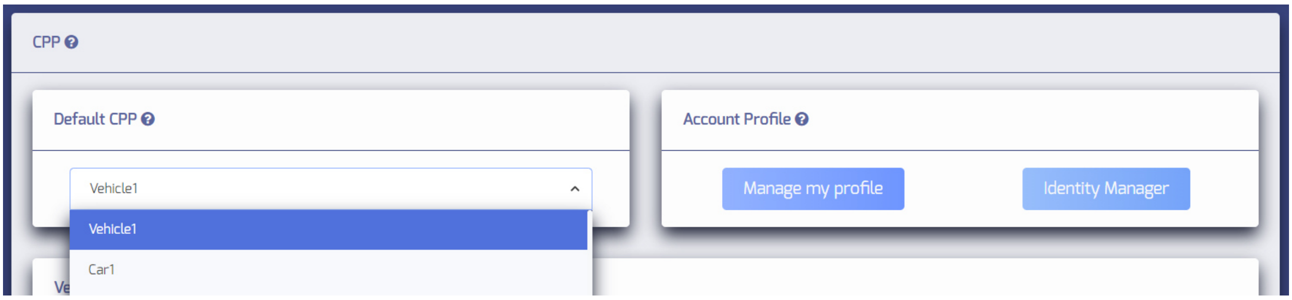


Figure 32. Change CPP on profile page

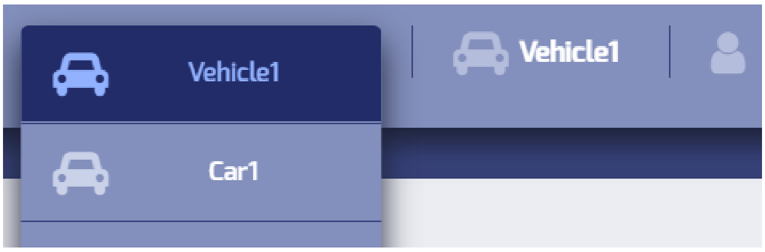
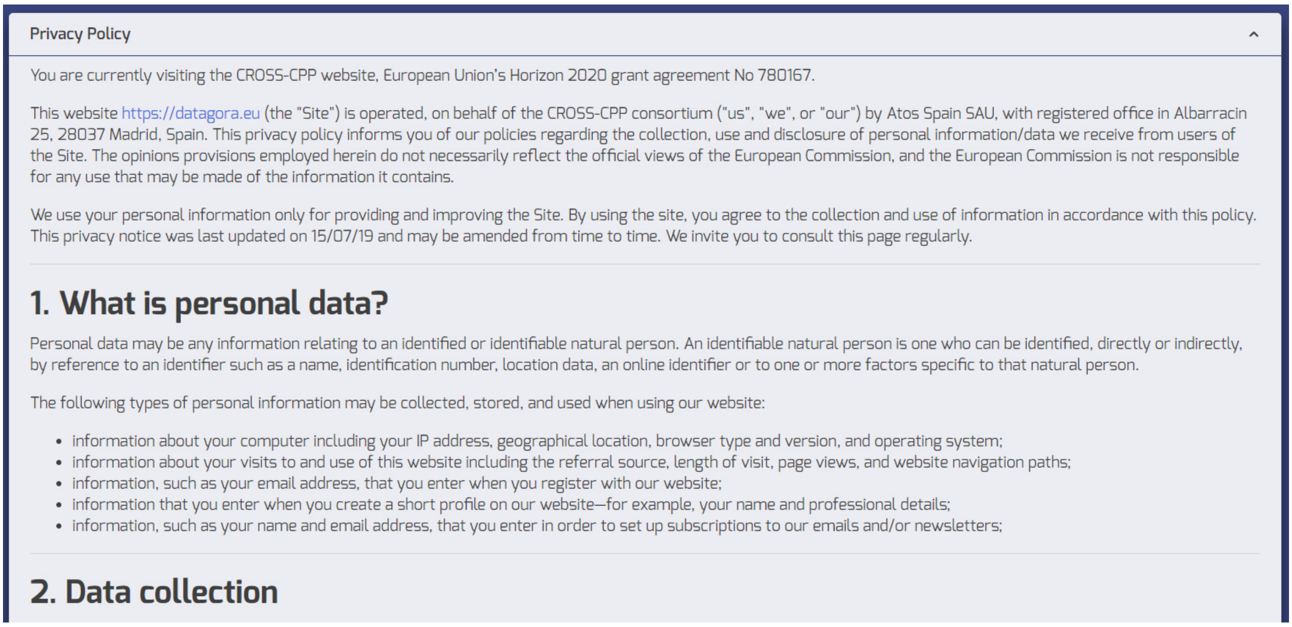


Figure 33. Change CPP on toolbar dropdown list

5.2. Account settings

In the setting page the CCP Owner can find a variety of actions, including reading the last-to-date Privacy Policy

Figure 34. CPP Owner account settings



Privacy Policy

You are currently visiting the CROSS-CPP website, European Union's Horizon 2020 grant agreement No 780167.

This website https://datagora.eu (the "Site") is operated, on behalf of the CROSS-CPP consortium ("us", "we", or "our") by Atos Spain SAU, with registered office in Albarracin 25, 28037 Madrid, Spain. This privacy policy informs you of our policies regarding the collection, use and disclosure of personal information/data we receive from users of the Site. The opinions provisions employed herein do not necessarily reflect the official views of the European Commission, and the European Commission is not responsible for any use that may be made of the information it contains.

We use your personal information only for providing and improving the Site. By using the site, you agree to the collection and use of information in accordance with this policy. This privacy notice was last updated on 15/07/19 and may be amended from time to time. We invite you to consult this page regularly.

1. What is personal data?

Personal data may be any information relating to an identified or identifiable natural person. An identifiable natural person is one who can be identified, directly or indirectly, by reference to an identifier such as a name, identification number, location data, an online identifier or to one or more factors specific to that natural person.

The following types of personal information may be collected, stored, and used when using our website:

- information about your computer including your IP address, geographical location, browser type and version, and operating system;
- information about your visits to and use of this website including the referral source, length of visit, page views, and website navigation paths;
- information, such as your email address, that you enter when you register with our website;
- information that you enter when you create a short profile on our website—for example, your name and professional details;
- information, such as your name and email address, that you enter in order to set up subscriptions to our emails and/or newsletters;

2. Data collection

Figure 35. Privacy Policy

5.2.1. Export account profile

CPP Owners can download a file including the account information and their list and summary of all their connected CPPs.

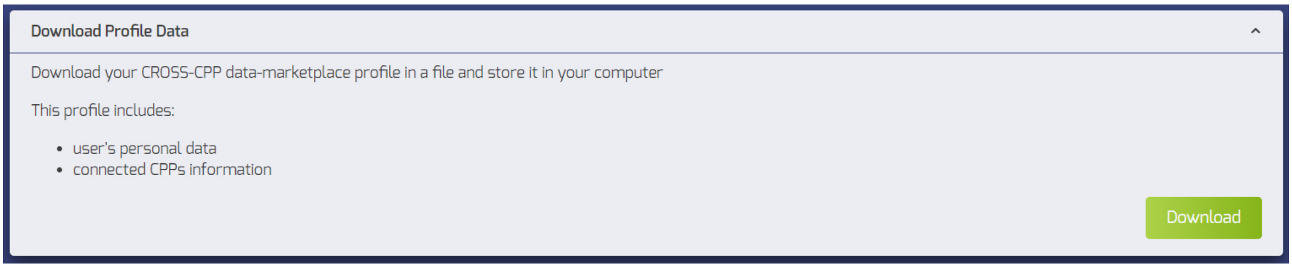


Figure 36. Download Profile Data

5.2.2. Delete account

CPP Owners can delete their account from this page. Following the European GDPR guidelines all related information from the user, including the users' CPPs data and metadata and collected data and its metadata are to be deleted on the process.

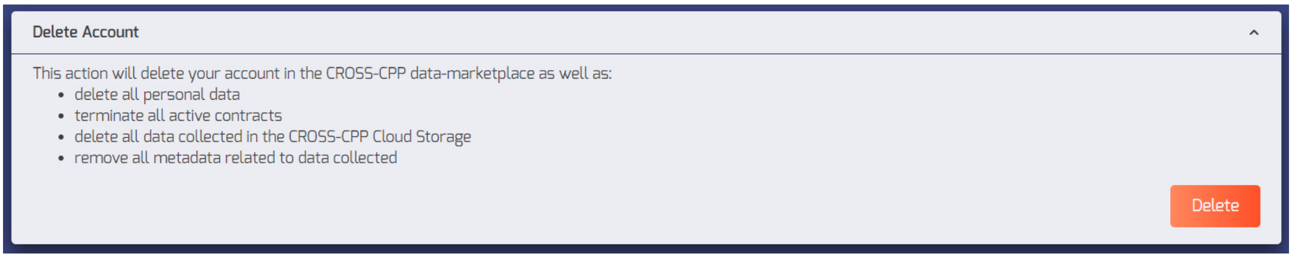


Figure 37. Delete account

F.A.Q.

Cross-CPP data-marketplace

Q: What is Cross-CPP data-marketplace?

A: Cross -CPP data-marketplace connects CPP owners and Service Providers for selling and acquiring Connected Vehicle and Home Building data under the Common Industrial Data model (CIDM). It offers a secure and privacy preserving experience when selling or buying sharing big data, by having the full control over your data shared, to whom and for what purposes.

Cross-CPP offers cross-sectorial Service Providers the possibility to search for more than 200 sensor signals, display advance visualization representations (Histograms, Geo-Histograms, Time Series) and retrieve those datasets in a seamless experience thanks to the open SDK-API created.

Q: How do I, as CPP owner, register into Cross-CPP data-marketplace?

A: When purchasing a device or service that can be connected to the Cross-CPP system your provider will give you the relevant information about the registry process. Once you sign the agreement and consent, you will be registered into the Cross-CPP Cloud Storage, where your data will be stored, and in the Cross-CPP Marketplace, where you can manage it. Both registrations will be confirmed via your provided email.

Q: What do I have to do in order to start working with Cross-CPP data-marketplace?

A: Once registered you must be familiar with the CIDM, as it is the format in which you will receive the data you request. You must also be familiar with AEON, as it is the communication channel used to send the data. You can find information for both in this guide.

Cross-CPP data model

Q: What is the Common Industrial Data Model (CIDM)?

A: The CIDM is a standardized data model for industrial data-driven services. If you are interested you can find extended information about the CIDM in the Service Providers Developers Guide

Q: What is a signal?

A: A signal is the information provider of each CPP. They are the perception organs of CPPs and it is their main duty to detect physical phenomenon and chemical quantities. They observe the environment and generate data in the CIDM format. An example could be “speed” or “latitude”

Q: What is a channel?

A: A channel is the way the physical signals and their sampled measurements are implemented and represented in the CIDM format. Some examples could be “Vehicle Speed” using the signal “Speed” in a time-series or in a histogram format, or “Position” using both “Latitude” and “Longitude” signals.

Q: Can I request a new signal or channel?

A: Cross-CPP data-marketplace offers a wide variety of signals provided by the manufacturers. The catalogue is really extensive and can be filtered in many ways. If even then you can't find the signal that you need and/or think can be provided by any of our company backends, please contact us in: cross-cpp-support@lists.atosresearch.eu.

Cross-CPP marketplace components

Q: What is the Data Discovery component and how does it work?

A: The Data Discovery component is a tool that allows Service Providers to find what data they can access through the marketplace. There they can use the filters provided to narrow or enlarge your results and create Data Request based on the configured search. You can think of it like a test of what they would receive if they publish that request. As a CPP owner you can use that tool to check the available data in the system.

Q: What is the Context Monitoring and Extraction module and how can it help me?

A: The Context Monitoring and Extraction module allows Cross-CPP to suggest signals to add to your current Data Discovery filters, based on the context model of the signals already selected. This might help you find data of interest that you would miss otherwise.

Q: How does the Suggestion component work?

A: The CME component allows Cross-CPP to suggest you signals to add to your current Data Discovery filters, based on the context model of the signals already selected. This might help you find data of interest that you would miss otherwise.

Q: How can the Service Provider use the Context Component in the Data Requests?

A: The Data Request is context sensitive in case the Service Provider selects some of the CME module data filters to restrict the data packages that it wants to receive. For instance, if the Service Provider just wants to receive data that is provided by vehicles that are driving in the highway and this type of context data is extracted then one case say that the Data Request (and data packages that are produced from this request) is context sensitive.

Q: What does it mean the Data Request is context sensitive?

A: The Data Request is context sensitive in case the Service Provider selects some of the CME module data filters to restrict the data packages that it wants to receive. For instance, if the Service Provider just wants to receive data that is provided by vehicles that are driving in the highway and this type of context data is extracted then one case say that the Data Request (and data packages that are produced from this request) is context sensitive.

Q: Which are the consequences of sharing my context?

A: In the case that you share your context, this will allow the use of this data to enhance the functionality that is offered by the marketplace components. Namely, once you allow for context to be extracted, you will have the possibility to allow for the Security module that protects your data to be flexible in the way that your data is provided. As an example, you could allow for your data to be transmitted to the Service Providers requesting your data (making you an offer) by selecting the parameter “on the weekdays” and not the one “on a weekend”. This means that any data produced by your CPP would be given to the Service Providers that have a contract with your just in case that the context extracted “says” that the data was produced on a weekday.

Q: Which are the consequences of not sharing some of my data because of its context?

A: In the case that the Service Provider wants to filter my data by context, this just means that you will not be able to accept that Data Request, as it will not receive any data from you.

Q: How does the Context Sensitive Security protect me and my data?

A: The Context Sensitive Security (CSS) module regulates the contracts made between the CPP owner (you, or better yet, a particular CPP) and the Service Providers that make a Data Request. Upon acceptance of the Data Request, a so-called policy is written that will entail which Service Provider may read which data, from which CPP. During runtime (while the CPP is producing data), the CSS module will allow or deny access to the data based on this policy.

In addition to this, the CSS can function both with taking the extracted context into account or, in the case that the user does not allow this, or that no context could be extracted, without.

Data Requests

Q: What is a Data Request?

A: A Data Request is a set of filters that defines which type of data would you be sharing with the Service Provider that created it. That Service Provider would receive data only from CPP owners that have accepted it.

Q: How do I accept a Data Request?

A: You have to get to the Available Data Requests page to list all published data requests. You can open the details view from each one and, if agree, accept it from there. A contract will be generated to keep the details of the offer acceptance.

Q: How do I review my contracts?

A: The Accepted Data Requests page list all your active and terminated contracts. You can open each one details view to review them.

Q: How do I terminate a contract?

A: In the details view of each contract. This operation cannot be undone.

Q: Can I activate a terminated contract?

A: No, contract termination is definitive. But you can look for the data request and accept it again, generating a new contract.

Q: Can the Service Provider modify the Data Request once accepted?

A: No. The acceptance of a request by a CPP owner implies a consent from its side. Modifying the request would make invalid such consent. Therefore, if a data request is to be modified a new one is generated instead to be accepted again by CPP owners.

Q: Can the data collected be used for other purposes not described in my contracts?

A: No. The acceptance of a request by a CPP owner implies a consent from his/her side. That user allows certain usage of the data given and only for the purposes described in the request.

Glossary

Administrator: Cross-CPP marketplace system administrator

AEON: AEON application

AEON application: publication/subscription based communication application

AEON channel: set configuration for communication between two actors through AEON application

Analytics Toolbox: set of available analytics functions to be requested by the Service Provider

CIDM: Common Industrial Data Model

CIDM model: standardized data model for industrial data-driven services

CME: Context Monitoring and Extraction

Company Backend: system of an OEM that provides its users data to the Cross-CPP marketplace

Contract: entity that summarises the acceptance of a data request from a CPP owner

CPP: cyber-physical product

CPP Data: data created by a CCP and sent to the system by the Company Backend

CPP owner: CPP owner which CPP is registered in the Cross-CPP data-marketplace

Cross-CPP: System

CSS: Context Sensitive Security

Data Request: set of configurations that define a scope for CPP Data to be received by a Service Provider

Marketplace: Marketplace Web Application

Measurement Channel: sampler of the data the signals process

MP: Marketplace

My Data Wallet: group of MP functionalities for Data Owners

OEM: Original Equipment Manufacturer

Offer: published and available Data Request

Service Provider: actor who receives the data created by owners to use it on the creation or improvement of services

Signal: information provider of the data the CPP sensors generate

System: the whole lot of applications that conforms Cross-CPP, including Marketplace Web Application and Marketplace Server.

Figures

CROSS-CPP

About Cross-CPP

The objective is to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research is to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project will develop: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to pro-vide enhanced local weather forecast and predict and optimise energy consumptions in households. The project will build upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, will be used as a basis for further development aiming to extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

Funded by the Horizon 2020 Framework Programme of the European Union

5 Annex 4: User Guide Service Provider

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

CROSS-CPP

Cross-CPP MARKETPLACE
USER GUIDE

CPP DATA CONSUMER GUIDE
(CROSS-SECTORIAL SERVICE PROVIDER)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780167

Introduction

Service Providers User Guide describes Service Providers functionalities within Cross-CPP Marketplace such as discovering available data, requesting data from CPP owners, managing AEON subscription channels, understanding the utilities and how to use the analytics toolbox and its requests and results, the data views filtering utility, and to understand the use of the context monitoring and extraction module.

Purpose

This guide aims to help users from Service Provider companies on how to use the platform and give knowledge about the different functionalities available.

Audience

This guide is meant for and solely for users of Cross-CPP Marketplace with Service Provider role. Other roles can find their own user guides.

Scope

The contents of this guide are meant to be taken into consideration only when using the Cross-CPP Marketplace and will only cover functionalities meant to be used by the role stated above.

Cross-CPP team does not take responsibility on bad use of the application or the data provided when not following the instructions given in this guide.

Troubleshooting

For any questions or inquiries about the use of the Cross-CPP Marketplace web application or the contents of it or this guide, , or if you find there is no content in this guide for some functionality please forward it to: marketplace-support@cross-cpp.eu.

Contact

Cross-CPP Project website: https://cross-cpp.eu

Cross-CPP Marketplace: https://datagora.eu

Marketplace support: marketplace-support@cross-cpp.eu

Context Monitoring and Extraction Module (CME): context-support@cross-cpp.eu.

Contents

Guide

There are two main functionalities in the Cross-CPP Marketplace (Figure 1):

- **Catalogue**: list of all available registered signals and their assigned measurement channels. (2.1 Catalogue)
- **Data Discovery**: discover the data available in the Cross-CPP Marketplace through a set of filters to create your Data Requests for the data you are interested in. (2.2 Data Discovery)

Cross-CPP Marketplace offers two distinct sections for Service Providers, grouped in the Service Provider Wallet, apart from the main functionalities (Figure 2):

- **Data Wallet**: is considered as the customer central point where they can take the control of their user profile options, data subscriptions and entity information. (3 Service Provider Wallet)
- **Toolbox**: offers a series of tools to get different metrics on the data received by users for your active data requests. (4 Toolbox)

At a glance, Cross-CPP Marketplace main functionalities can be access directly through the side menu

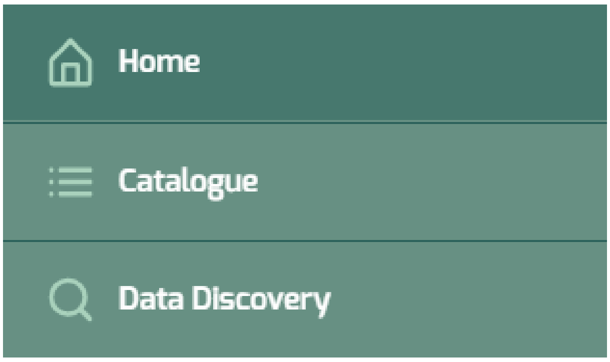


Figure 1. Main functionalities menu

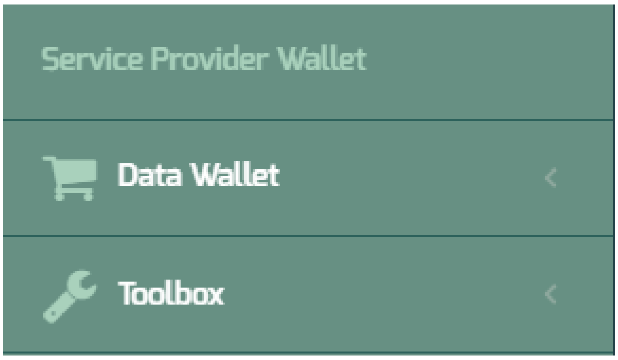


Figure 2. Service Provider Wallet menu

Other user related functions can be found in the top right corner of the toolbar which shows the F.A.Q. and a menu including Marketplace *Profile*, User *Settings* and *Log out.*

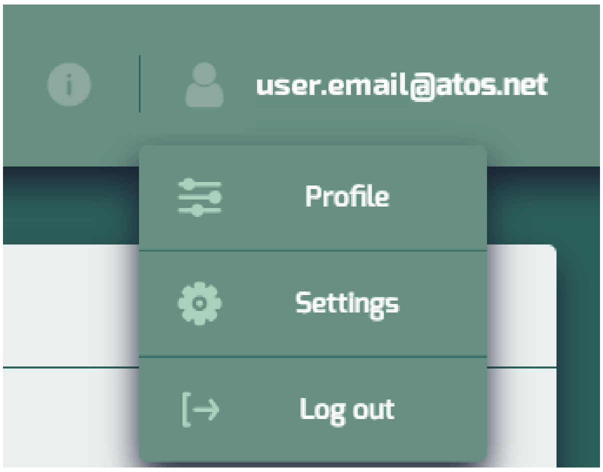


Figure 3. User menu

1. How does Cross-CPP Marketplace work?

1.1. Basic knowledge

Cross-CPP Marketplace works under the premise of letting Service Providers decide which data they want to receive and subscribe to, while giving CPP Owners control over their data at all times.

This leads to the following premises:

- Service Providers request specific data through a set of filters given, creating Data Requests
- CPP Owners decide which Data Requests they want to accept based on the filters decided by the Service Provider
- Any number of CPP Owners can accept any number of Data Requests

Data Requests are further explained in section 3.1 What is a data request?

Cross-CPP Marketplace offers two ways of getting data:

- Pull mode: traditional API request using the Data Requests identification
- Push mode: subscription to an AEON channel to get data as soon as it is available in the system

Both approaches are further explained in section 3.3 Mechanisms to subscribe to Cross-CPP data requests (pull & push mode).

Cross-CPP Marketplace offers a Toolbox including a series of Analytics and Data views. These act as a service themselves and allows Service Providers to use data coming from Data Requests the way they could do in case they have those tools in their own systems, while saving the time and effort to configure them. The Toolbox is further explained in section 4 Toolbox.

1.2. Step by step

In a nutshell, these are the steps on how to work within Cross-CPP Marketplace and the possible interactions:

1. User creation
    a. Service Provider registration
    b. System Administrator validates Service Provider registration
2. Discovering data
3. Getting data
    a. Create Data Request
    b. Subscribe to provided AEON channel for that Data Request
    c. CPP owners accept Data Request
        i. If subscribed, Service Provider would receive data as soon as it is available in the system
        ii. Service Provider can always request data collected within Data Request via API request (see Service Provider Developers Guide)
4. Analyse data
    a. Create analytics from Data Request
        i. If one-time analytics results will be instantly delivered and displayed on screen
        ii. In case analytics are services that consume data delivering results over time an AEON channel will be provided to subscribe to
    b. Delete analytics

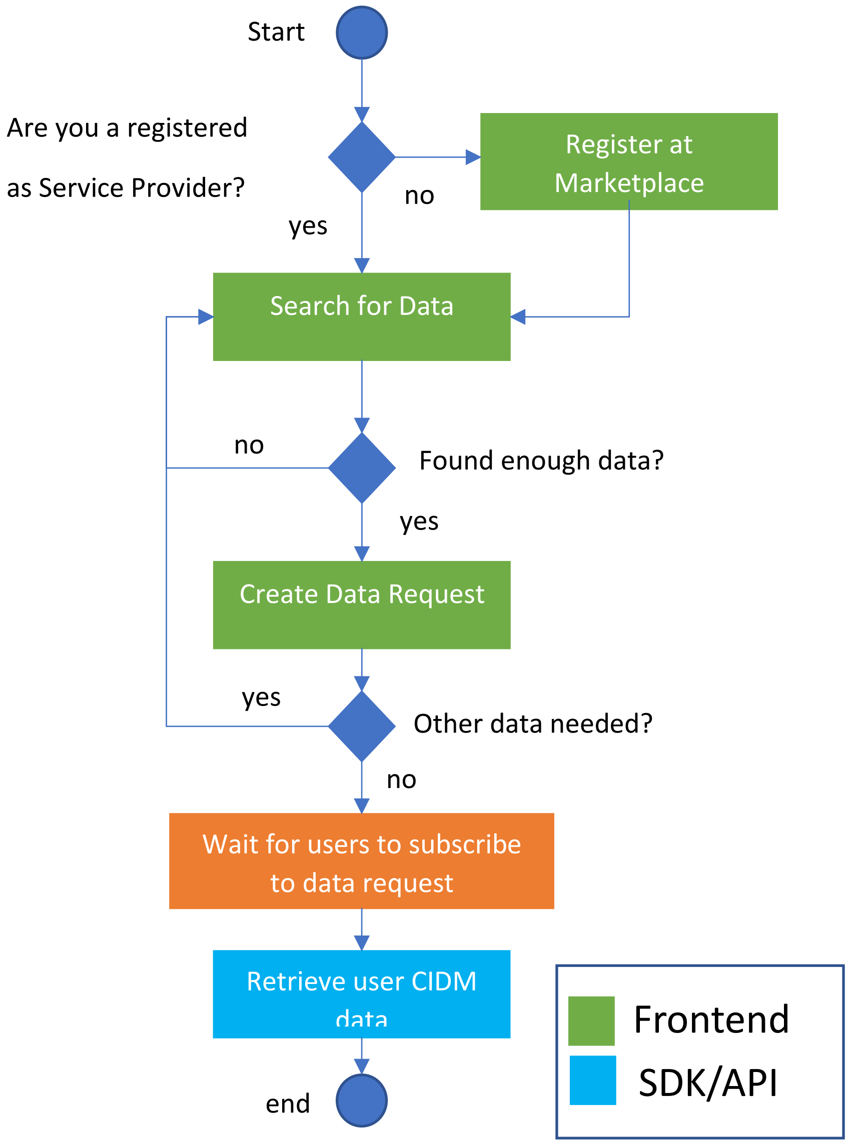


Figure 4. Cross-CPP Marketplace general workflow

1.3. Registration at Marketplace

In order to identify towards the Marketplace, Service Providers need to create an account. In the 

button to create a Service Provider account request.

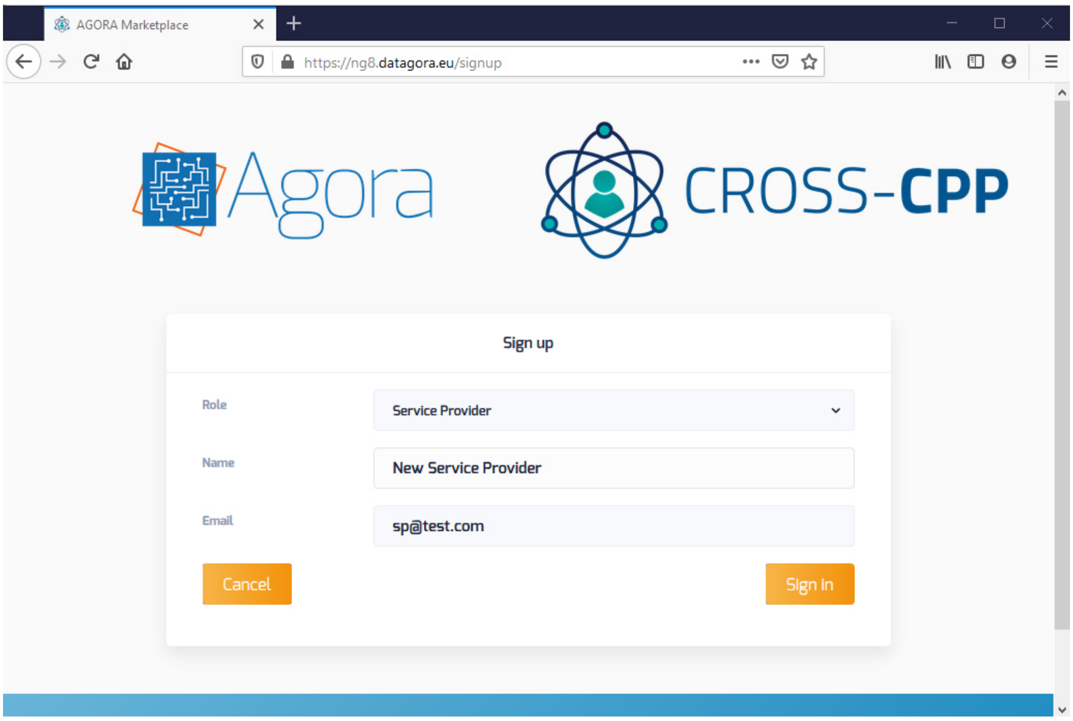


Figure 5. Cross-CPP Marketplace sign up form

After filling the sign-up form, the marketplace administrator will receive a notification to validate the service provider. Once the Service Provider user is validated, the user receives the credentials for Sign In into the Marketplace.

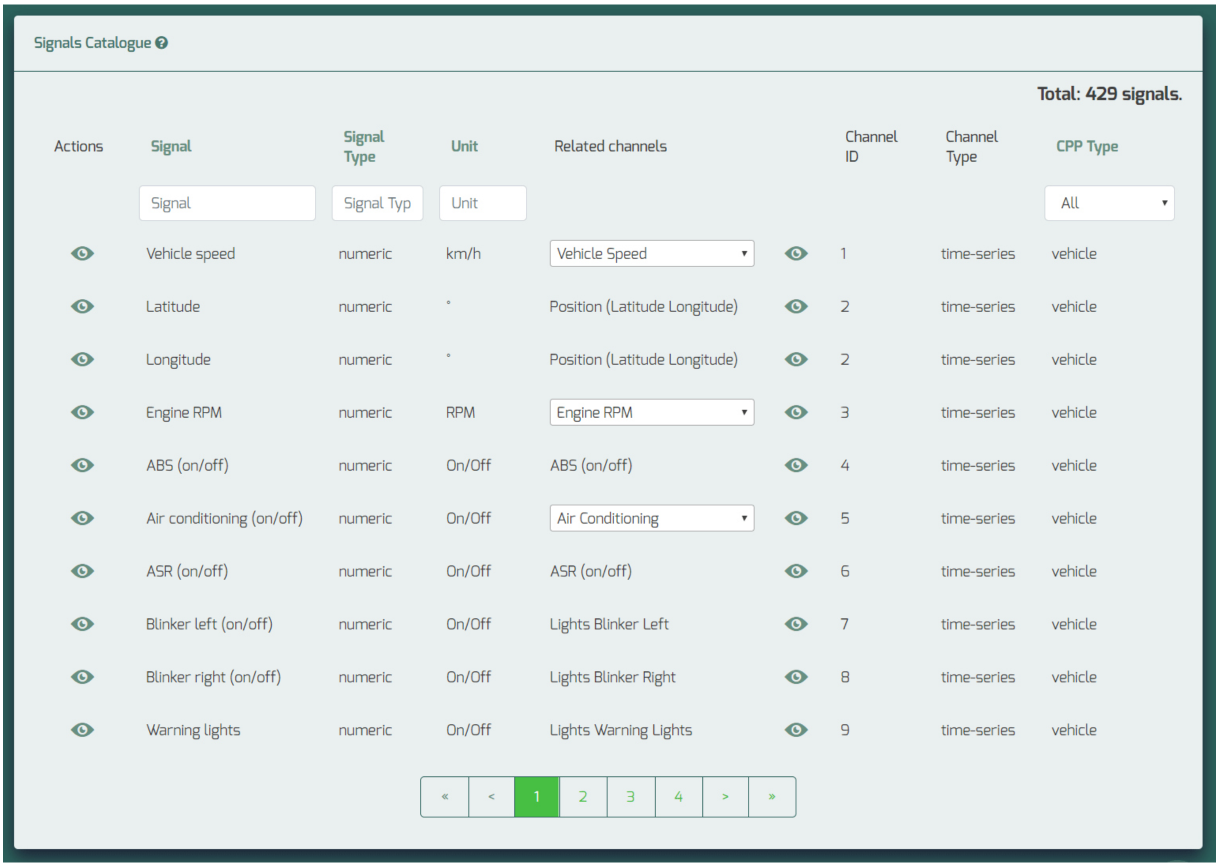
2. Main functionalities

2.1. Catalogue

The Catalogue (see Figure 6) shows a list of all available signals under a harmonized data model called Common Industrial Data Model (from now on CIDM – see F.A.Q in section 8) which enables to retrieve data from Data Providers (CIDM has a dedicated guide to be found in cross-cpp.eu). The data in the Cloud Storage is stored following the CIDM which harmonizes the proprietary OEM formats into brand independent data formats (made then available in the catalogue of the marketplace).

In the Catalogue, a Service Provider can search and filter by different attributes and see the channels that use each signal. Below, the filters which appears in Signals Catalogue are explained (see Figure 7):

- **Signal**: the name of the signal describing what is measured.
- **Signal type**: type of the signal data. Following the CIDM this can be "numeric", "enumeration", "information" or "general-purpose".
- **Unit**: units of the data
- **Related Channels**: data channels that use the data received from this signal. In case there is more than one that use the same signal, a list is available to select which one is of interest.
- **Channel ID**: identification of the selected channel that receives data from the signal
- **Channel Type**: selected channel type of data receiving. Following the CIDM this can be "time-series", "histogram", "geo-histogram" or "general-purpose".
- **CPP Type**: from which type of CPP this signal receives data from. At the moment this can be from vehicles or buildings.



Signals Catalogue

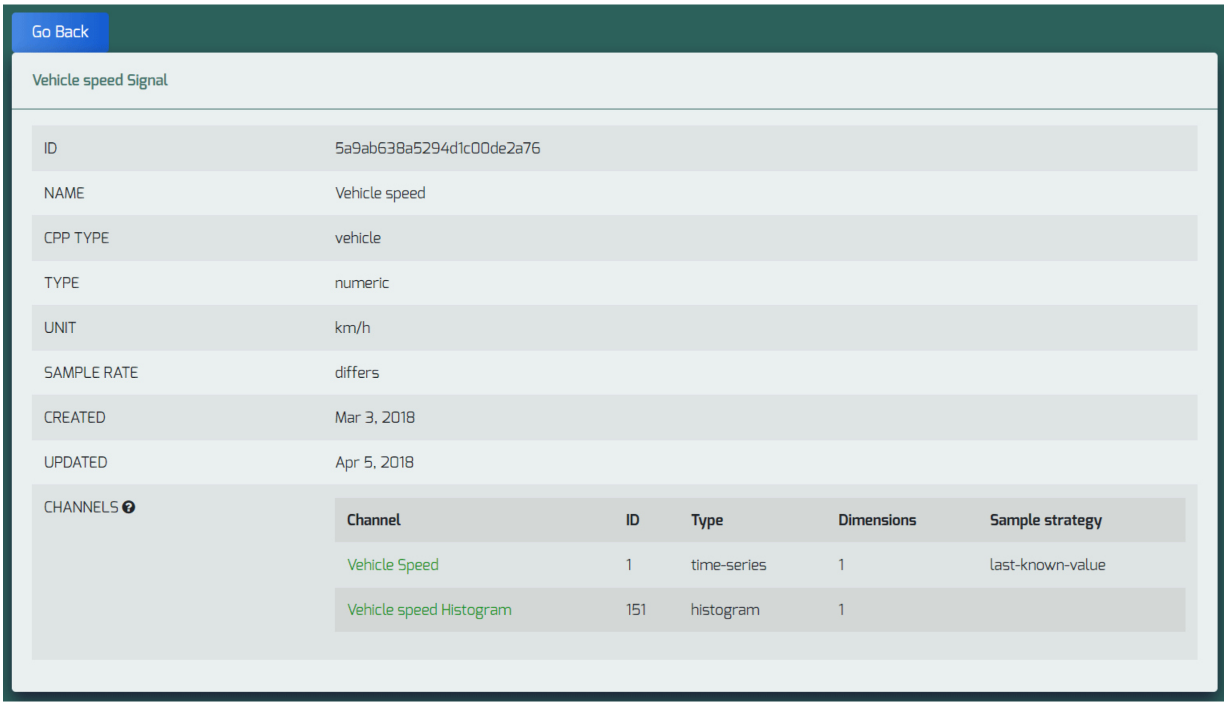
Total: 429 signals.

Actions	Signal	Signal Type	Unit	Related channels		Channel ID	Channel Type	CPP Type
	Signal	Signal Typ	Unit					All
	Vehicle speed	numeric	km/h	Vehicle Speed		1	time-series	vehicle
	Latitude	numeric	°	Position (Latitude Longitude)		2	time-series	vehicle
	Longitude	numeric	°	Position (Latitude Longitude)		2	time-series	vehicle
	Engine RPM	numeric	RPM	Engine RPM		3	time-series	vehicle
	ABS (on/off)	numeric	On/Off	ABS (on/off)		4	time-series	vehicle
	Air conditioning (on/off)	numeric	On/Off	Air Conditioning		5	time-series	vehicle
	ASR (on/off)	numeric	On/Off	ASR (on/off)		6	time-series	vehicle
	Blinker left (on/off)	numeric	On/Off	Lights Blinker Left		7	time-series	vehicle
	Blinker right (on/off)	numeric	On/Off	Lights Blinker Right		8	time-series	vehicle
	Warning lights	numeric	On/Off	Lights Warning Lights		9	time-series	vehicle

« < 1 2 3 4 > »

Figure 6. Signals catalogue

Also, each signal and channel have its own detailed view with all available information, including direct links to each other.



Go Back

Vehicle speed Signal

ID	5a9ab638a5294d1c00de2a76
NAME	Vehicle speed
CPP TYPE	vehicle
TYPE	numeric
UNIT	km/h
SAMPLE RATE	differs
CREATED	Mar 3, 2018
UPDATED	Apr 5, 2018
CHANNELS	

Channel	ID	Type	Dimensions	Sample strategy
Vehicle Speed	1	time-series	1	last-known-value
Vehicle speed Histogram	151	histogram	1	

Figure 7. Signal details

2.2. Data Discovery

The Data Discovery offers Service Providers a tool to select which data do they want to receive from CPP owners. A wide variety of filters are provided in order to help Service Providers to narrow their desired results.

In a first step, the Service Provider has to select the signals for the data of interest. If none is selected, a search for all channels will be performed. Once a signal is selected the channel that uses that signal will be added to the selection list. Selected channels can be removed from the list at any time.

If one or more channels are selected suggestions can be requested. This is possible thanks to the implementation of the Context Monitoring and Extraction module (CME) (check additional information in F.A.Q. in section 0) within the Data Discovery process. In case suggestions are requested, the CME will provide a list of channels related to those already selected[1]. Channels of interest can be added directly from the suggestion list. Also, suggestions can be requested again each time a new channel is added.

Once the channels are selected, the user may apply several filters to the data packages (see Figure 8):

- **Date and duration** section filter the results by data package submission date, measurement recording date and measurement time duration.
- The **geographical selection** offers a way to narrow results to view data only from a specific geographical area. This can be set selecting a country in the list, or drawing a customized area entering the area bounds coordinates.

[1] Suggestions are provided based on a context model that is the result of analysis of the physical relation between signals (e.g. in case temperature is a selected signal, the CME would suggest the humidity and sun intensity

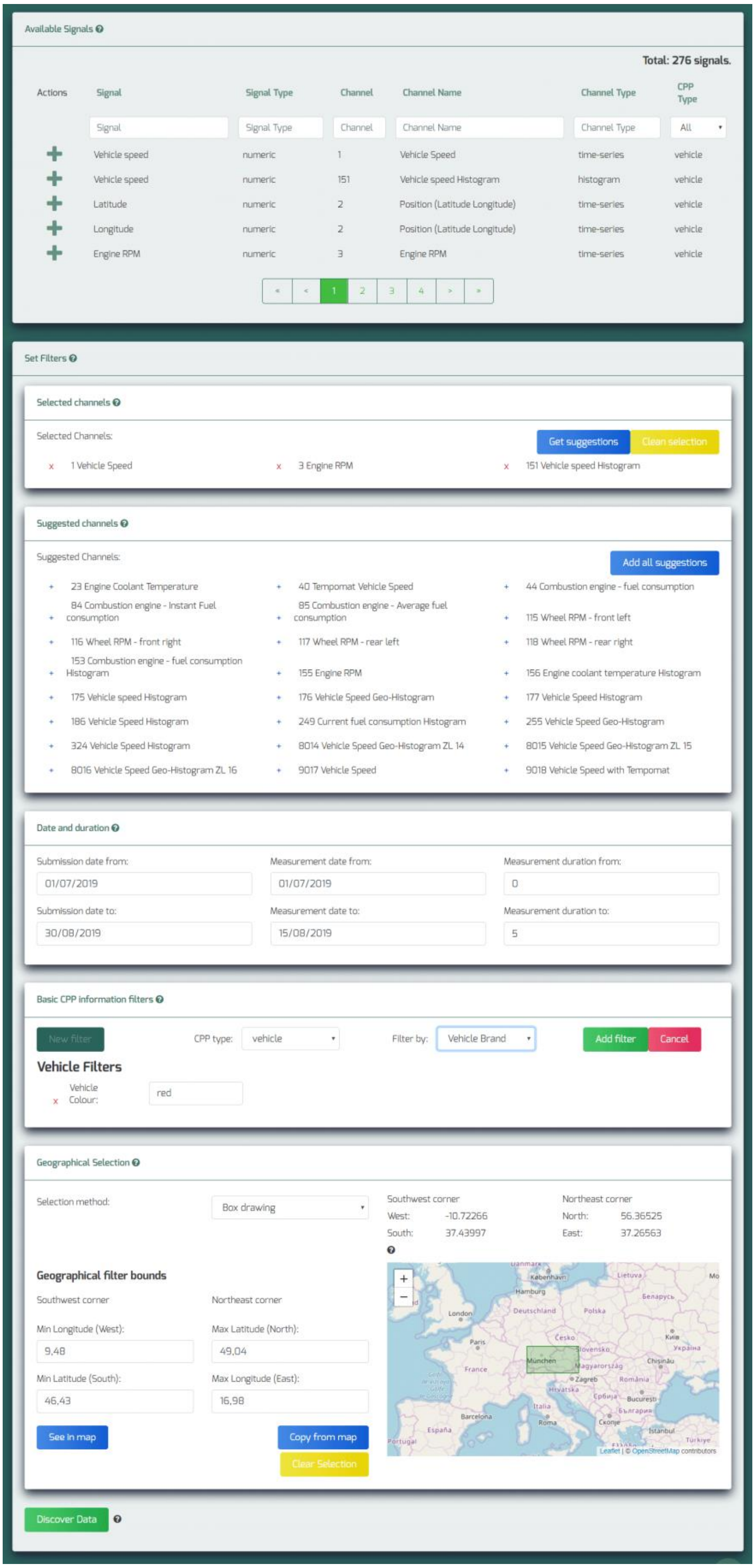


Figure 8. Data Discovery filters

Once the filters are set, Service Providers can see the data they would receive from a data request with selected configuration (See Figure 9). In this process only metadata is used in this process and no real data can be shown or retrieved through this functionality:

- A box with some **general statistics** shows the total of CPP owners and the number of selected channels that contains any data, and the number of data packages retrieved in the discovery.
- For each CPP type with data within selected channels a box is displayed showing the number of entities and channels with data, and the minimum, maximum and average duration of its measurements.
- The **heatmap** offers an overview of the geographical areas that contain data, in case it is desired to narrow the discovery to a specific area.
- The **pie charts** show how the data is distributed across the different channels. Sectors can be hidden if needed.
- The **line and bar charts** display the amount of data packages retrieved depending on its submission date and time duration.

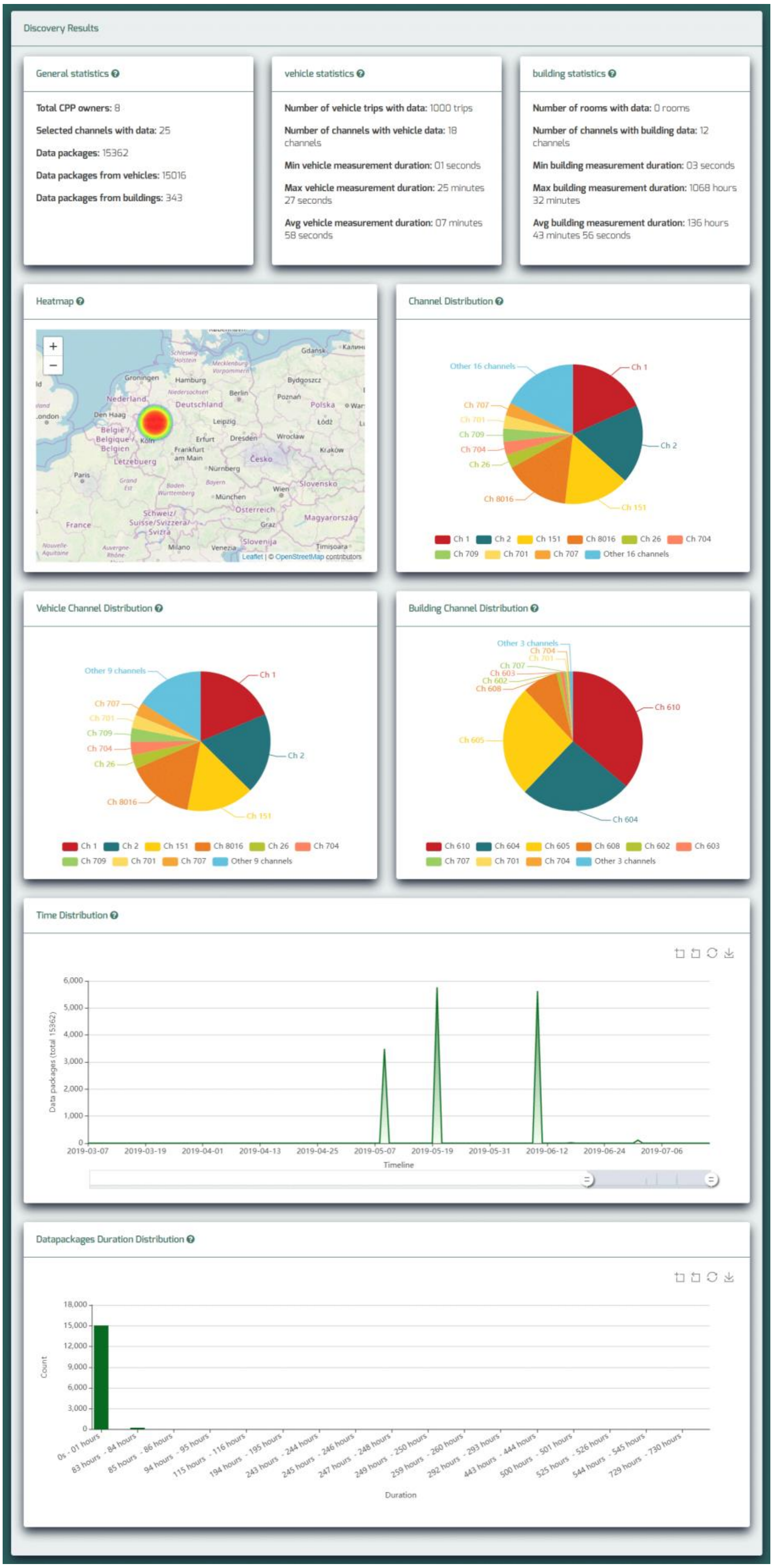


Figure 9. Data Discovery results

If the Service Provider is not satisfied with the search results the filters can be modified without having to restart the process from scratch. Service Providers can adjust the filters and browse the data until they find what they were looking for.

Once Service Providers agree with the data they would receive, they can configure a set of additional options (Figure 10):

- **Analytics**: by activating this option the Service Provider declares the intentionality of using the data to be received for analytics, so the owner can give the consent to do so (see more details in section 4 Toolbox).
- **Context**: by activating this option the Service Provider declares that he wants data to be received to be analysed by its context and to receive data fitting selected context filters. A list of filters will be provided to configure the context filter for the Data Request, such as day of the week, meteorological condition, and so on (see more details in section 3.4.1 Context Sensitive filtering). Service Providers will then receive only the data fitting those filters produced by owners that allow the context analysis.

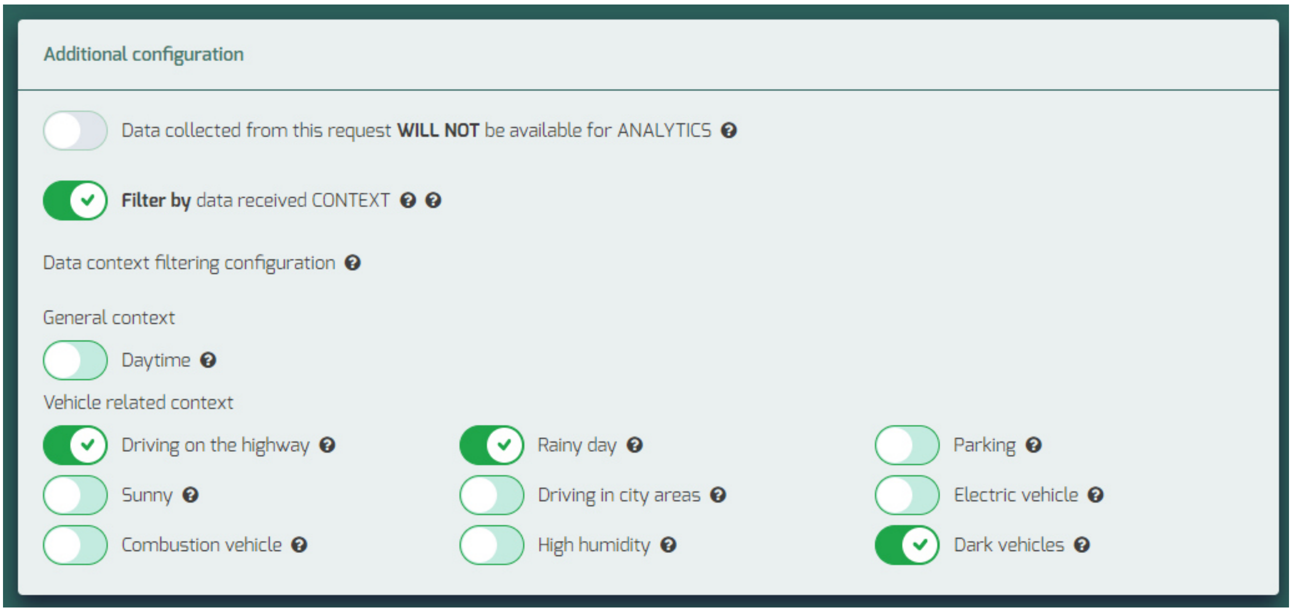


Figure 10. Data Discovery – Data Request additional configuration

Finally, they can set a description name for the data Request and indicate additional options for the data they will receive.

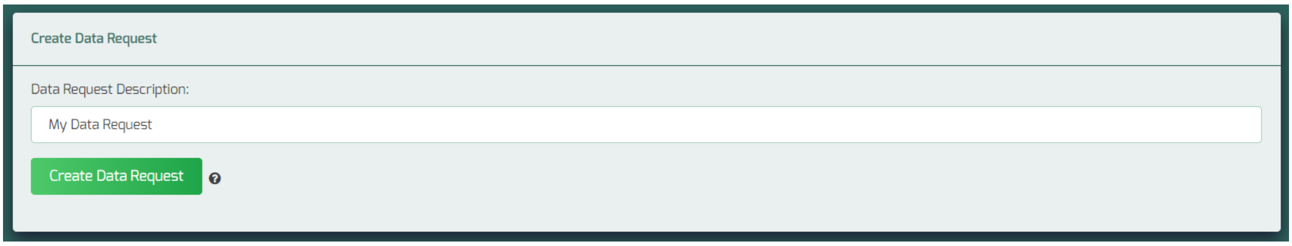


Figure 11. Data Discovery - Create Data Request

3. Service Provider Wallet

The Service Provider wallet (from now on Data Wallet) (see Figure 12) group different views for data review in which the user can:

- Data Requests:
  - List published data requests
  - Review each data request details
  - Find each data request AEON channel configuration
  - See the data packages received from each data request
- Data Transactions:
  - Get a resume of each data request package transactions

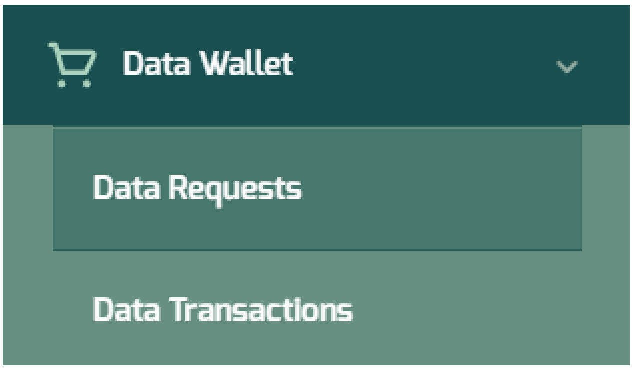


Figure 12. Data Wallet menu

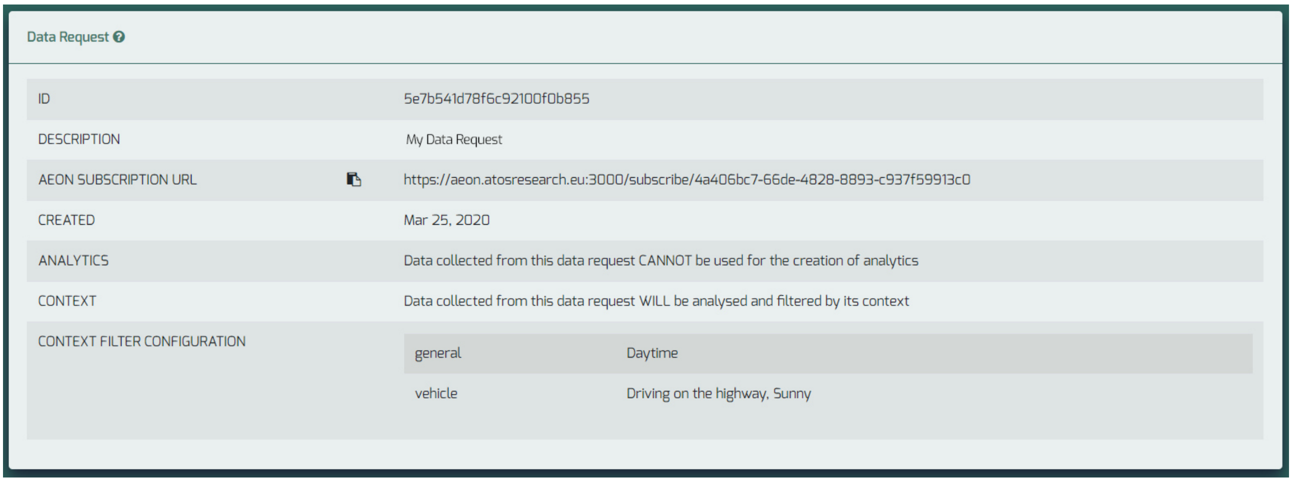
3.1. What is a data request?

Data Request stands for a configuration set by a group of filters in order to receive specific data within the desired scope of our services designing. This scope is set during the Data Discovery (see section 2.2 Data Discovery above) and saved when creating the Data Request.

Once the Data Request is published by the Service Provider, the CPP owners can then review and accept it, and thus consent to deliver their data through this specific data request and to this specific Service Provider.

Service Providers will receive all data from CPP owners that accepted that request and that is exclusively for the scope of the Data Request published.

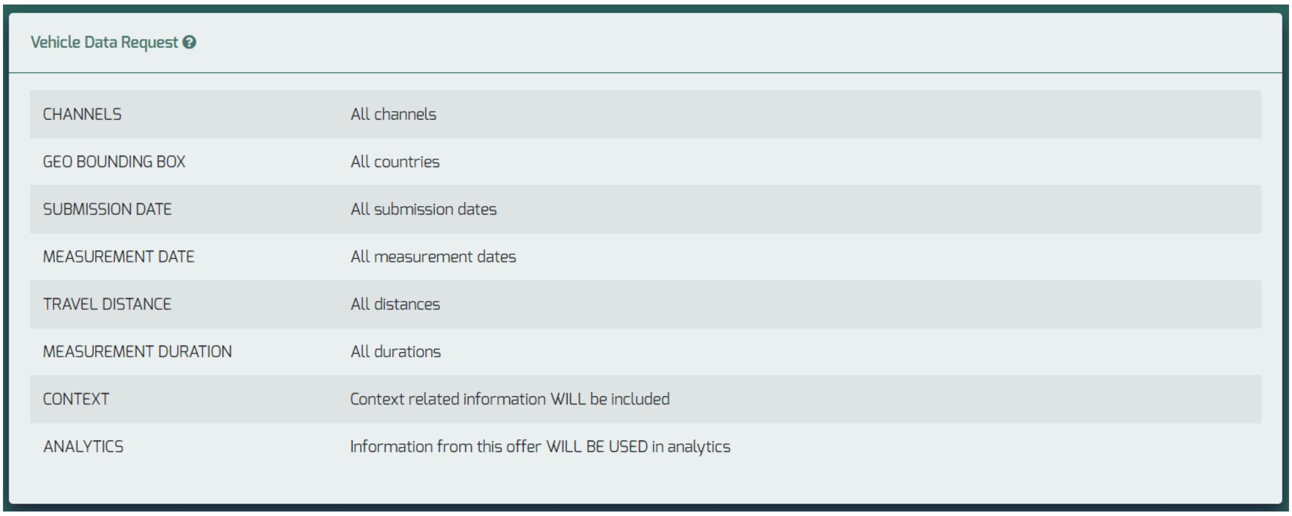
In the Data Requests view a table with published Data Request is shown. Selecting one will lead to a detailed page in which the AEON subscription channel ID and URL can be consulted, along with the full configuration and a table of known sent data packages.



Data Request

ID	5e7b541d78f6c92100f0b855	
DESCRIPTION	My Data Request	
AEON SUBSCRIPTION URL	https://aeon.atosresearch.eu:3000/subscribe/4a406bc7-66de-4828-8893-c937f59913c0	
CREATED	Mar 25, 2020	
ANALYTICS	Data collected from this data request CANNOT be used for the creation of analytics	
CONTEXT	Data collected from this data request WILL be analysed and filtered by its context	
CONTEXT FILTER CONFIGURATION	general	Daytime
	vehicle	Driving on the highway, Sunny

Figure 13. Data Request details



Vehicle Data Request

CHANNELS	All channels
GEO BOUNDING BOX	All countries
SUBMISSION DATE	All submission dates
MEASUREMENT DATE	All measurement dates
TRAVEL DISTANCE	All distances
MEASUREMENT DURATION	All durations
CONTEXT	Context related information WILL be included
ANALYTICS	Information from this offer WILL BE USED in analytics

Figure 14. Data Request filter details

3.2. How to publish a data request?

A data request is configured through the Data Discovery process. For more information about the Data Discovery please go to section 2.2 Data Discovery above.

3.3. Mechanisms to subscribe to Cross-CPP data requests (pull & push mode)

There are two possible methods to retrieve data received through the published Data Requests.

3.3.1.1.Pull mode

Data can be manually pulled by the Service Provider from the Marketplace.

This mode relies on the Service Provider to get the data from an API endpoint. More information about this can be found in the Service Providers Developers Guide (to be found under cross-cpp.eu).

3.3.1.2. Push mode

Data is automatically pushed from the Marketplace to the Service Provider through a provided AEON channel.

The AEON channel subscription URL can be found in the details view provided in Data Requests view (Figure 15). The clipboard button can be clicked to copy the content of each text.

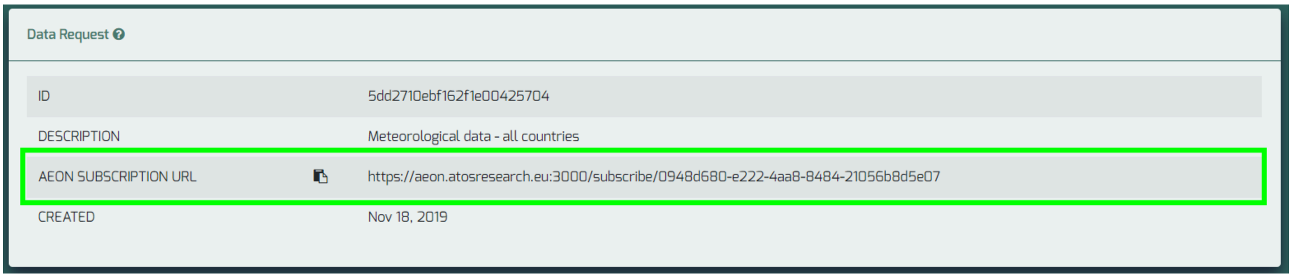


Figure 15. Data Request AEON channel details

Information about AEON configuration can be found in the Service Providers Developers guide (to be found under cross-cpp.eu).

3.4. Additional Data Request configuration

There are some additional configuration options available during the creation of the Data Request such as context sensitive filtering and declaration of purposes to which the CPP owner must give consent in order to approve sharing data.

3.4.1. Context Sensitive filtering

During the Data Discovery process, the Service Providers may apply special filters to the data packages that they want to receive. These filters are based on the context information that can be extracted by the Context Monitoring and Extraction (CME) module from the data provided by Data Owners (the CPP data) as well as CPP basic information that is not CPP runtime monitored data but basic information such as vehicle brand, model or colour.

The extracted context information is inferred from the collected data, the context models are developed for the CPP type in question, by combining several signals and applying rules to the data. An example of such extracted information is to infer if the data is collected while a vehicle is on a highway (or not) or if it's raining (or not).

These data filters will be included in the Data Request and applied to the data being collected at runtime and filtered before being sent to the Service Provider.

3.4.1.1. What is the Context Monitoring and Extraction module (CME)?

The CME is the module that by monitoring CPP data streams, basic CPP information and context models of the different CPPs being handled by the Cross-CPP Marketplace, can provide extracted context to other Cross-CPP modules for semantic adaptation, such as the here explained context sensitive filtering or for the Context Sensitive Security (see section 5.1 Context Sensitive Security).

More details can be seen in the Cross-CPP Ecosystem guide (to be found under www.cross-cpp.eu).

3.4.1.2. How to configure the context sensitive filtering for a Data Request

Service Providers may apply special filters to the data packages that they want to receive. In case the Service Provider would like to further refine the data to be received from the CPPs this can be achieved by selecting the option "Filter by data received CONTEXT" in the UI of the Marketplace within the *Additional configuration* tab. Once this is selected all possible context parameters that may be extracted once the Data Owner accepts the Data Request (see Figure 16).

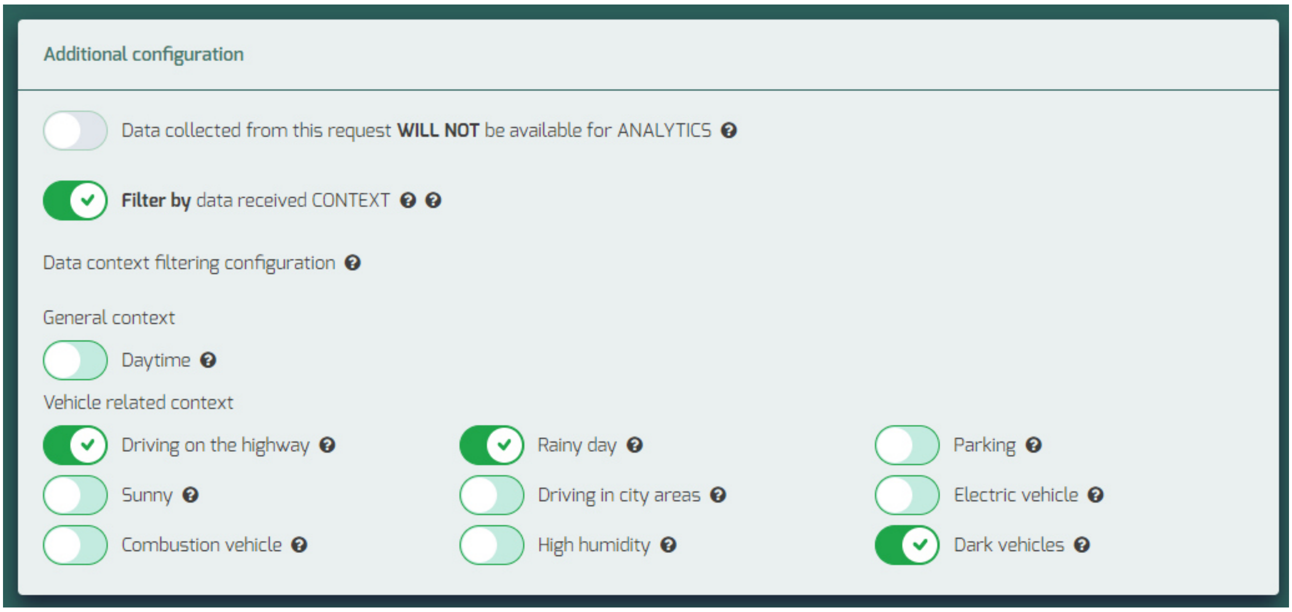


Figure 16: Context information filters

These data filters will be included in the Data Request and applied to the data being collected at runtime and filtered before being sent to the Service Provider.

Extracted context parameters can be extended by creating new context models and adding/editing the current CME rules. More information on this possibility can be seen in the Service Provider developer guide or by contact directly the CME team at context-support@cross-cpp.eu.

3.4.2. Data Views

Data Requests can be further filtered to receive specific values through dedicated AEON channels to be used as notifications or alerts, or to generate different services. For more information please see section 4.1 Data Views.

3.5. Data Transactions

Data Transactions view displays a summary of data collected from the published Data Requests, including the following information:

- total transactions
- amount of pull type transactions
- amount of push type transactions
- total size of available data
- first package received
- last package received

The details view leads to the Data Requests detail view (see Figure 13). Data Transactions view is depicted as follows including the following fields (see Figure 17):

- **Data Request**: The name of the Data Request the transactions belongs to
- **Contract transactions**: total number of transactions from users which accepted this data request.
- **Pulls**: number of times the data has been pulled
- **Pushes**: number of times the data has been sent through the AEON channel
- **Total size**: number of bytes between all transactions


Data Transactions

Total: 100 transactions.

Actions	Data Request ID	Data Request	Contract transactions	Pulls	Pushes	Total size	First	Last
	Data Request ID	Data Request						
	5cd9290dff89151c002d03e6	All channels	86	46	40	5448796	Jun 28, 2019	Jun 28, 2019
	5cd421f89c0f591e006515ab	test offer	14	4	10	24250	Jun 19, 2019	May 17, 2019

Figure 17. Data Transactions list

4. Toolbox

4.1. Data Views

4.1.1. What is a Data View?

A Data View is a configured filter of values received from a Data Request measurement channels. This filtered data is sent to the Service Provider in a dedicated AEON channel in order to provide it as a notification mechanism.

The Data Views can be used also to request the latest data available for the configured value filters.

Data Views provide the primary mechanism for providing training data in the process of building Machine Learning analytics functions.

4.1.2. What is a category and how to assign it?

A category is a way to use a Data View to analyse its collected data. It can be assigned when creating the Data View or later at the details page. Once a Data View has been assigned with a category, its rows can be assigned with a value from that category to identify the type of information that row provides.

This is a mandatory step in order to use a Data View for a Machine Learning model classifying input data into the defined categories.

4.1.3. How to create a Data View

The creation page can be found through the button above the Data Views main view.

Data Views filter data coming from already existing Data Requests, meaning the Service Provider must first select the Data Request from which data shall be filtered.

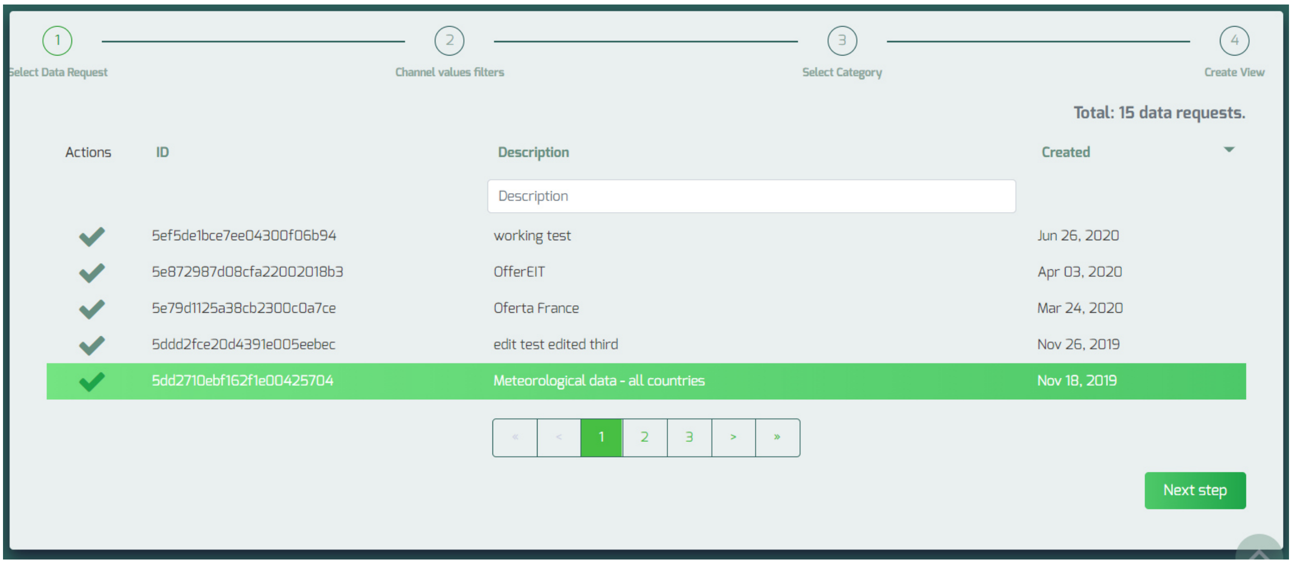


Figure 18. Create Data View – Select Data Request

Every Data Request comprises one or more measurement channels from which the request receives data. Next step is to decide from which of those measurement channels value filtering is desired.

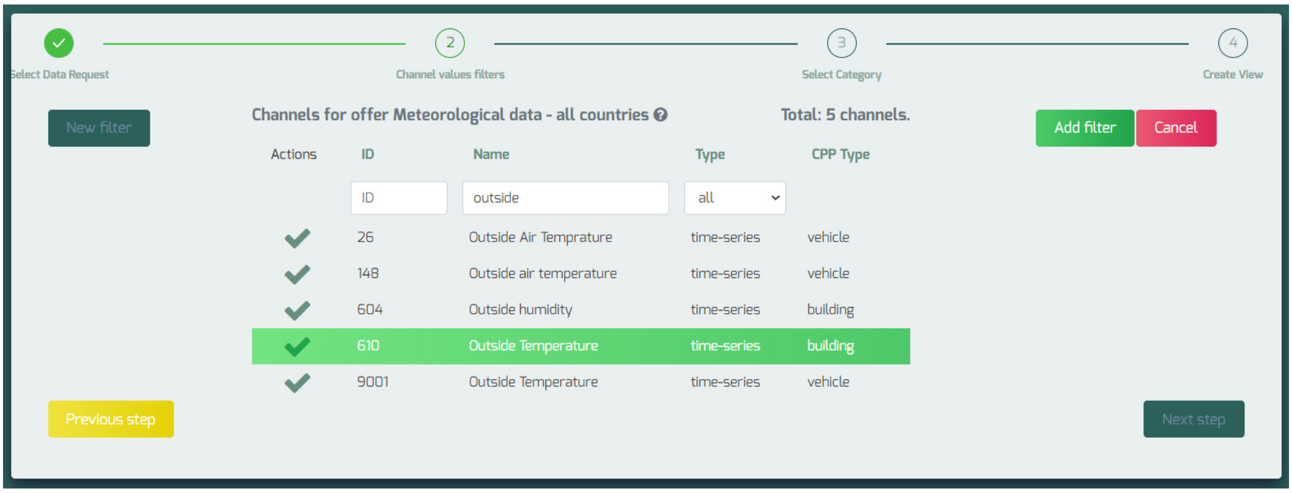


Figure 19. Create Data View - Select measurement channel to be filtered

A value filter consists of an operator and the actual value that must be configured for every selected measurement channel. Notice that a measurement channel can have more than one value filtering, such as less than a value but more than other value, but if that configuration is not possible the system will not let the user continue.

There is also the possibility that all data coming from the selected measurement channel is of interest, in which case the filter can be left empty.

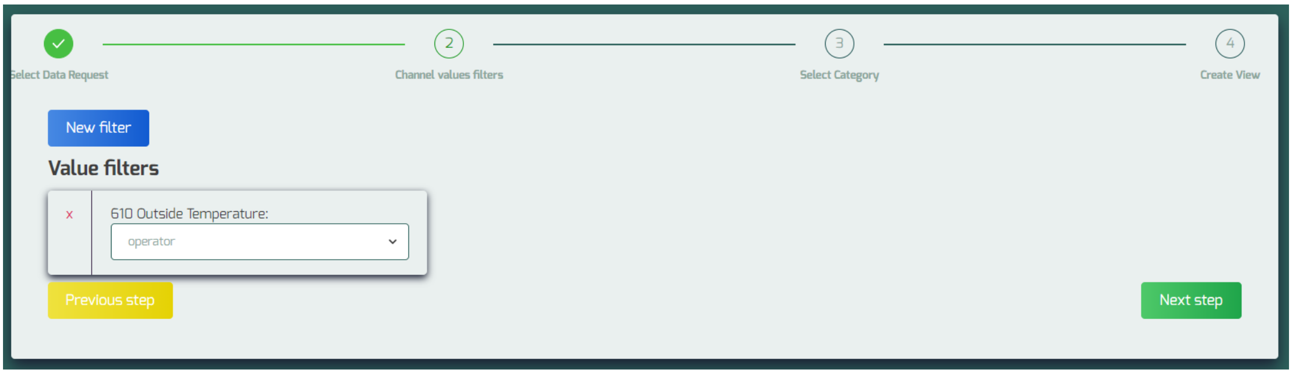


Figure 20. Create Data View – Measurement channel without value filter

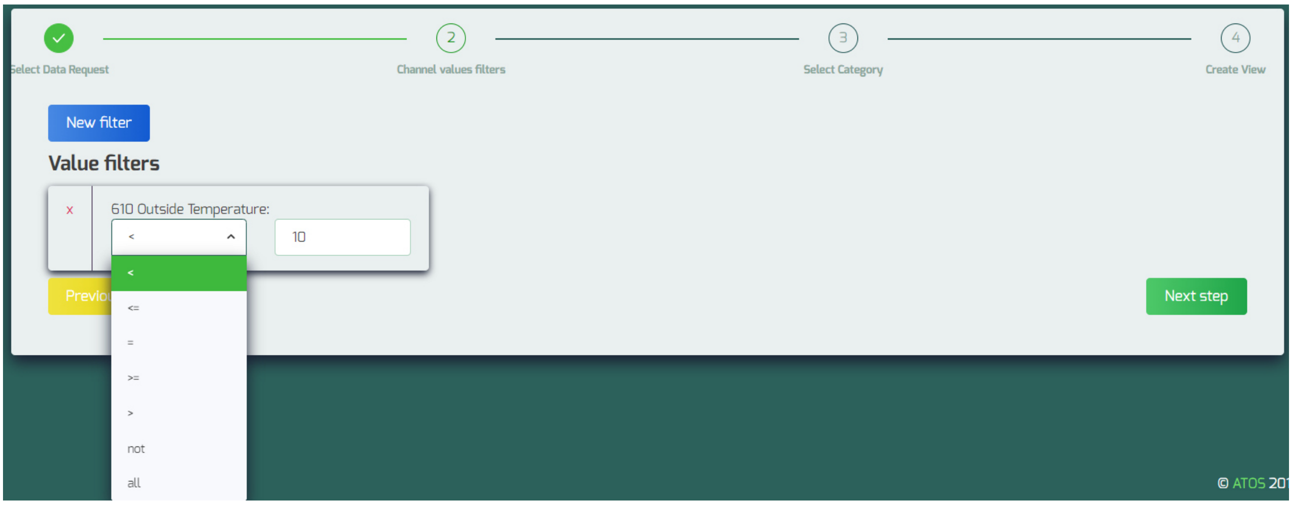


Figure 21. Create Data View - Input measurement channel value filter

Once agreed with the selected filtering, it is asked if a category is desired for the Data View. The category can be assigned now or later at the details page. The user can select an already created category or create a new one at this moment.

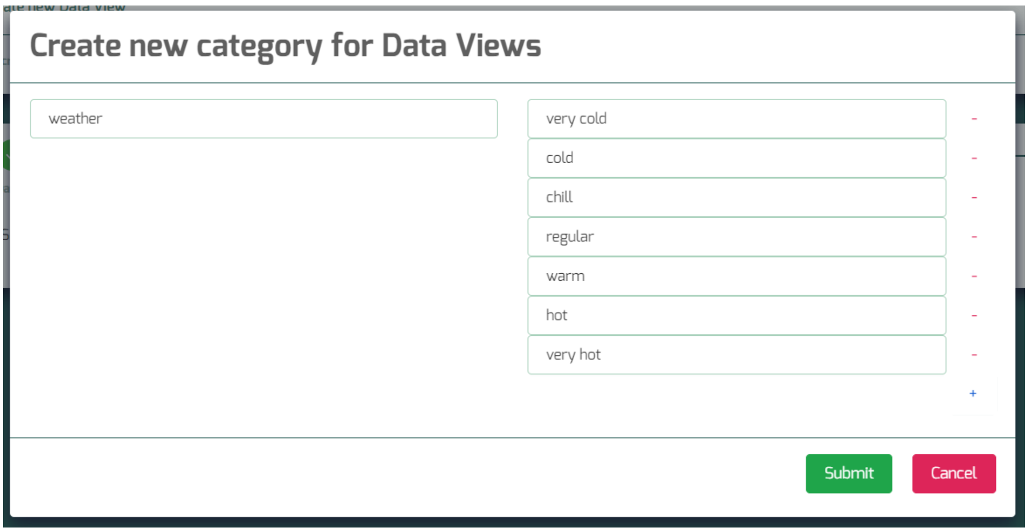


Figure 22. Create Data View – Create category

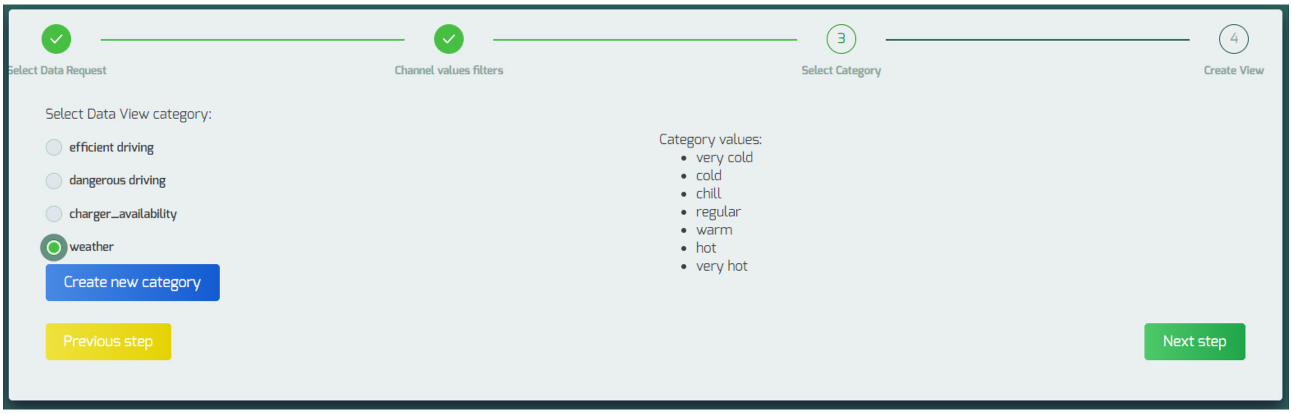


Figure 23. Create Data View – Assign category

Finally, a name and a description of the Data View are requested.

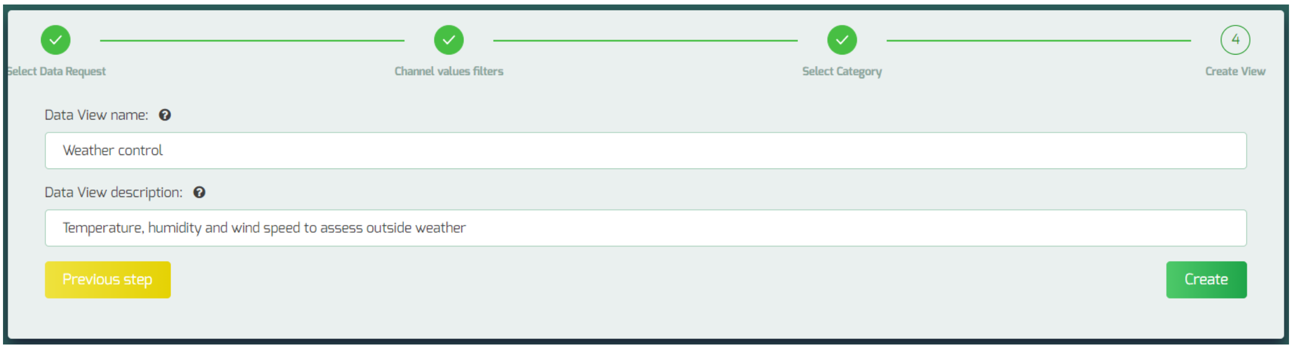


Figure 24. Create Data View - Input name and description

If successful, a view with the generated Data View name and description will be shown with a button to go to the details.

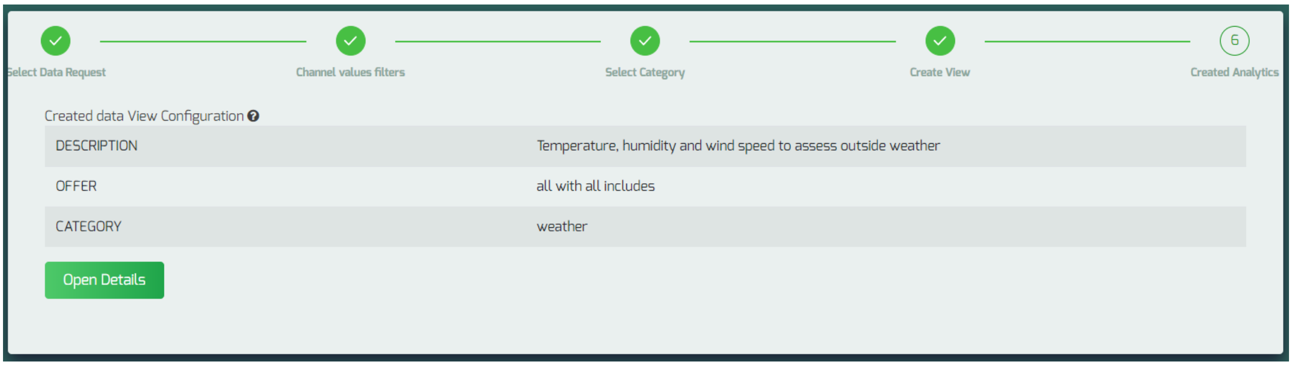


Figure 25. Create Data View - generated Data View details

4.1.4. Review Data View details

Just clicking in the Data Views menu will lead to the list view, showing a list of all Data Views created by the Service Provider.

In this list, users can filter displayed results by description, Data Request name or date.

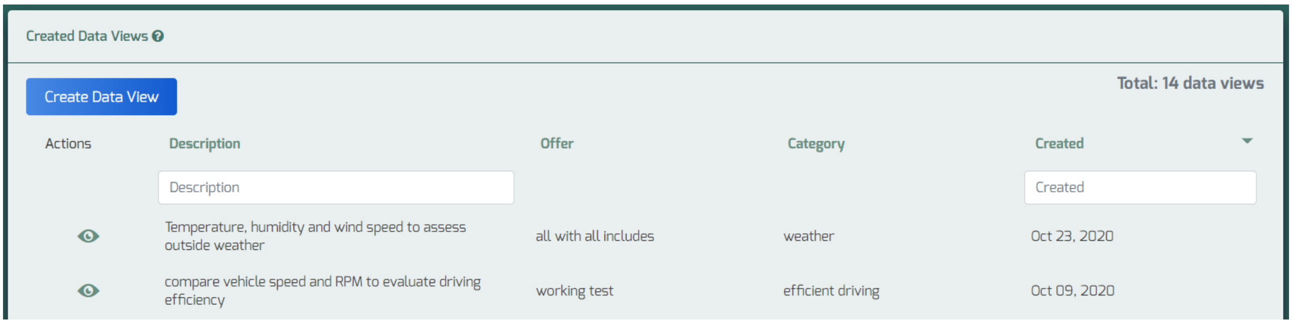


Figure 26. Created Data Views list

Clicking on any of them will open the details of that Data View, displaying the configuration, including name, description, Data Request and value filtering configuration, along with the AEON channel to subscribe to receive the data filtered.

There is also a "Delete Data View" button. By clicking and confirming this button the Service Provider will stop receiving this filtered data and the associated configuration will be deleted.

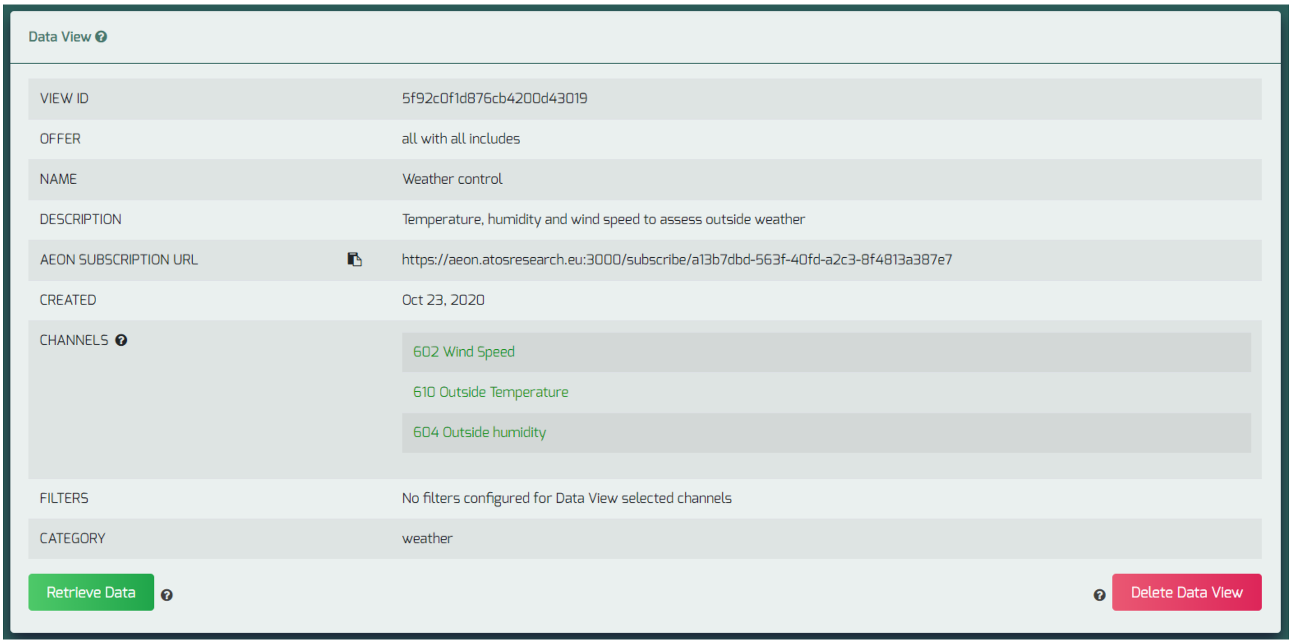
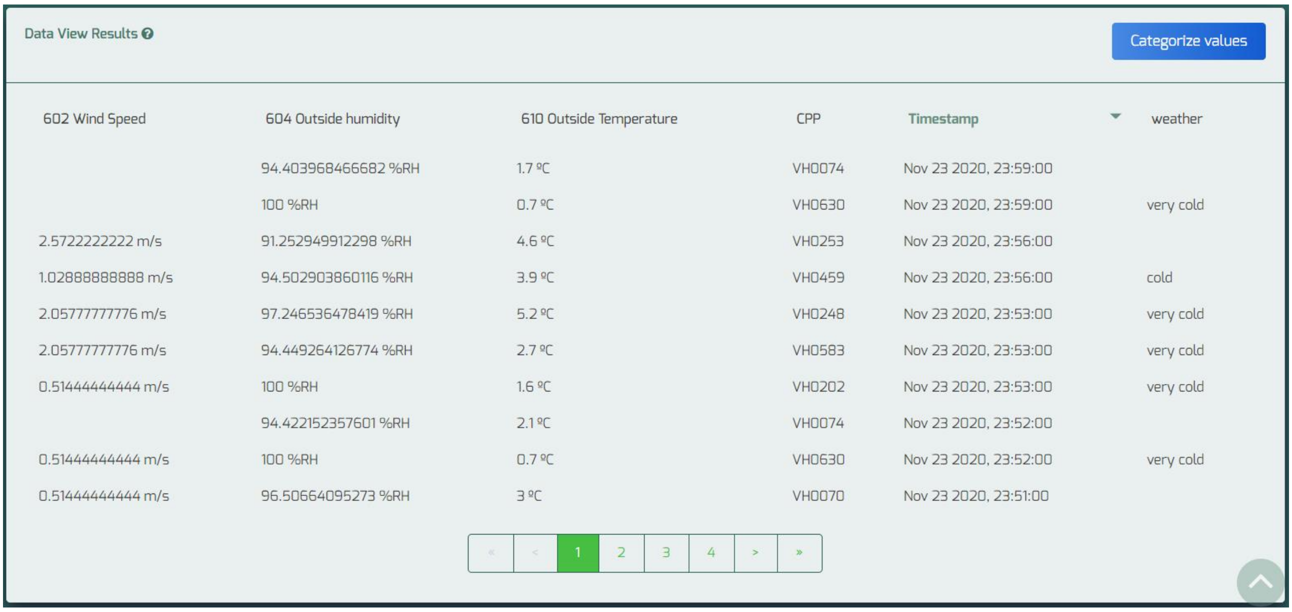


Figure 27. Data View configuration details

By clicking in the "Retrieve Data" button a table will display the latest data collected by the Data View.

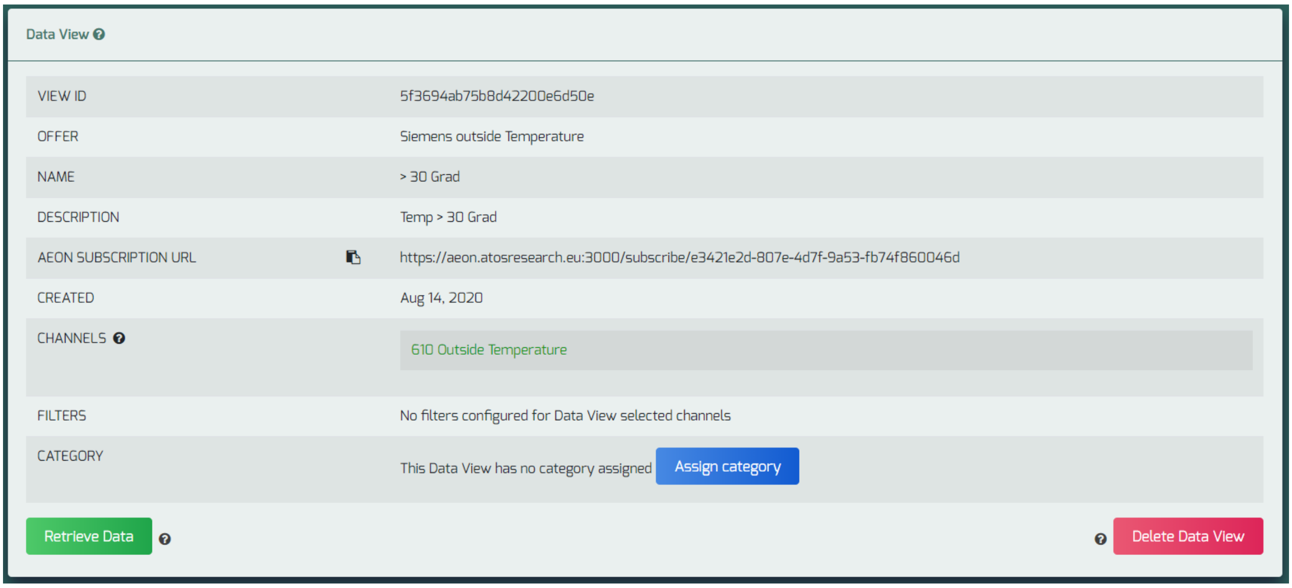


602 Wind Speed	604 Outside humidity	610 Outside Temperature	CPP	Timestamp	weather
	94.403968466682 %RH	1.7 ºC	VH0074	Nov 23 2020, 23:59:00	
	100 %RH	0.7 ºC	VH0630	Nov 23 2020, 23:59:00	very cold
2.5722222222 m/s	91.252949912298 %RH	4.6 ºC	VH0253	Nov 23 2020, 23:56:00	
1.02888888888 m/s	94.502903860116 %RH	3.9 ºC	VH0459	Nov 23 2020, 23:56:00	cold
2.05777777776 m/s	97.246536478419 %RH	5.2 ºC	VH0248	Nov 23 2020, 23:53:00	very cold
2.05777777776 m/s	94.449264126774 %RH	2.7 ºC	VH0583	Nov 23 2020, 23:53:00	very cold
0.51444444444 m/s	100 %RH	1.6 ºC	VH0202	Nov 23 2020, 23:53:00	very cold
	94.422152357601 %RH	2.1 ºC	VH0074	Nov 23 2020, 23:52:00	
0.51444444444 m/s	100 %RH	0.7 ºC	VH0630	Nov 23 2020, 23:52:00	very cold
0.51444444444 m/s	96.50664095273 %RH	3 ºC	VH0070	Nov 23 2020, 23:51:00	

Figure 28. Data View – retrieve data

4.1.5. Assign category to a Data View

To assign a category to an already created Data View just go into the Data View details page and click on the Assign Category button.



VIEW ID	5f3694ab75b8d42200e6d50e
OFFER	Siemens outside Temperature
NAME	> 30 Grad
DESCRIPTION	Temp > 30 Grad
AEON SUBSCRIPTION URL	https://aeon.atosresearch.eu:3000/subscribe/e3421e2d-807e-4d7f-9a53-fb74f860046d
CREATED	Aug 14, 2020
CHANNELS	610 Outside Temperature
FILTERS	No filters configured for Data View selected channels
CATEGORY	This Data View has no category assigned

Figure 29. Data View – details view with assign button

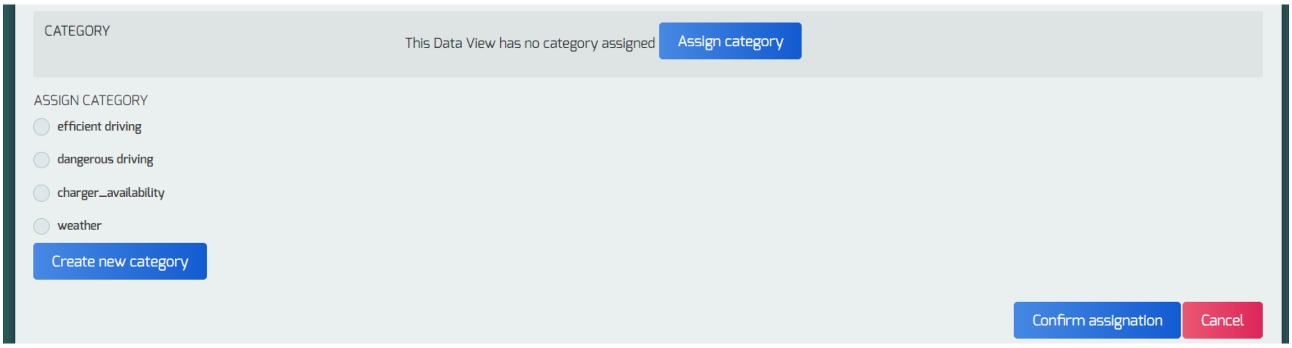


Figure 30. Data View – assign category

4.1.6. Assign category values to a Data View rows

If a Data View has a category assigned a button will appear in the Data View results table.

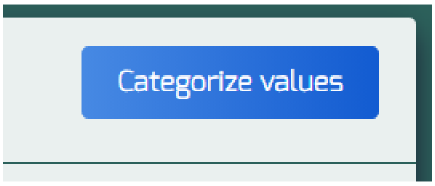


Figure 31. Data View – categorize values button

In this mode a pencil icon will appear in every row. Clicking the icon will open the value assignment popup.

602 Wind Speed	604 Outside humidity	610 Outside Temperature	CPP	Timestamp	weather
	94.403968466682 %RH	1.7 ºC	VH0074	Nov 23 2020, 23:59:00	
	100 %RH	0.7 ºC	VH0630	Nov 23 2020, 23:59:00	very cold
2.5722222222 m/s	91.252949912298 %RH	4.6 ºC	VH0253	Nov 23 2020, 23:56:00	
1.02888888888 m/s	94.502903860116 %RH	3.9 ºC	VH0459	Nov 23 2020, 23:56:00	cold
2.05777777776 m/s	97.246536478419 %RH	5.2 ºC	VH0248	Nov 23 2020, 23:53:00	very cold
2.05777777776 m/s	94.449264126774 %RH	2.7 ºC	VH0583	Nov 23 2020, 23:53:00	very cold
0.51444444444 m/s	100 %RH	1.6 ºC	VH0202	Nov 23 2020, 23:53:00	very cold
	94.422152357601 %RH	2.1 ºC	VH0074	Nov 23 2020, 23:52:00	
0.51444444444 m/s	100 %RH	0.7 ºC	VH0630	Nov 23 2020, 23:52:00	very cold
0.51444444444 m/s	96.50664095273 %RH	3 ºC	VH0070	Nov 23 2020, 23:51:00	

Figure 32. Data View – categorize values mode

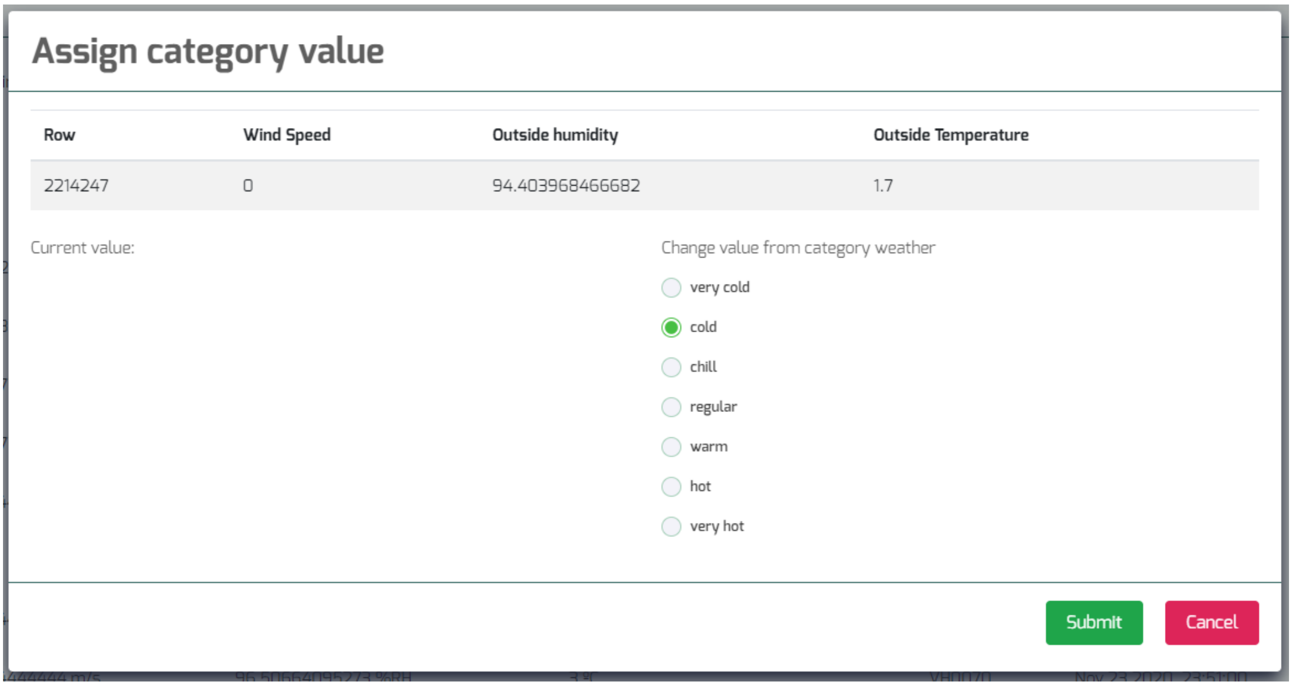


Figure 33. Data View – assign category value popup

Once all desired rows have been assigned to a value click on the “Save values” button to display a popup with a table containing the changed rows and assigned values and confirm the selection.

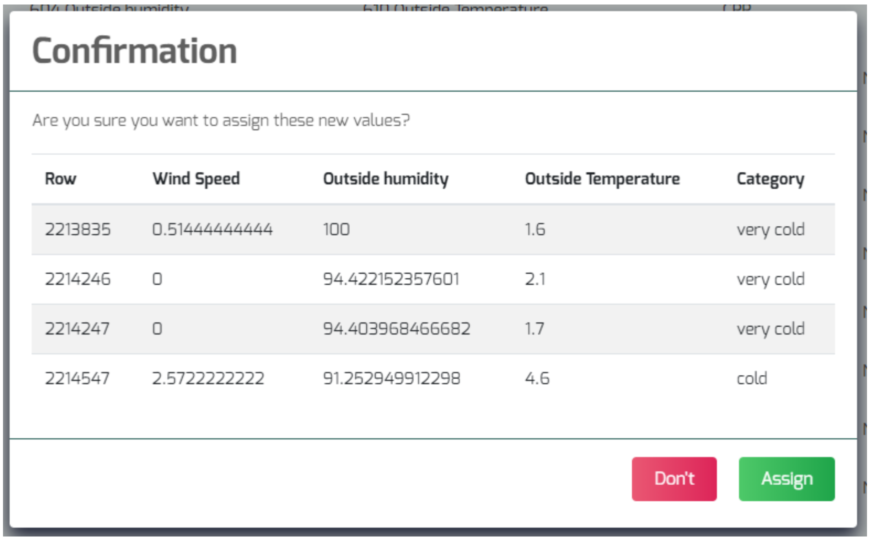


Figure 34. Data View – confirm category value assignments

4.2. Analytics

4.2.1. What is the Analytics Toolbox?

The Analytics Toolbox is a collection of tools that offers different ways of analysing the data collected from the different data requests.

Depending on the signals of the data requests some analytics may not be available.

The side menu displays the different categories of analytics. The different available tools can be found in each category.

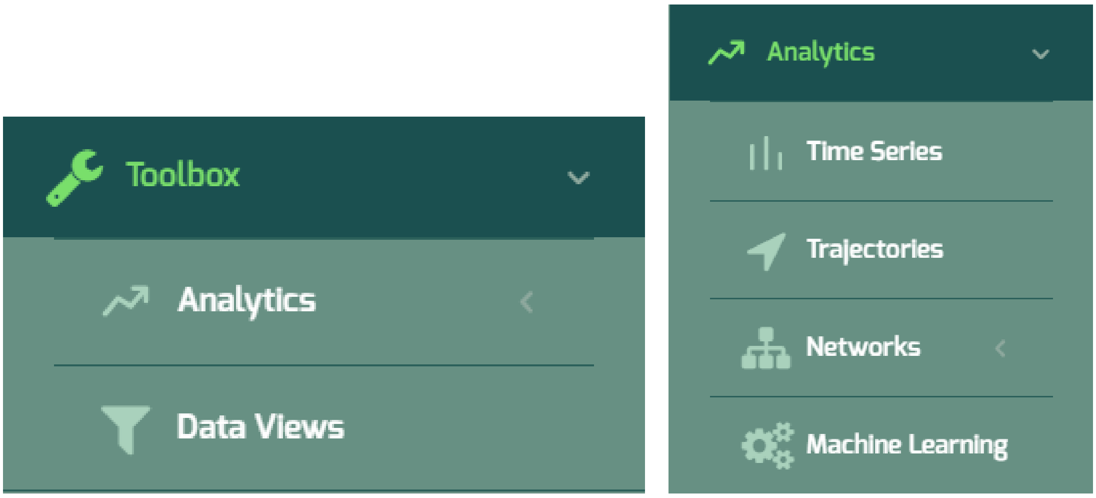


Figure 35. Toolbox menu

4.2.2. Time Series Analytics

Time series functionalities are aimed towards the management of temporal data (typically paired to timestamps that allow to trace its evolution) by enabling the inspection of its properties, as well as prediction capabilities to be able to forecast the future behaviour of a certain variable's development. In specific, the developed functions cover the following ambits:

- Drift detection
- Forecasting
- Correlation Statistics

Details about each of these subjects is presented in the following sections.

4.2.2.1. Available analytics functions services and examples

4.2.2.1.1. Drift detection

The drift present in a time series can be thought of as the changes present in the input data. In this regard, this set of analytics functions include the provision of different metrics with which to evaluate this factor, coupling it with the concept of complexity, understood as a measure of the presence of nonlinear patterns that explain the behaviour of the data.

This way, a time series is said to be highly complex when it presents underlying non-linear patterns that reveal its behaviour. On the other hand, a time series with random fluctuations in its data would have low complexity, as it would not present (as many) regular patterns in its behaviour.

With respect to the domain of this application, it provides a mean of assessing the stability with which a specific feature varies, as well as a preliminary step into further assessing its behaviour over time.

4.2.2.1.1.1. Sample Entropy

Sample Entropy is a probabilistic measure of a time series' randomness. It relies in two main parameters, an embedding dimension **e** and a tolerance value **t**, for computing the probability that, given two sets of **e** simultaneous data points that cover a distance lower than **t**, two sets of **(e+1)** simultaneous data points would also cover that distance lower than **t**.

Consequently, the parameters to be adjusted for the functioning of this process are:

- An **embedding dimension** value representing the number of points to consider in each set.
- A **tolerance** value as the estimated distance for each set of simultaneous data.
- Whether or not to **normalize** the data of the time series' values (which, depending on the nature of the data, may be necessary for obtaining a valid output from this measure).

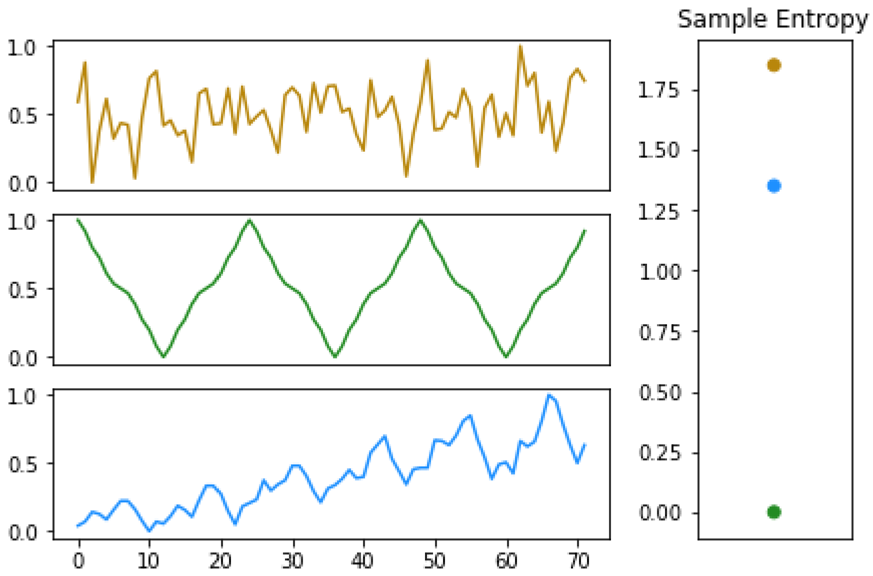


Figure 36. Expected behaviour of the Sample Entropy function given different types of input

4.2.2.1.1.2. Permutation Entropy

Permutation Entropy measures the variability of a time series behaviour. Given the ordinal pattern of sequential sets of **e** points (each spaced by **d** points from the other) in a time series, this metric is computed by assessing the behaviour of the series through all possible changes in each set's ordinal pattern.

The parameters associated to the functioning of this process are:

- An **embedding dimension** value representing the number of points to consider in each set.
- A **delay** value referred to the number of spacing points between sets

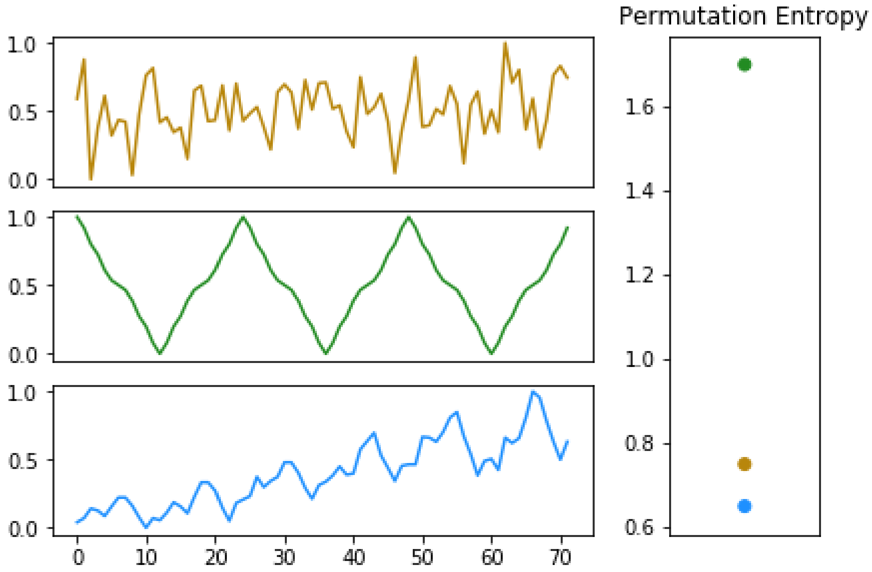


Figure 37. Expected behaviour of the Permutation Entropy function given different types of input

4.2.2.1.1.3. Irreversibility

The Irreversibility measure quantifies the variability in the behaviour of a time series when its sequence of timestamps is reversed. It is computed over sets of **e** data points given by an embedding dimension. Thus, the unique parameter on which adjustment relies the functioning of this process is:

- An **embedding dimension** value representing the number of points to consider in each set.

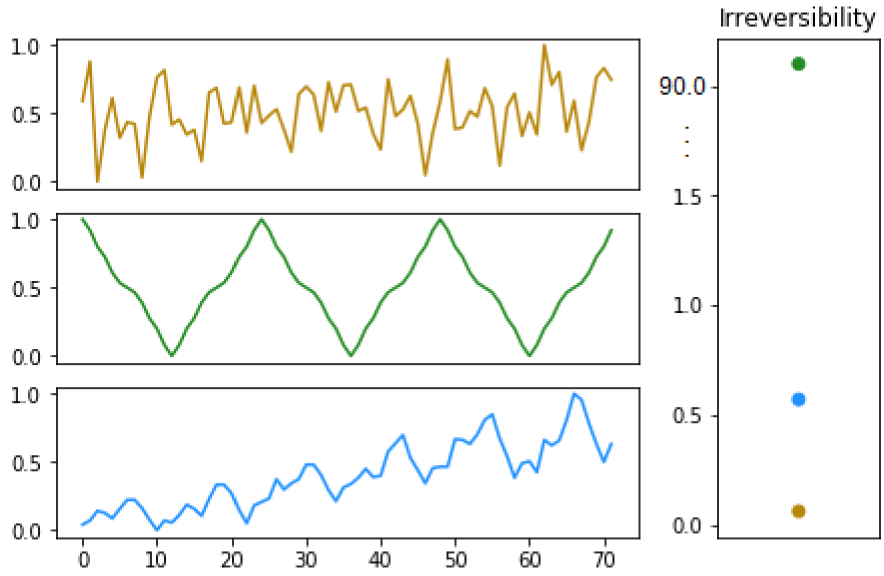


Figure 38. Expected behaviour of the Irreversibility function given different types of input

4.2.2.1.2. Value Prediction

Among the capabilities developed for the management of time series, there are functions aimed towards the prediction of their future values. These include data processing procedures with the following algorithms:

- Regression Trees
- Neural Networks
- ARIMA models

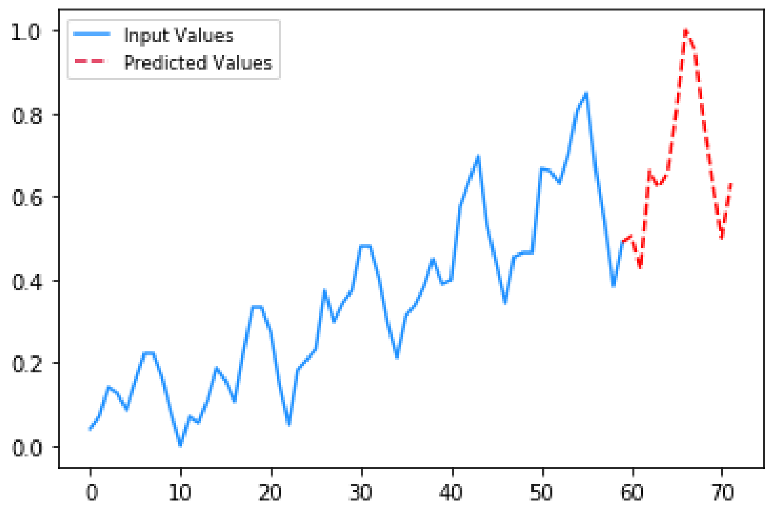


Figure 39. Prediction of a time series unknown data points

4.2.2.1.2.1. Regression Tree

The Regression Tree algorithm's mechanism emulates a flowchart where each node describes a test on the values from the assessed variable. This tests control whether or not a specific value from the series is greater or lower than a certain threshold computed by the algorithm. Coherently, each of the two branches that derive from each node represent a result of the test (lower or greater than the threshold).

New values are predicted based on the set of rules that emanate from the final tree's tests.

The parameters to be adjusted for the functioning of this process are:

- The number of data points to **forecast**.
- The index of the time series representing the point at which the **division** between training and test sets is to be performed.

4.2.2.1.2.2. Neural Network

A Multilayer Perceptron is trained and used for the prediction of new data points for a given time series.

The parameters to be adjusted for the functioning of this process are:

- The number of data points to **forecast**.
- The index of the time series representing the point at which the **division** between training and test sets is to be performed.

4.2.2.1.2.3. ARIMA

ARIMA is a time series forecasting algorithm based on the following principles:

- Autoregression (AR): uses the dependent relationship between an observation and some number of lagged observations.
  ARIMA assumes that the input data verifies that the values correspondent to a specific time depend of the values of previous data points.
- Integrated (I): differencing of observations to make the series stationary (same probability distribution for all data points).
- Moving Average (MA): uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

In coherence, the adjustable parameters associated to its functioning are

- The number of data points to **forecast**.
- A numeric value (**alpha**) for the statistical significance of the hypothesis test conducted within the model (0.05 is recommended as a widespread practice).
- The **order** of each ARIMA component. Namely, the components to adjust are:
  - **p**: related to AR's regular part. Number of lag observations included in the model.

- **d**: related to I's regular part. Number of times that the raw observations are differenced.
- **q**: related to Q's regular part. Size of the moving average window.
- **P**: related to AR's stationary part. Number of lag observations included in the model.
- **D**: related to I's stationary part. Number of times that the raw observations are differenced.
- **Q**: related to Q's stationary part. Size of the moving average window.

4.2.2.1.3. Correlation Statistics

Correlation coefficients are used to find how strong a relationship is between data. The formulas return a value between -1 and 1, where:

- 1 indicates a strong positive relationship.
- -1 indicates a strong negative relationship.
- A result of zero indicates no relationship at all.

The developed features include the calculation of such coefficients based on Pearson's and Spearman's correlation.

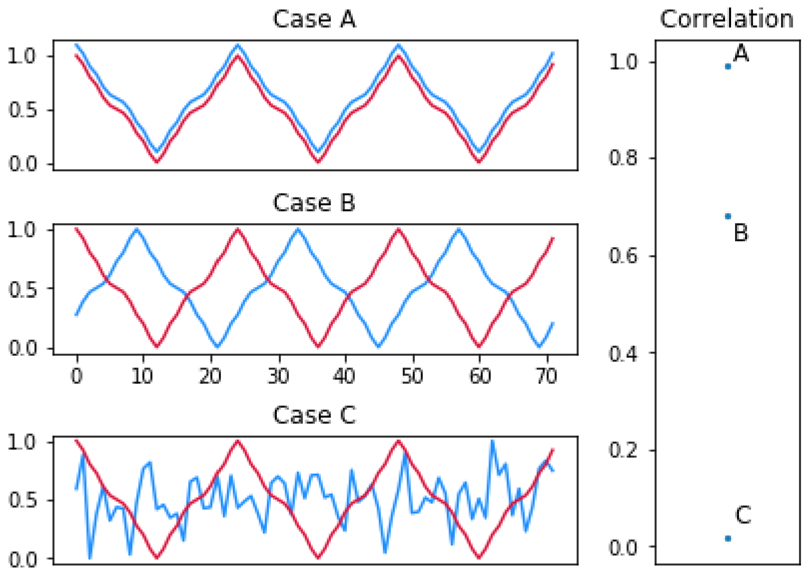


Figure 40. General expected behaviour of a correlation function

4.2.2.1.3.1. Pearson

Pearson correlation quantifies the strength of a linear relationship between paired data (the pair of time series which correlation is being computed).

It relies on assumptions of bot normality of distribution and homoscedasticity equal variance across variables) of the variables (time series) being evaluated.

4.2.2.1.3.2. Spearman

Spearman's correlation coefficient is a statistical measure of the strength of a monotonic relationship between paired data (the pair of time series which correlation is being computed).

Since it is a rank correlation measure (based on ordinality of samples), it does not rely on the assumptions of normality and homoscedasticity that Pearson's correlation does.

4.2.2.2. How to create a Time Series Analytics

The creation form can be found through de button in the Time Series list view:

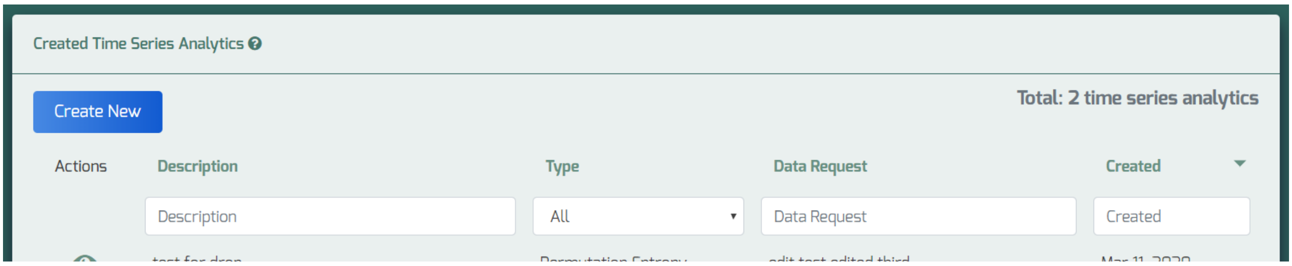


Figure 41. Create new Time Series Analytics button

Once inside a window with the form resume is shown in the top part of the view:

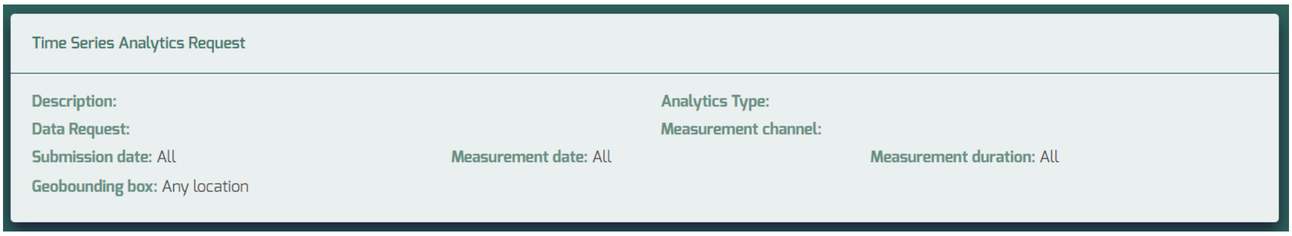


Figure 42. Time Series form resume

The first step here is to select the type of analysis to be performed. The options are displayed in the button list, with a brief explanation of the analysis selected shown at the right.

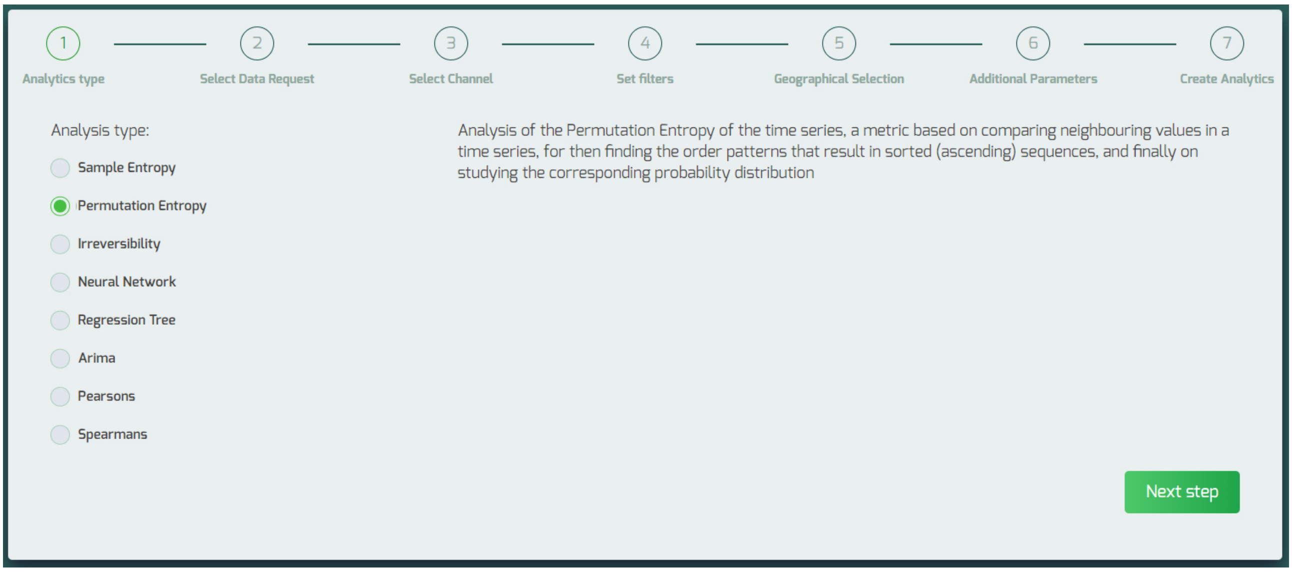


Figure 43. Create Time Series analytics - select type

In order to perform analytics on a set of data, the Service Provider needs an active data request that has been approved for analytics by the CPP owner.

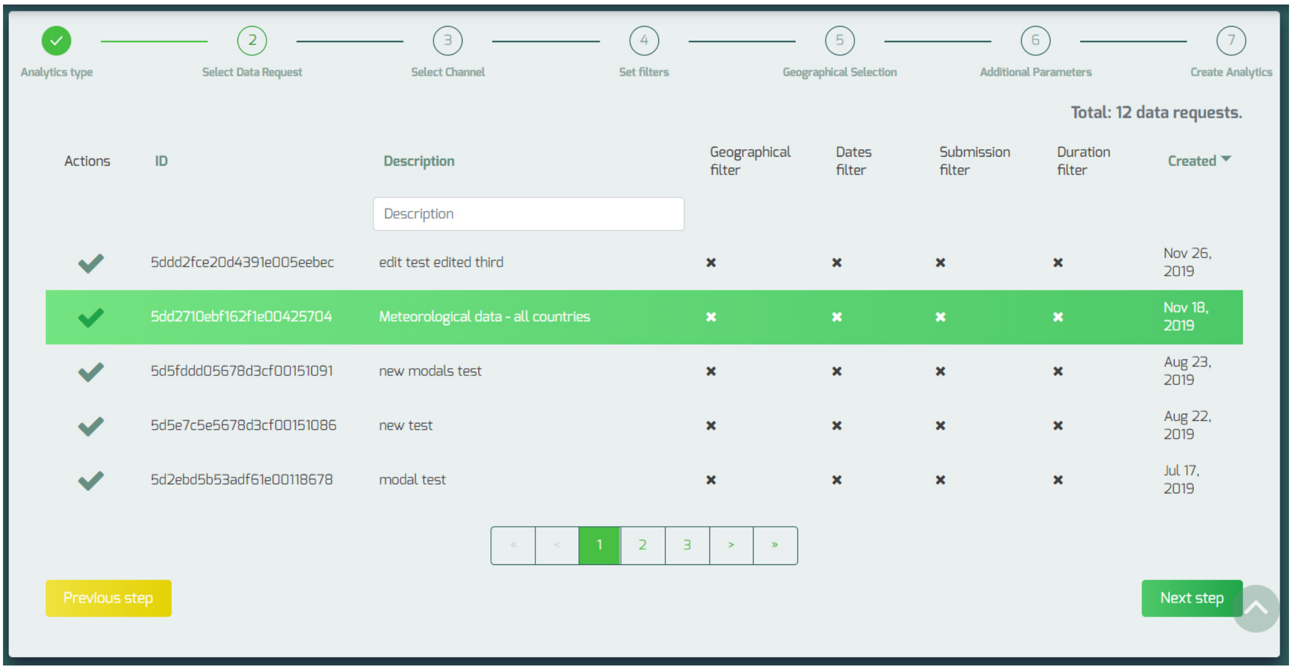


Figure 44. Create Time Series analytics - select Data Request

Every Data Request comprises one or more measurement channels from which the request receives data. Next step is to decide from which of those measurement channels data analysis is desired.

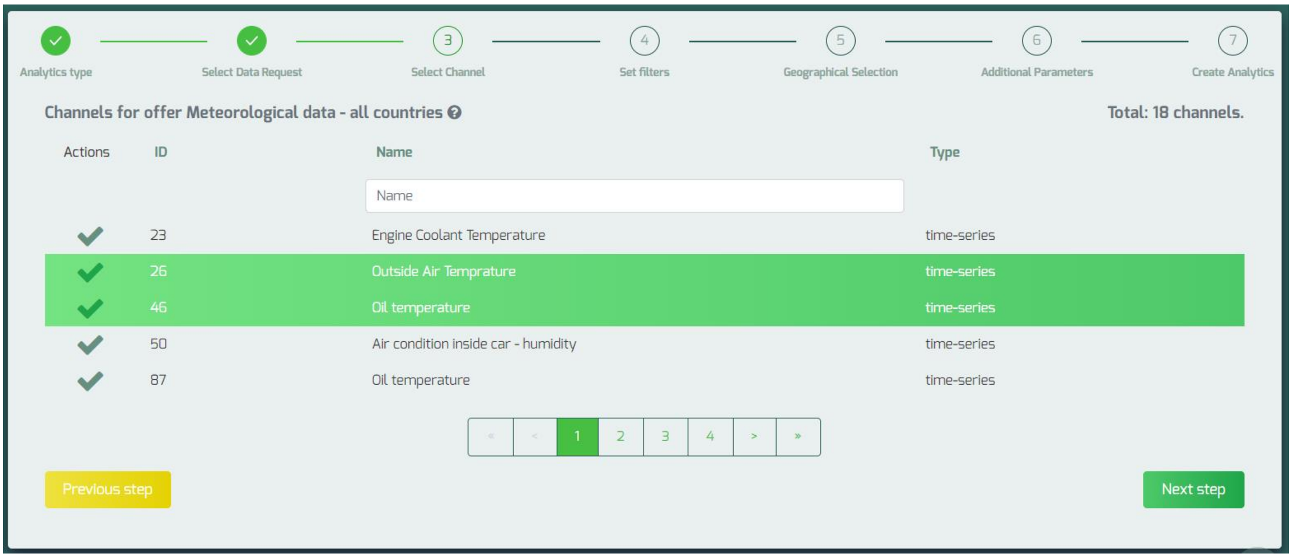


Figure 45. Create Time Series analytics - select measurement channels

From the selected Data Request initial filters (if any) data can be further filtered:

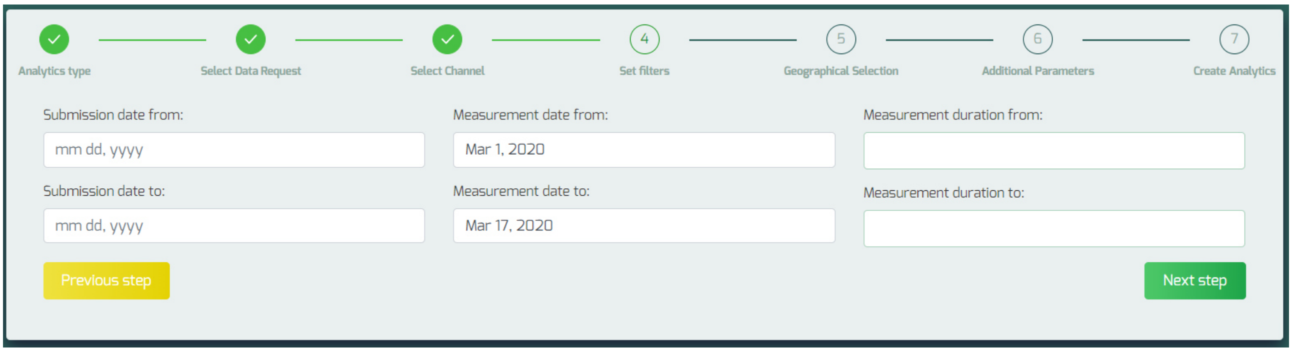


Figure 46. Create Time Series analytics - select filters

From the selected Data Request initial geographical bounding (if any) data can be further filtered:

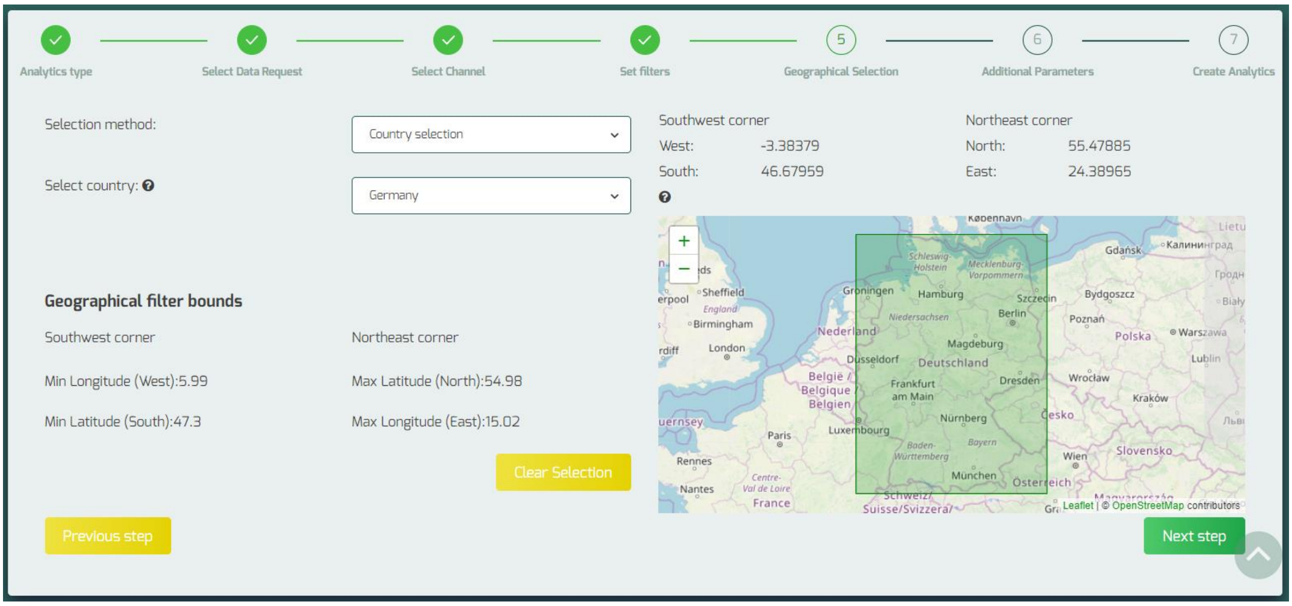


Figure 47. Create Time Series analytics - select geographical filter

Depending on the Time Series type additional parameters can be requested. A brief explanation of each parameter can be found in the question marks next to each of them:

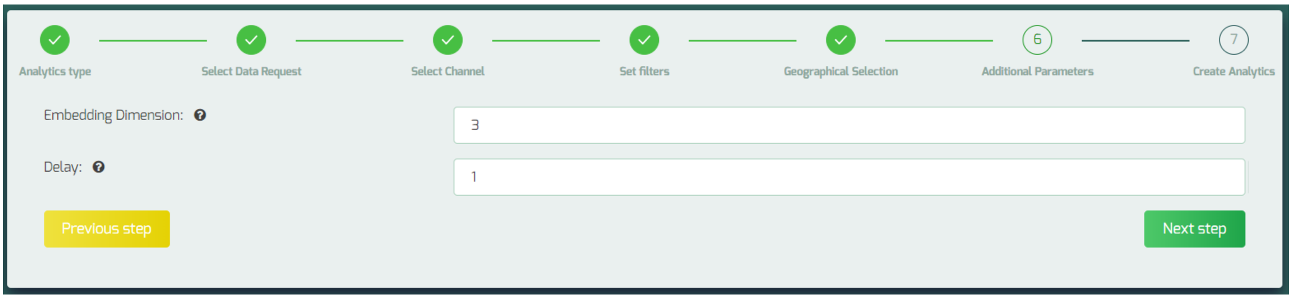


Figure 48. Create Time Series analytics - additional parameters

Finally, a description for the analytics is requested so it can be found easily by the user:

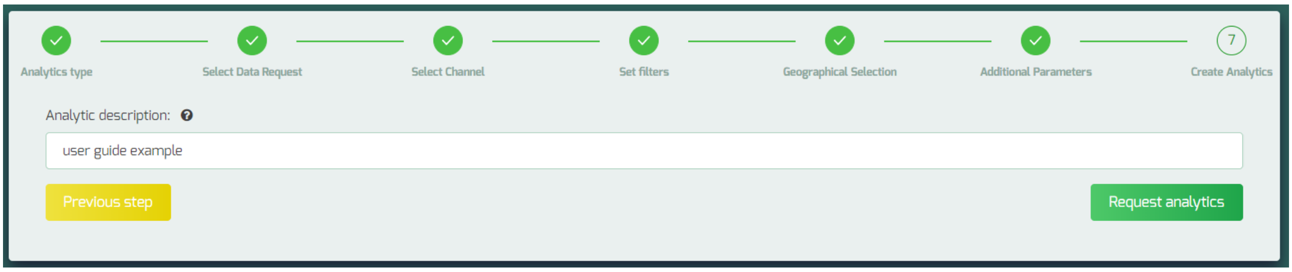


Figure 49. Create Time Series analytics - insert description

If successful, a summary of the created analytics will be displayed:

Figure 50. Create Time Series analytics - final resume

4.2.2.3. Review and stop a Time Series Analytics

Just clicking in the Time Series menu will lead to the list view, showing a list of all Time Series analytics created by the Service Provider.

In this list the user can filter the displayed results by description, analytics type, Data Request name or date.

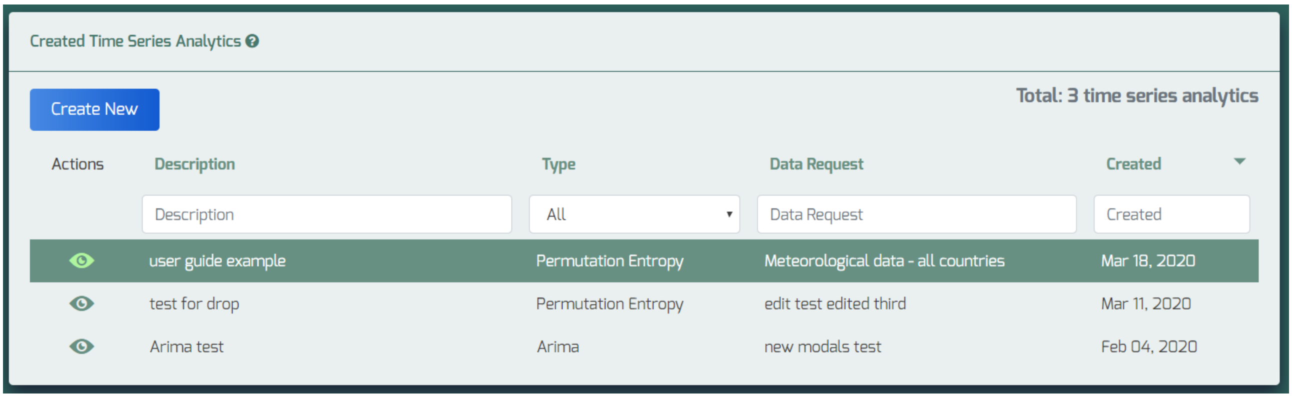


Figure 51. Created Time Series analytics list

Clicking on any of them will open the details view of that analytics, displaying the detailed configuration, including description, type and the AEON channel to subscribe to receive the analysed data.

There is also a “Delete Analytics” button. By clicking and confirming this button the Service Provider will stop receiving this analytics data and the configuration will be deleted..

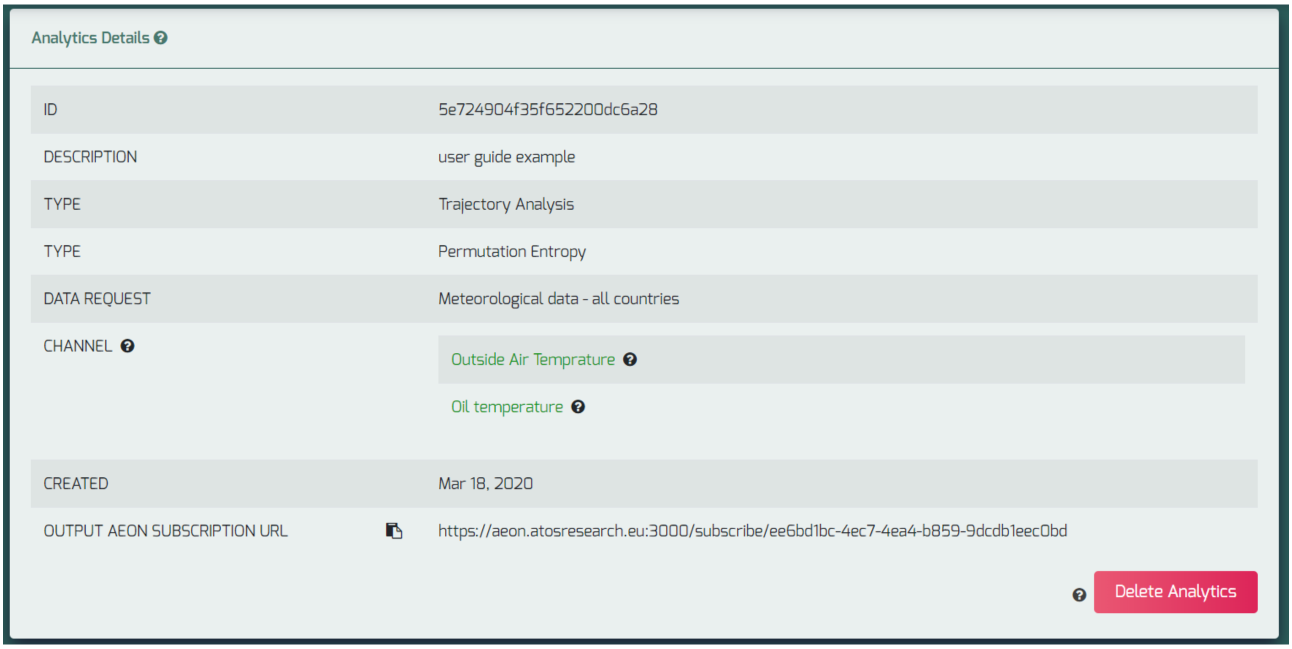


Figure 52. Time Series analytics configuration details

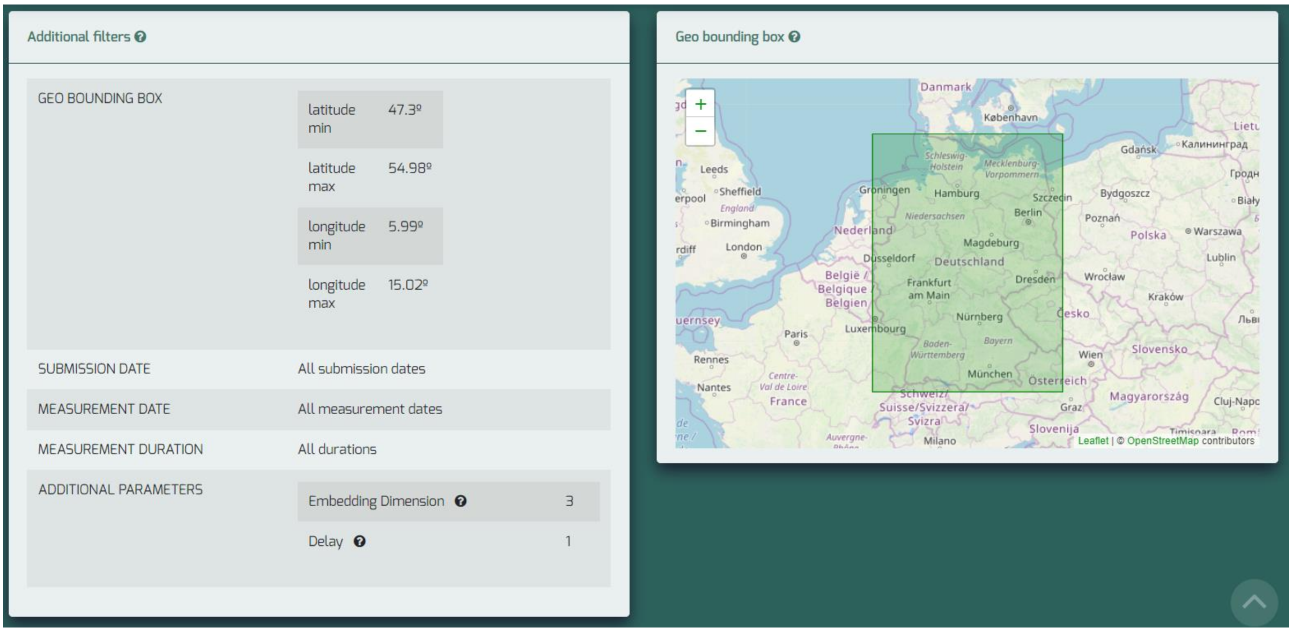


Figure 53. Time Series analytics filter details

4.2.3. Trajectory Analysis

Trajectory Analysis, understood as the management and treatment of GPS data (X-Y coordinates in which respects to this project), implies the necessity to operate with trajectories in different ways that enable the understanding of their main characteristics. With this purpose, the set of functionalities developed for the fulfilment of such requirements allow the user to evaluate the behaviour and properties of trajectories in different ways, including:

- Computation of simple statistics associated to a trajectory.
- Clustering of similar trajectories.
- Linear interpolation of a trajectory's GPS coordinates.
- Detection of anomalous coordinates within a trajectory.

These services are described in detail in the subsequent sections of this document.

4.2.3.1. Available analytics functions services and examples

4.2.3.1.1. Statistics

The statistics functionality provides a series of summarizing properties for a given trajectory, namely the total distance traveled in meters, its duration in seconds and its average velocity in meters per second.

The user needs to provide the trajectories for which to obtain these metrics, with no additional adjustable fields.

As noted in the previous description, there is only one requirement for this process' functioning:

- The coordinates of the trajectory to be resampled as pairs of (x, y) coordinates. Each of these points must have a timestamp associated indicating the date and time of occurrence.

4.2.3.1.2. Clustering

The clustering functionality enables the grouping of similar trajectories. Neighboring trajectories are labeled as belonging to the same cluster based on a metric of pair-wise distance between coordinates.

Since the adjustment of the parameters intrinsic to the algorithm is automatic, the user only needs to provide the trajectories to be grouped, with no additional adjustable fields.

In the example below (Figure 54) the left graphic represents the original scenario given as input to the algorithm, with the correspondent cluster centroids identified in the right part of the image.

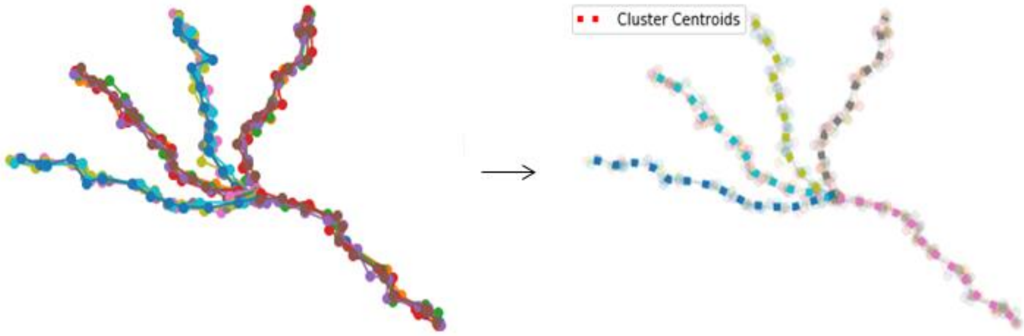


Figure 54. Example of a clustered trajectories.

With respect to this process' input, and as noted in the previous description, there is only one requirement for its functioning:

- The coordinates of the trajectories to be grouped as pairs of (x, y) coordinates. Each of these points must have a timestamp associated indicating the date and time of occurrence.

4.2.3.1.3. Interpolation

The interpolation functionality provides a mean of resampling a given trajectory. The type of interpolation performed is linear.

By providing a desired time resolution (in seconds) the user is able to set the frequency of appearance of each point in the new resampled trajectory.

In the example below (Figure 55) the solid blue circles represent the original coordinates, while the orange crosses are the resampled trajectory points (for a given time resolution).

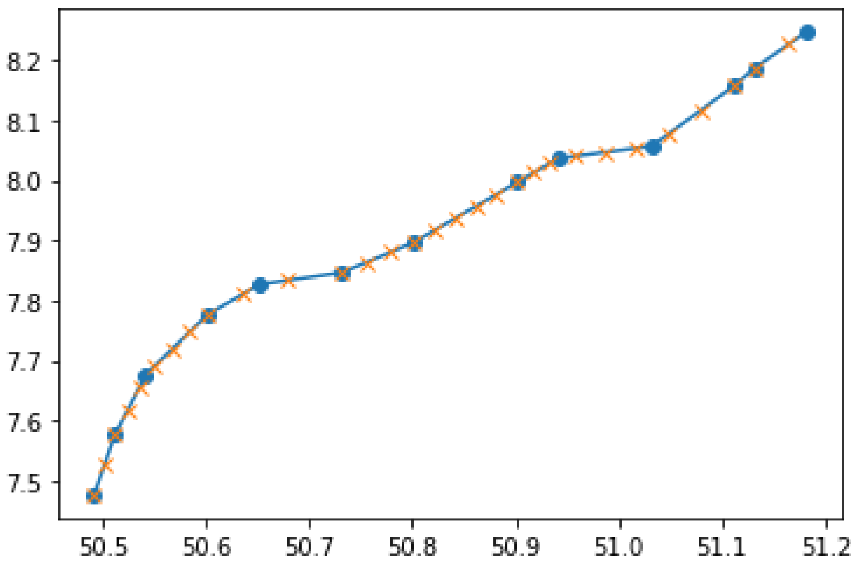


Figure 55. Example of a resampled trajectory.

The parameters required for the functioning of this process are:

- The coordinates of the trajectory to be resampled as pairs of (x, y) coordinates. Each of these points must have a timestamp associated indicating the date and time of occurrence.
- The time resolution (in seconds) to which the trajectory is intended to be resampled as a numeric value. This value should fulfil the following conditions:
    - Be an integer.
    - Be greater than 0.
    - Be smaller than the time difference between the initial and last coordinates' timestamps.

4.2.3.1.4. Anomaly detection

The anomaly detection functionality allows to identify abnormal points in a given trajectory. After associating the input trajectory to a cluster from the system's stored information, it compares the pair-wise distance between each of its coordinates and the group-centroid's. If this distance surpasses a certain threshold, the associated coordinates are marked as outlying.

The user has the option to change the type of threshold to apply in this process between a standard deviation protocol and an interquartile range one.

In the example below (Figure 56) the solid orange line and points represent the trajectory provided as input, with the outliers marked with black crosses.

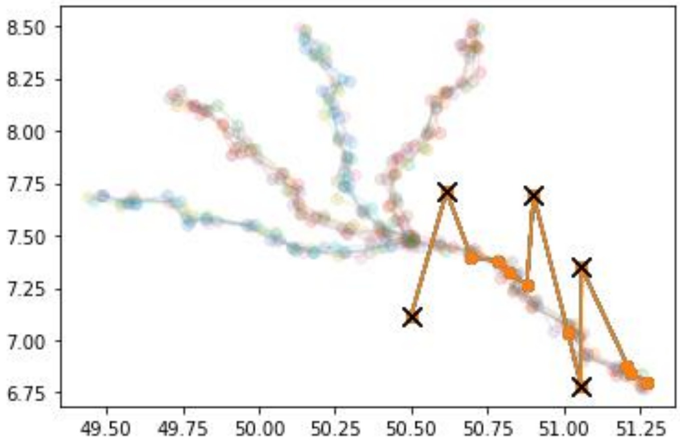


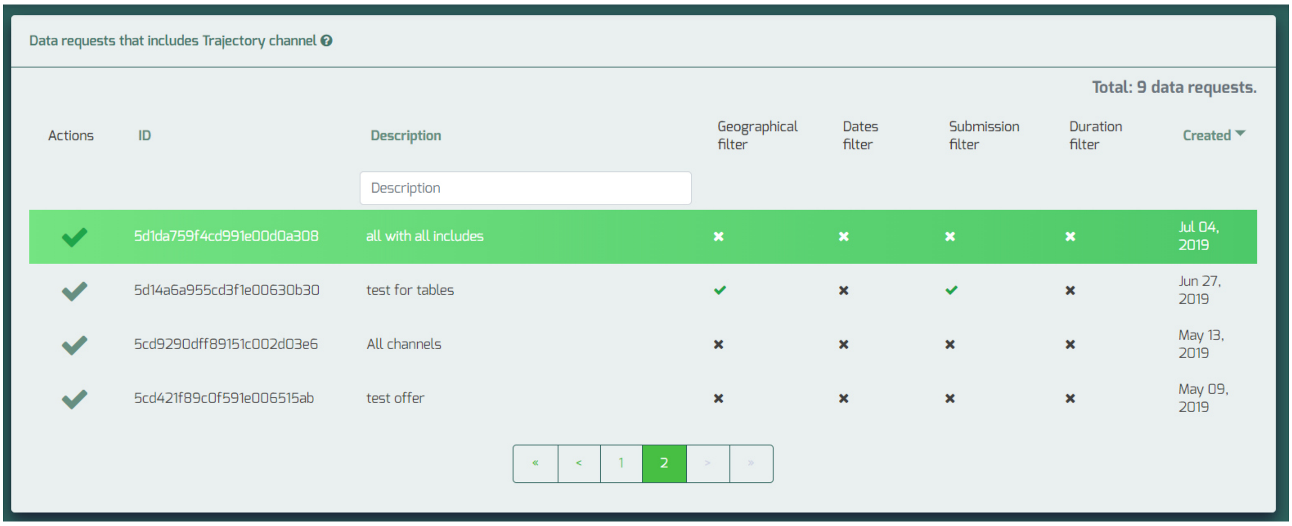
Figure 56. Example of positional anomalies' identification.

The parameters required for the functioning of this process are:

- The coordinates of the trajectory to be evaluated as pairs of (x, y) coordinates. Each of these points must have a timestamp associated indicating the date and time of occurrence.
- Optionally, the threshold method to apply for the identification of outliers. If the user does not provide this parameter, then the standard deviation protocol is applied by default. The available threshold metrics are standard deviation and interquartile range.

4.2.3.2. How to create a Trajectory Analysis

In order to perform analytics on a set of data, the Service Provider needs an active data request that has been approved for analytics by the CPP owner. Notice that only Data Requests containing measurement channel Position will be eligible, as that channel is the only one that includes the necessary data.


Data requests that includes Trajectory channel

Total: 9 data requests.

Actions	ID	Description	Geographical filter	Dates filter	Submission filter	Duration filter	Created
✓	5d1da759f4cd991e00d0a308	all with all includes	✗	✗	✗	✗	Jul 04, 2019
✓	5d14a6a955cd3f1e00630b30	test for tables	✓	✗	✓	✗	Jun 27, 2019
✓	5cd9290dff89151c002d03e6	All channels	✗	✗	✗	✗	May 13, 2019
✓	5cd421f89c0f591e006515ab	test offer	✗	✗	✗	✗	May 09, 2019

« ‹ 1 2 › »

Figure 57. Create Trajectory Analysis - select Data Request

From the selected Data Request initial filters and geographical bounding (if any) data can be further filtered:

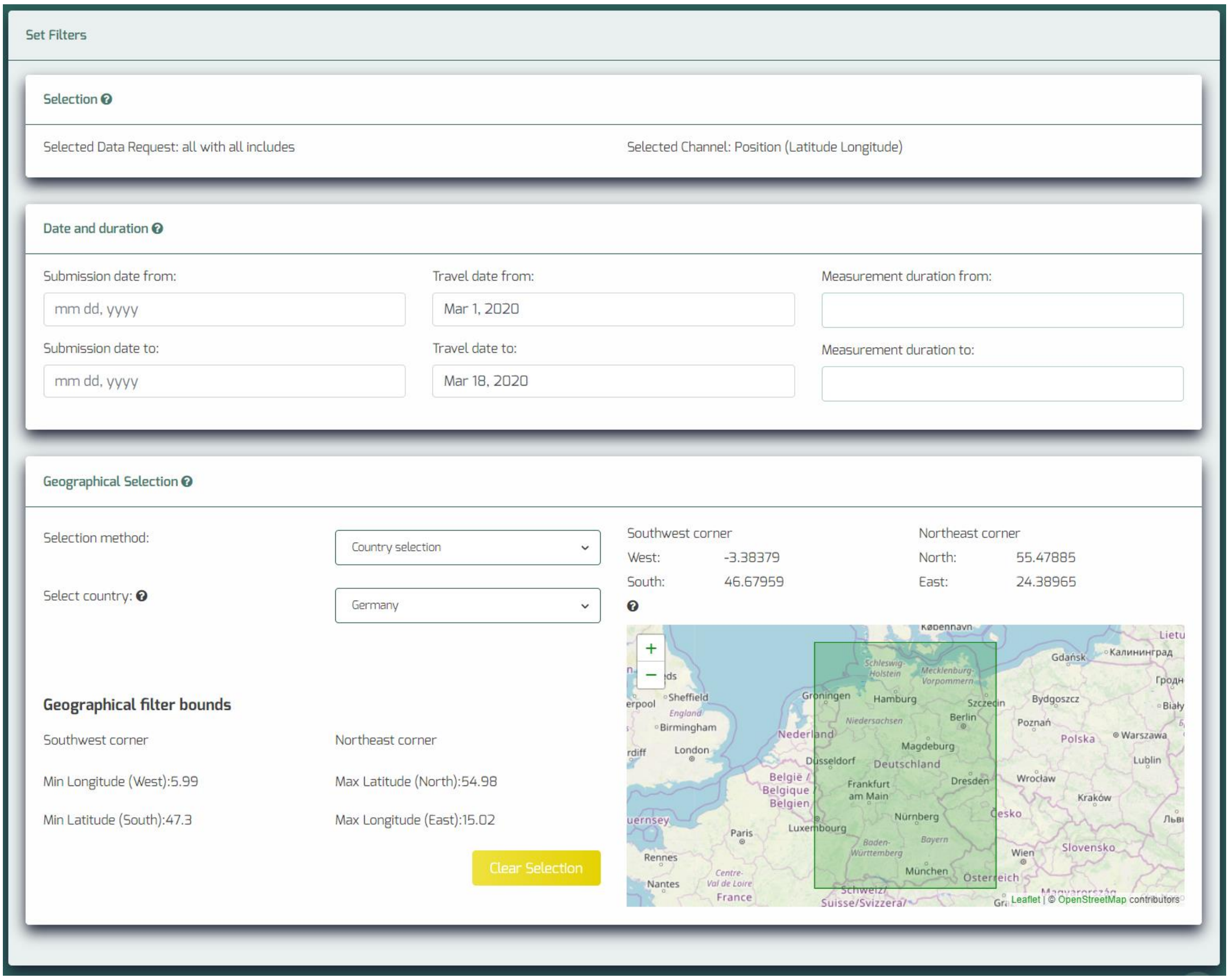


Figure 58. Create Trajectory Analysis - select filters

Next user has to select which type of analysis is desired to be performed in the trajectories contained in the data selected.

Depending on the analysis type additional parameters can be requested. A brief explanation of each parameter can be found in the question marks next to each of them:

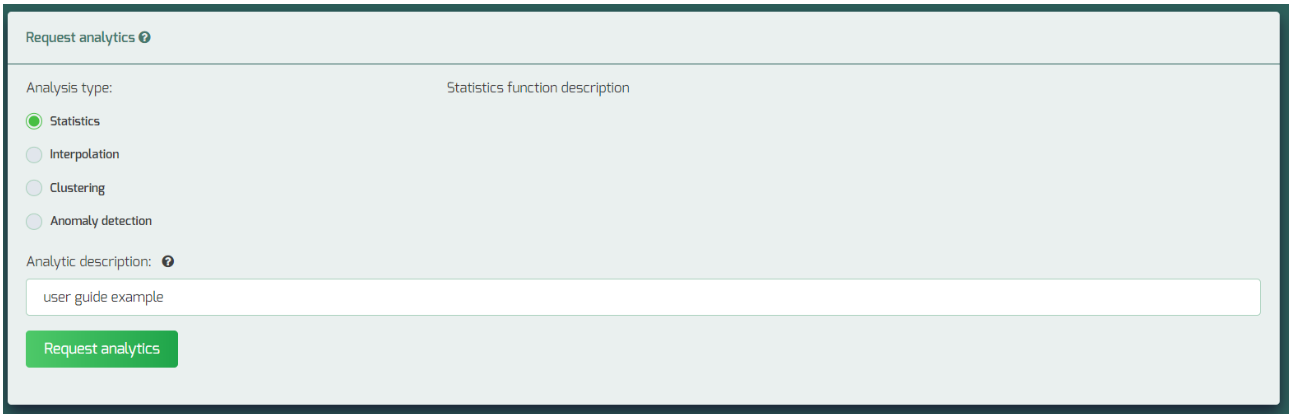


Figure 59. Create Trajectory Analysis - additional parameters

Finally, after labelling and describing the analysis, a message will be displayed leading to the trajectories list if the analysis was successful:

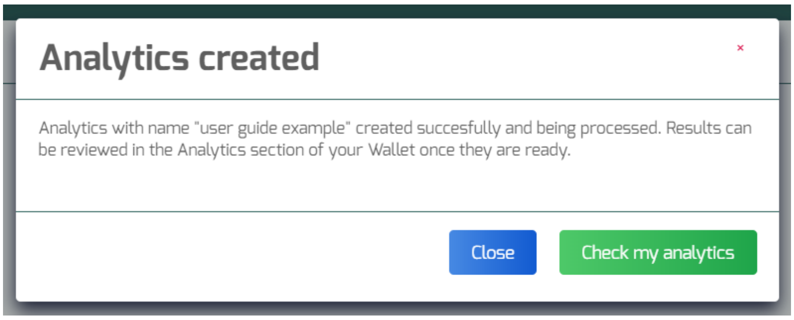


Figure 60. Create Trajectory Analysis - insert description

4.2.3.3. Review Trajectory Analysis results

Just clicking on the Trajectories menu will lead to the list view, showing a list of all Trajectory Analysis created by the Service Provider.

In this list users can filter the displayed results by description, analytics type, Data Request name or date.

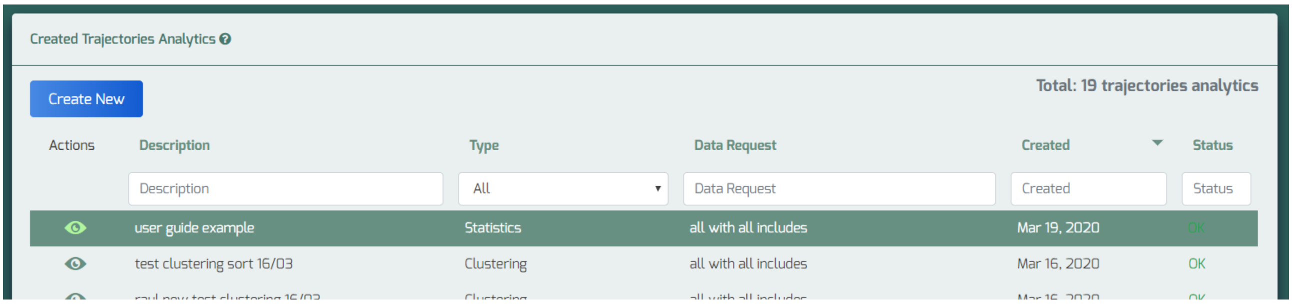


Figure 61. Created Trajectory Analysis list

Clicking on any of them will open the details view of that analytics, showing us the detailed configuration, including description, type and the results of the analysed data.

Analytics Details

ID	5e6f5c63b352112200bfcf1d
DESCRIPTION	raul new test statistics 16/03
TYPE	Trajectory Analysis
TYPE	Statistics
STATUS	OK
DATA REQUEST	all with all includes
CHANNEL	Position (latitude longitude)
CREATED	Mar 16, 2020
TRAVEL DATE	min Jan 1, 2018 max Mar 15, 2020
GEO BOUNDING BOX	latitude min 35.95º latitude max 43.75º longitude min -9.39º longitude max 4.3º

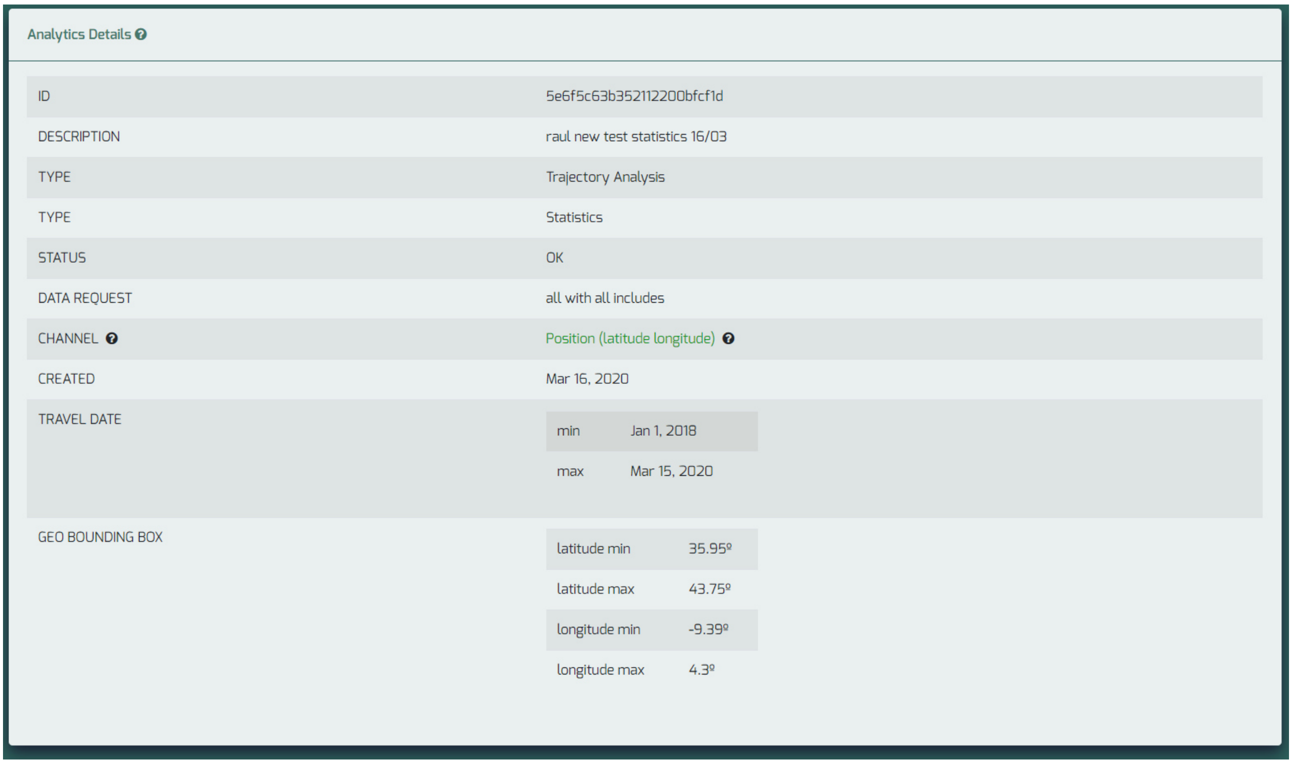


Figure 62. Trajectory Analysis configuration details

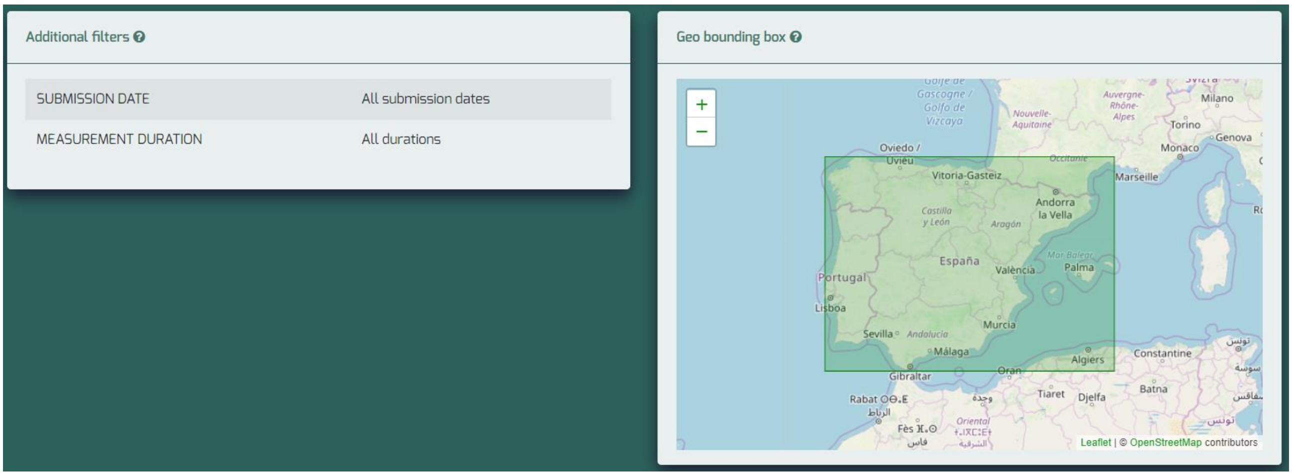
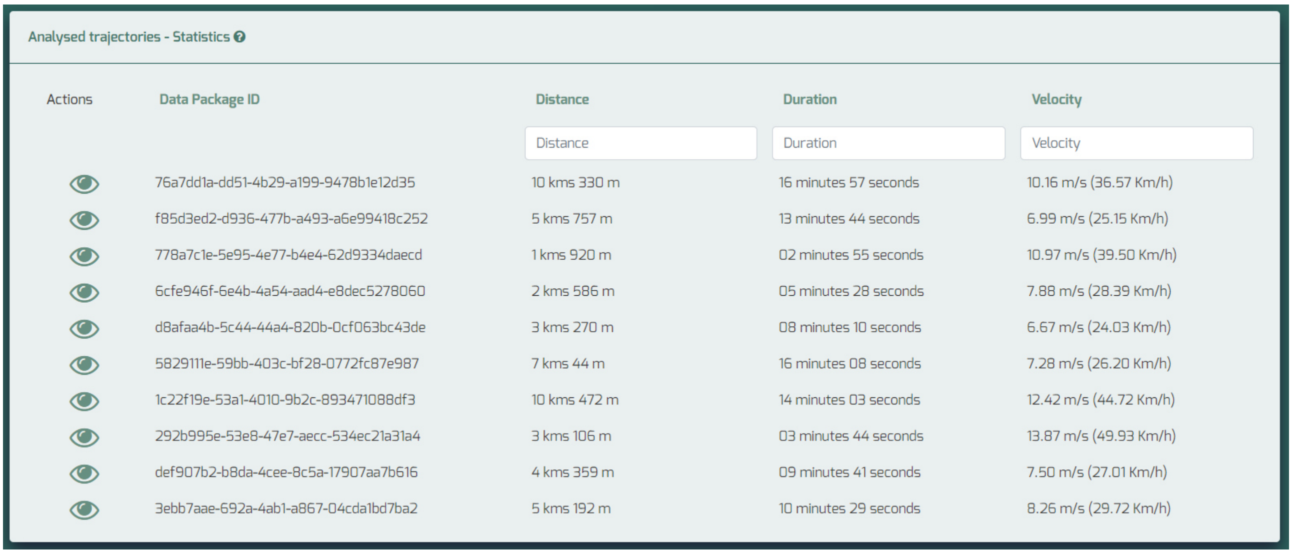


Figure 63. Trajectory Analysis filter details

Each type of Trajectory Analysis has its own kind of results and way to be displayed.

4.2.3.3.1. Statistics

First a table with all analysed trajectories is shown (Figure 64). Selecting one will display another box containing the trajectory statistics and map representation (Figure 65).



Analysed trajectories - Statistics

Actions	Data Package ID	Distance	Duration	Velocity
	76a7dd1a-dd51-4b29-a199-9478b1e12d35	10 kms 330 m	16 minutes 57 seconds	10.16 m/s (36.57 Km/h)
	f85d3ed2-d936-477b-a493-a6e99418c252	5 kms 757 m	13 minutes 44 seconds	6.99 m/s (25.15 Km/h)
	778a7c1e-5e95-4e77-b4e4-62d9334daecd	1 kms 920 m	02 minutes 55 seconds	10.97 m/s (39.50 Km/h)
	6cfe946f-6e4b-4a54-aad4-e8dec5278060	2 kms 586 m	05 minutes 28 seconds	7.88 m/s (28.39 Km/h)
	d8afaa4b-5c44-44a4-820b-0cf063bc43de	3 kms 270 m	08 minutes 10 seconds	6.67 m/s (24.03 Km/h)
	5829111e-59bb-403c-bf28-0772fc87e987	7 kms 44 m	16 minutes 08 seconds	7.28 m/s (26.20 Km/h)
	1c22f19e-53a1-4010-9b2c-893471088df3	10 kms 472 m	14 minutes 03 seconds	12.42 m/s (44.72 Km/h)
	292b995e-53e8-47e7-aecc-534ec21a31a4	3 kms 106 m	03 minutes 44 seconds	13.87 m/s (49.93 Km/h)
	def907b2-b8da-4cee-8c5a-17907aa7b616	4 kms 359 m	09 minutes 41 seconds	7.50 m/s (27.01 Km/h)
	3ebb7aae-692a-4ab1-a867-04cda1bd7ba2	5 kms 192 m	10 minutes 29 seconds	8.26 m/s (29.72 Km/h)

Figure 64. Trajectory Analysis - statistics results list

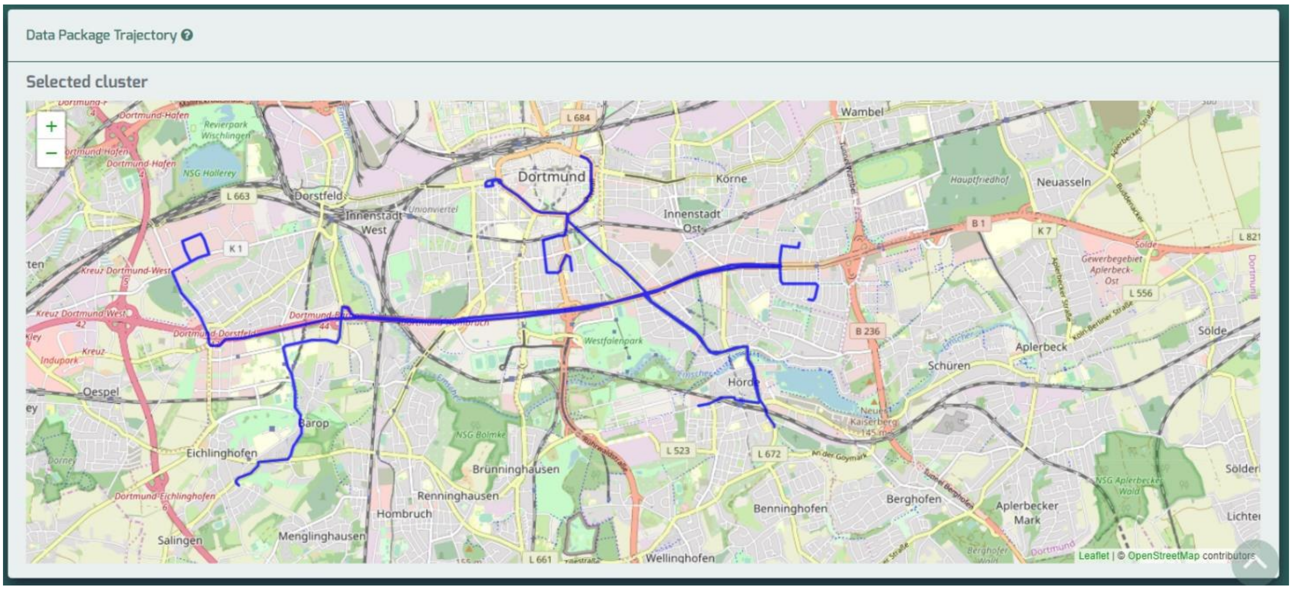


Figure 65. Trajectory Analysis - statistics results display

4.2.3.3.2. Clustering

First a table with resulting clusters from all analysed trajectories is shown (Figure 66). Selecting one will display another box containing the trajectory statistics and map representation (Figure 67).

Analysed trajectories - Clustering

Actions	Cluster Nº	DP Amount	Data Packages
	0	5	76a7dd1a-dd51-4b29-a199-9478b1e12d35,f85d3ed2-d936-477b-a493-a6e99418c252,d8afaa4b-5c44-44a4-820b-0cf063bc43de,1c22f19e-53a1-4010-9b2c-893471088df3,292b995e-53e8-47e7-aecc-534ec21a31a4
	1	1	6cfe946f-6e4b-4a54-aad4-e8dec5278060
	2	1	5829111e-59bb-403c-bf28-0772fc87e987
	3	3	778a7c1e-5e95-4e77-b4e4-62d9334daecd,def907b2-b8da-4cee-8c5a-17907aa7b616,3ebb7aae-692a-4ab1-a867-04cda1bd7ba2

Figure 66. Trajectory Analysis - clustering results list

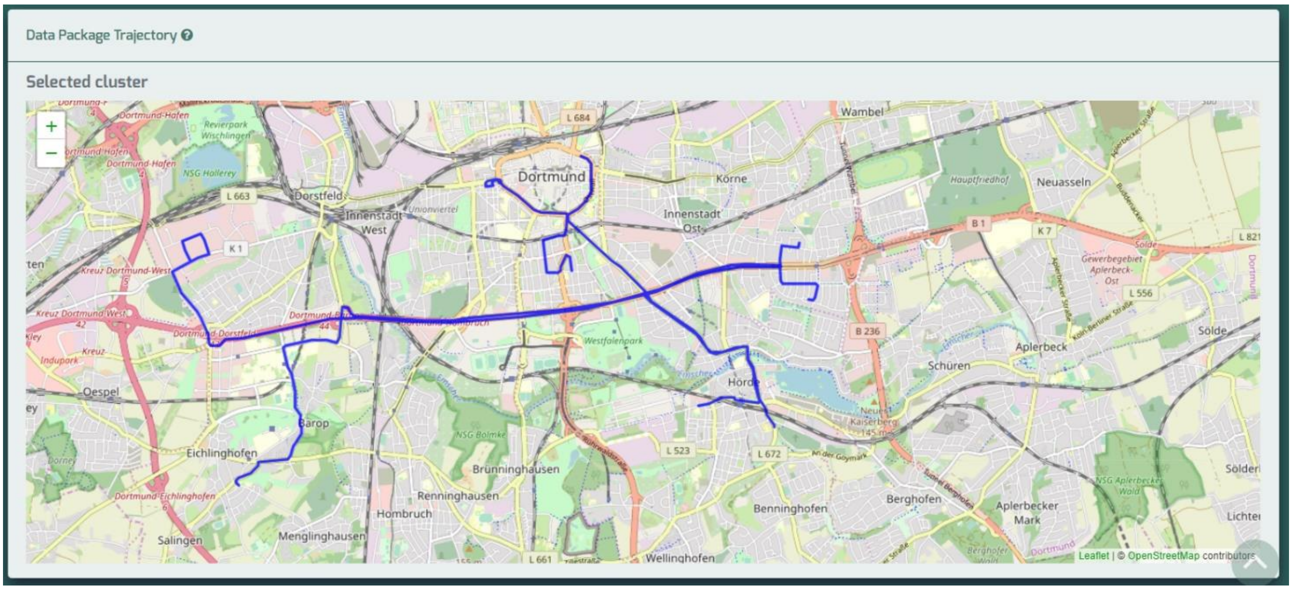
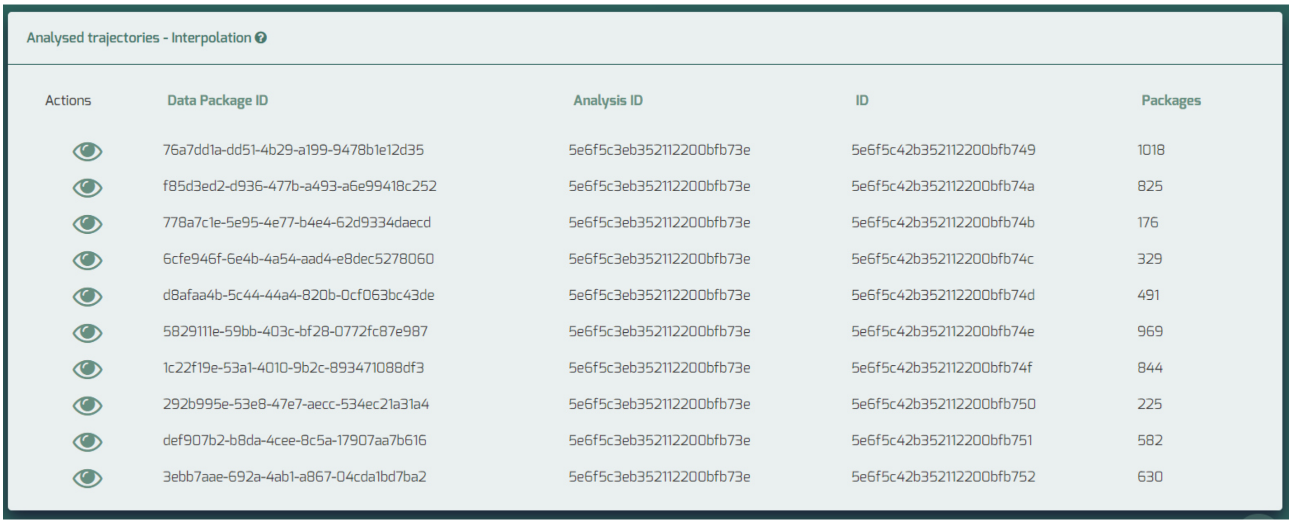


Figure 67. Trajectory Analysis - clustering results display

4.2.3.3.3. Interpolation

First a table with all interpolated trajectories is shown (Figure 68). Selecting one will display another box containing map representations of both the original trajectory and the resulting one from the interpolation function (Figure 69).



Analysed trajectories - Interpolation

Actions	Data Package ID	Analysis ID	ID	Packages
	76a7dd1a-dd51-4b29-a199-9478b1e12d35	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb749	1018
	f85d3ed2-d936-477b-a493-a6e99418c252	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb74a	825
	778a7c1e-5e95-4e77-b4e4-62d9334daecd	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb74b	176
	6cfe946f-6e4b-4a54-aad4-e8dec5278060	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb74c	329
	d8afaa4b-5c44-44a4-820b-0cf063bc43de	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb74d	491
	5829111e-59bb-403c-bf28-0772fc87e987	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb74e	969
	1c22f19e-53a1-4010-9b2c-893471088df3	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb74f	844
	292b995e-53e8-47e7-aecc-534ec21a31a4	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb750	225
	def907b2-b8da-4cee-8c5a-17907aa7b616	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb751	582
	3ebb7aae-692a-4ab1-a867-04cda1bd7ba2	5e6f5c3eb352112200bfb73e	5e6f5c42b352112200bfb752	630

Figure 68. Trajectory Analysis - interpolation results list

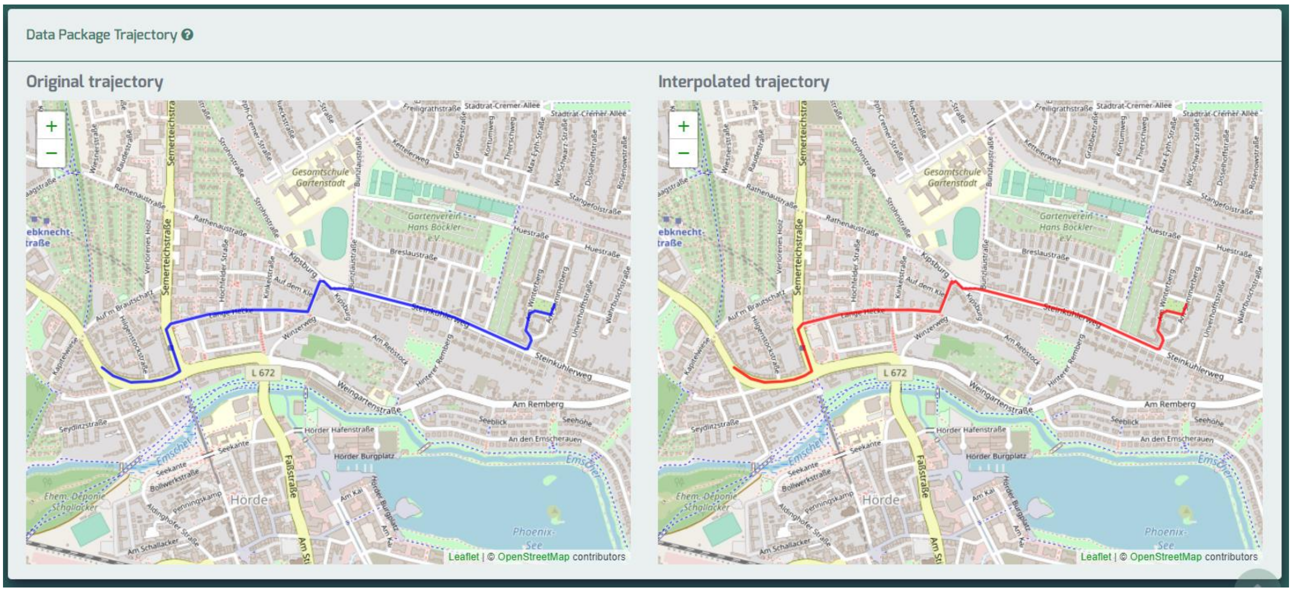


Figure 69. Trajectory Analysis - interpolation results display

4.2.3.3.4. Anomaly detection

First a table with all analysed trajectories is shown (Figure 70). Selecting one will display another box containing the map representation of the anomaly detection of the trajectory (Figure 71).

Analysed trajectories - Anomaly detection

Actions	Data Package ID	Analysis ID	ID	Geolocation points
	778a7c1e-5e95-4e77-b4e4-62d9334daecd	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f533	176
	6cfe946f-6e4b-4a54-aad4-e8dec5278060	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f534	329
	f85d3ed2-d936-477b-a493-a6e99418c252	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f532	825
	d8afaa4b-5c44-44a4-820b-0cf063bc43de	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f535	491
	76a7dd1a-dd51-4b29-a199-9478b1e12d35	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f531	1018
	5829111e-59bb-403c-bf28-0772fc87e987	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f536	969
	292b995e-53e8-47e7-aecc-534ec21a31a4	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f538	225
	def907b2-b8da-4cee-8c5a-17907aa7b616	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f539	582
	3ebb7aae-692a-4ab1-a867-04cda1bd7ba2	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f53a	630
	1c22f19e-53a1-4010-9b2c-893471088df3	5e8deeca40e3b62100e7f526	5e8deee640e3b62100e7f537	844

Figure 70. Trajectory Analysis – anomaly detection trajectories list

Data Package Trajectory

Analyzed trajectory

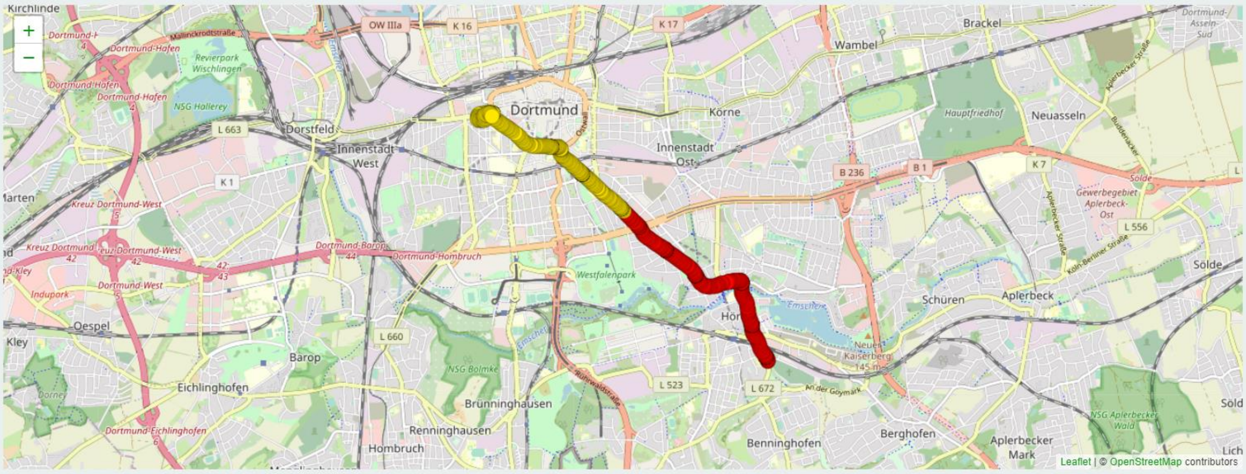


Figure 71. Trajectory Analysis – anomaly detection results display

4.2.4. Network Analysis

Networks can be understood as the graphic representation of relationships between entities. Each of these entities would be a node, connected to other similar ones by a link. The following figure depicts this concept.

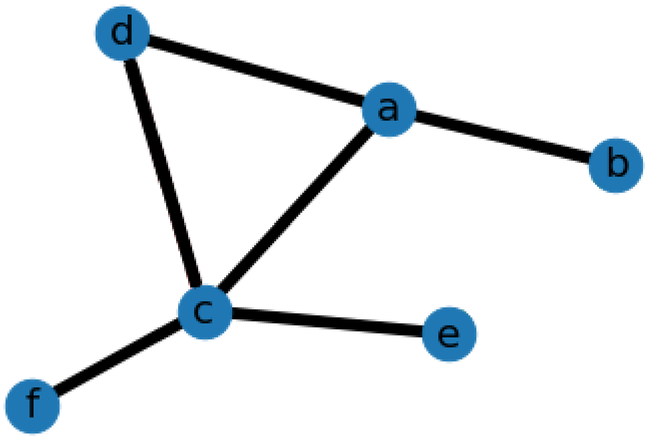


Figure 72. Networks - example of a simple network connecting entities (labeled with an alphabet letter)

In the CROSS-CPP ambit, a node would be a uni-dimensional signal from an available Measurement Channel (e.g. vehicle speed, temperature…). A link would simply be an indicator of the similarity between two signals (which is the reason why these links are called "weighted", because they are associated to a metric telling us how similar it is to its neighbour).

The specific shape of this type of graphs encloses information with respect to each specific node and its links (e.g. position with respect to other nodes, measurements of how outlying each node is…), as well as more general features referred to the arrangement of the network (e.g. how sparse or well-connected it is). This data can, in turn, be interpreted to provide the user with valuable insight with respect to the signals of the channel being analysed.

With this purpose in mind, the API described in this document provides a set of functionalities that enable the user to create and inspect a network's properties over time. Specifically, the developed functionalities allow to generate a network based on the available signals of a measurement channel. After the network is created, it will be continuously updated as new data from the analysed signals arrives, providing with updated information of their status.

Each of these services are described in detail in the forthcoming sections.

4.2.4.1. How to create a network

In order to perform analytics on a set of data, the Service Provider needs an active data request that has been approved for analytics by the CPP owner.

In the example the Data Request with name "all with all includes" is selected.

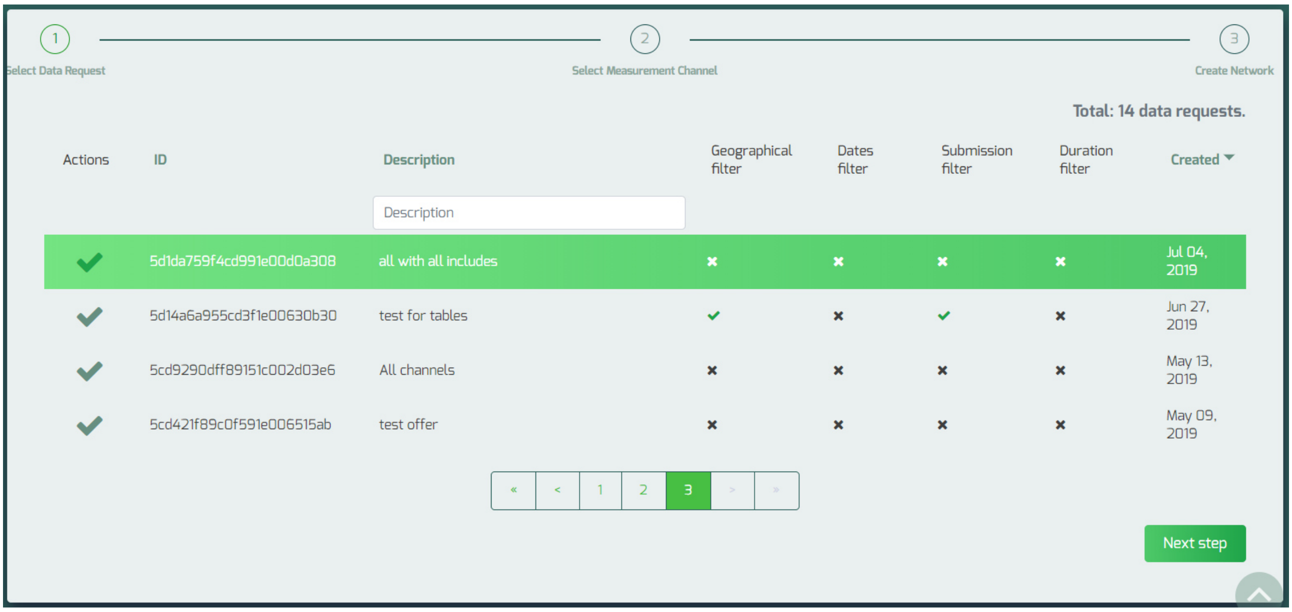


Figure 73. Create Network - select Data Request

Every Data Request comprises one or more measurement channels from which the request receives data. Next step is to decide from which of those measurement channels data analysis is desired. Every CPP in the network will conform a node with its value of the selected measurement channel.

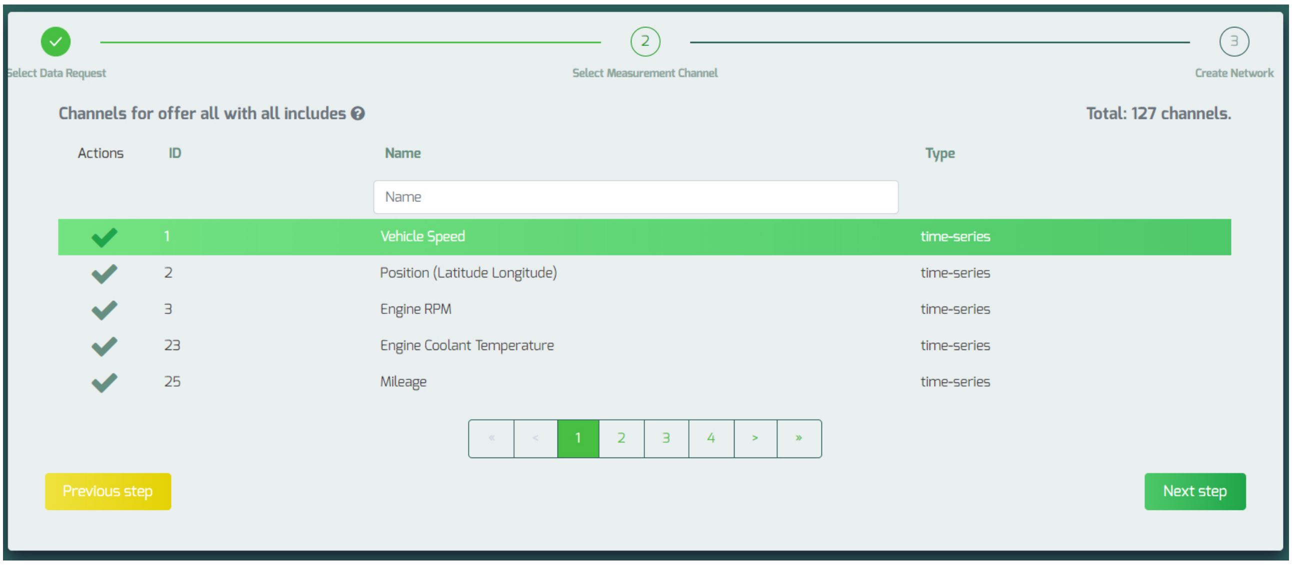


Figure 74. Create Network - select Measurement Channel

When selecting the Measurement Channel, a check of the network availability is performed with three possible outcomes.

Networks are unique, so if there is already an existing network for the selected Data Request and Measurement Channel a message will be shown.

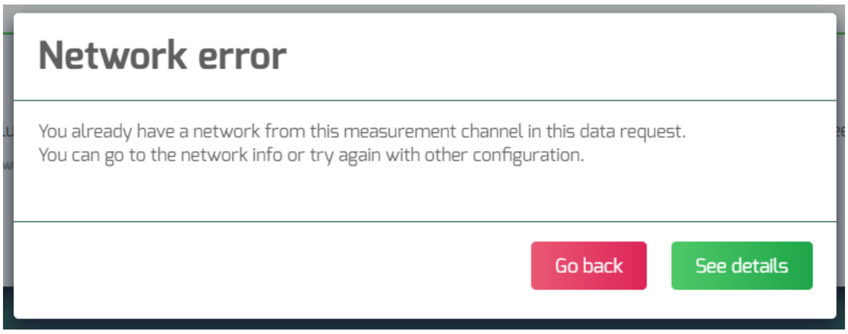


Figure 75. Create Network – Network already exists

If the Network does not exist but there are not enough available nodes a message will be displayed explaining it.

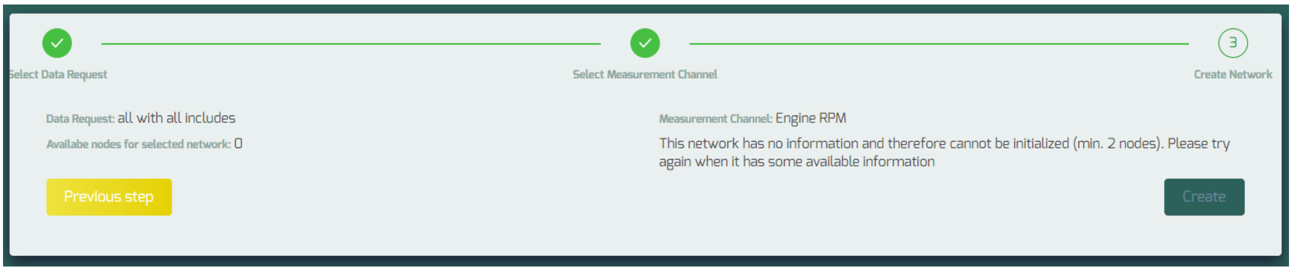


Figure 76. Create Network – no nodes available

If there is enough number of nodes creation will be enabled.

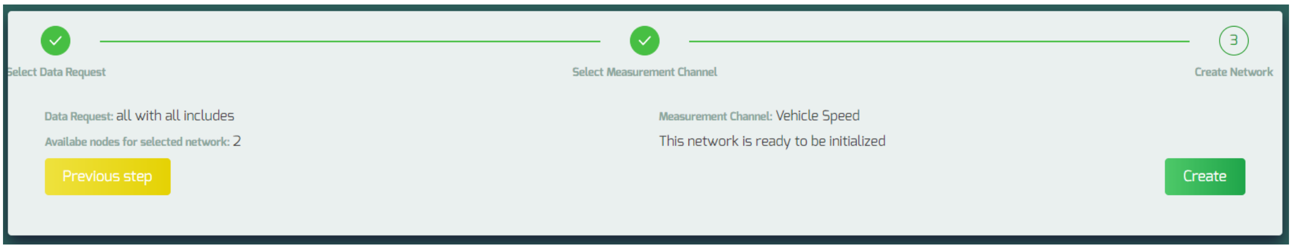


Figure 77. Create Network – Network ready

Once created a message will be displayed letting the user review the details and see the status of the network.

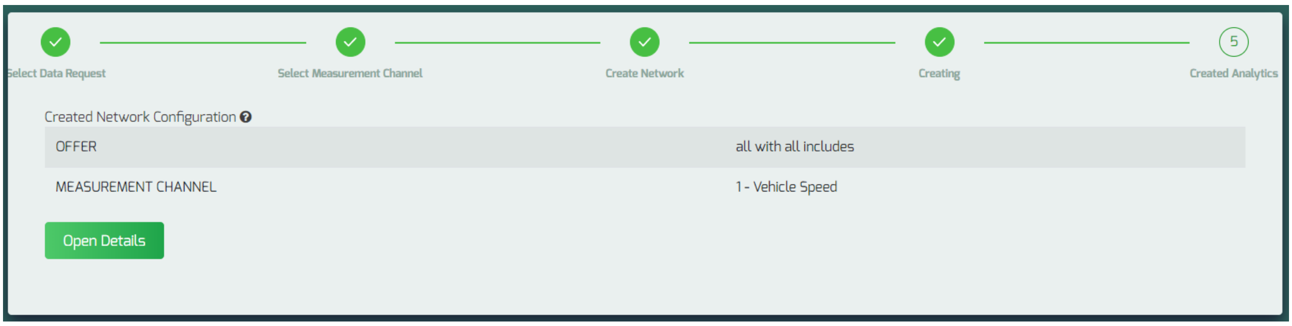


Figure 78. Create Network – Network created successfully

4.2.4.2. Get network status

Just clicking on the Networks menu will lead to the list view, showing a list of all Networks created by the Service Provider.

In this list users can filter displayed results by Data Request and Measurement Channel of the network.

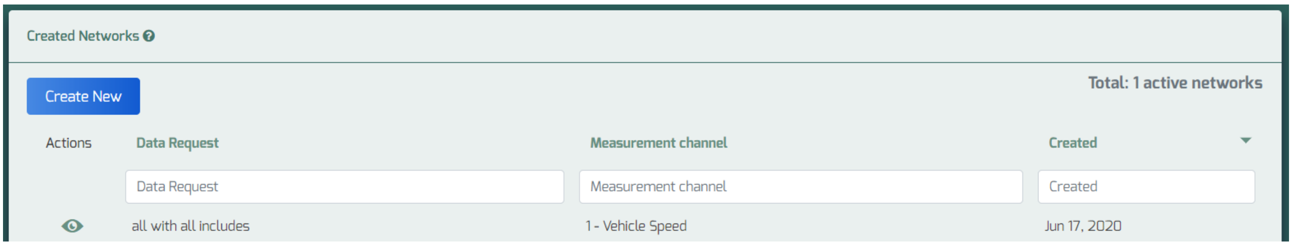


Figure 79. Networks list

Clicking on any of them will open the details view of that network, showing us the detailed configuration.

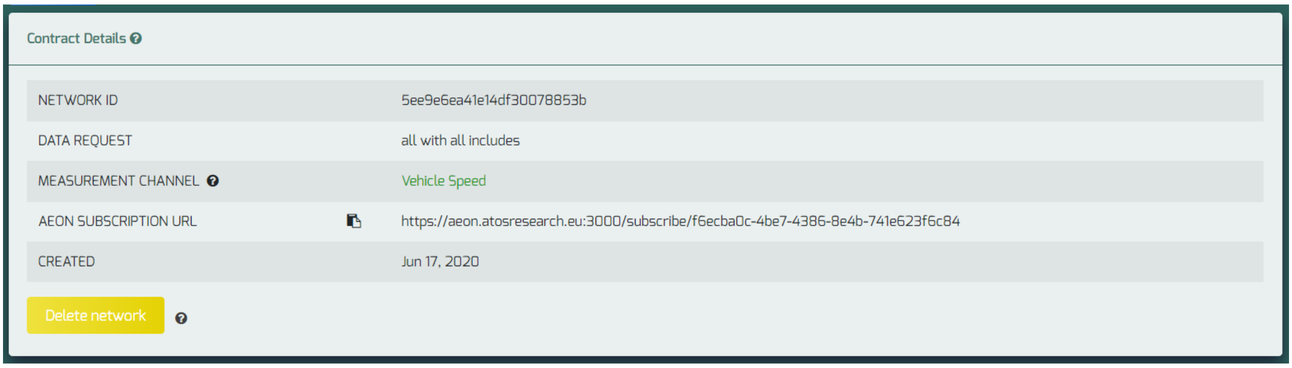


Figure 80. Network configuration details

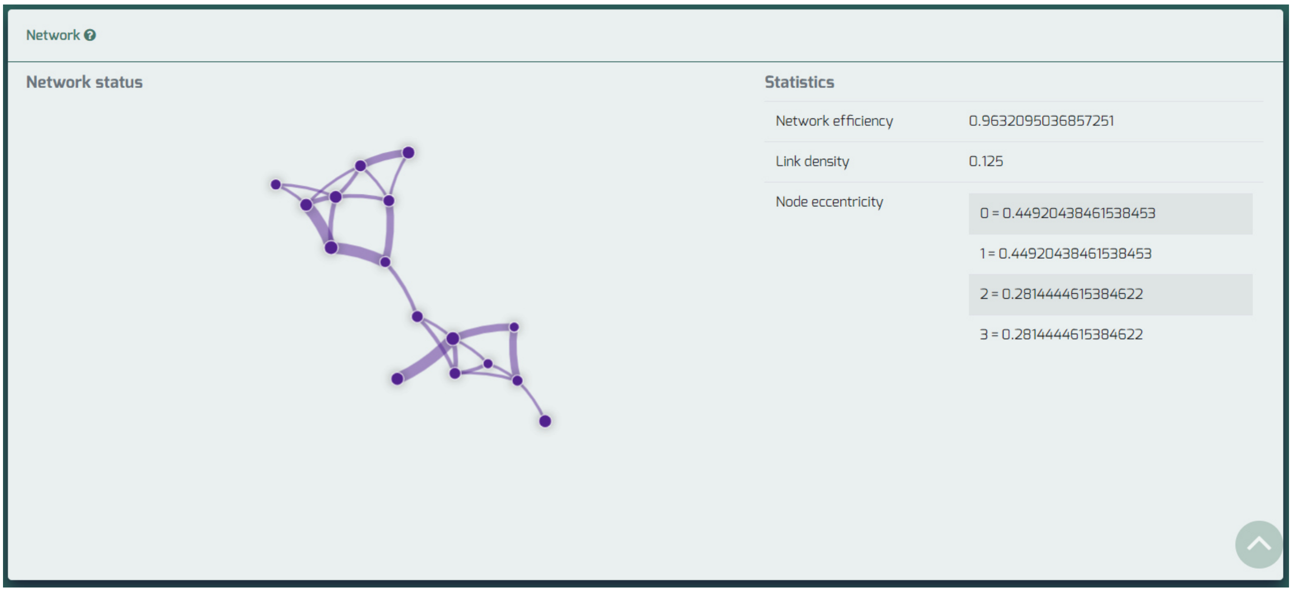


Figure 81. Network status

Network status include the following information:

- **Map of the network**: network representation on how nodes are connected.
- **Network Efficiency**: measure of how well-connected a network is. This metric is representative of the number of nodes that are connected between themselves by a direct link or through their connection to a path of nodes.
  Consequently, this metrics provides insight into how well connected the different groups present in a network are. This is, how similar are each group's signals between themselves (a desirable occurrence in well-formed networks). The higher it is, the more connections between nodes there are.
- **Link Density**: ratio expressing the amount of links present in the network with respect to its nodes. Homogeneous networks are typically link-dense, this is, they have a high amount of links connecting most of its nodes. A high value for this metric would be indicative of a network with its nodes closely related or, in other words, the corresponding signals being highly related.
  As this metric becomes smaller, the corresponding network is less well-connected and would thus imply the presence of separated groups of signals and/or outliers.
  This metric is generally directly correlated to network efficiency.
- **Node Eccentricity**: as an additional list containing the indices of the nodes from which to obtain the eccentricity. It can be thought of as a measure of how eccentric a node is. Thus, nodes with high eccentricity would represent a potentially outlying signal that may require further analysis or filtering.

4.2.5. Machine Learning

The last module of the Data Analytics Toolbox provides a generic interface for building advanced machine learning models from large data available in the Cross-CPP Data Marketplace. It interconnects the Marketplace with existing frameworks (currently supported are Scikit-learn, Tensorflow, and PyTorch).

Note, that the processing pipeline differs from that of other analytics components as the training phase can take a significant time for large data. Consequently, the endpoints mostly work in an asynchronous manner. For example, the invocation of the ML model building just initiates the process and returns an identifier of this task. The caller can check the status of the process regularly. When the model is ready, it can be applied to new data. The basic scheme of the interaction with the machine learning components is shown on the following figure.

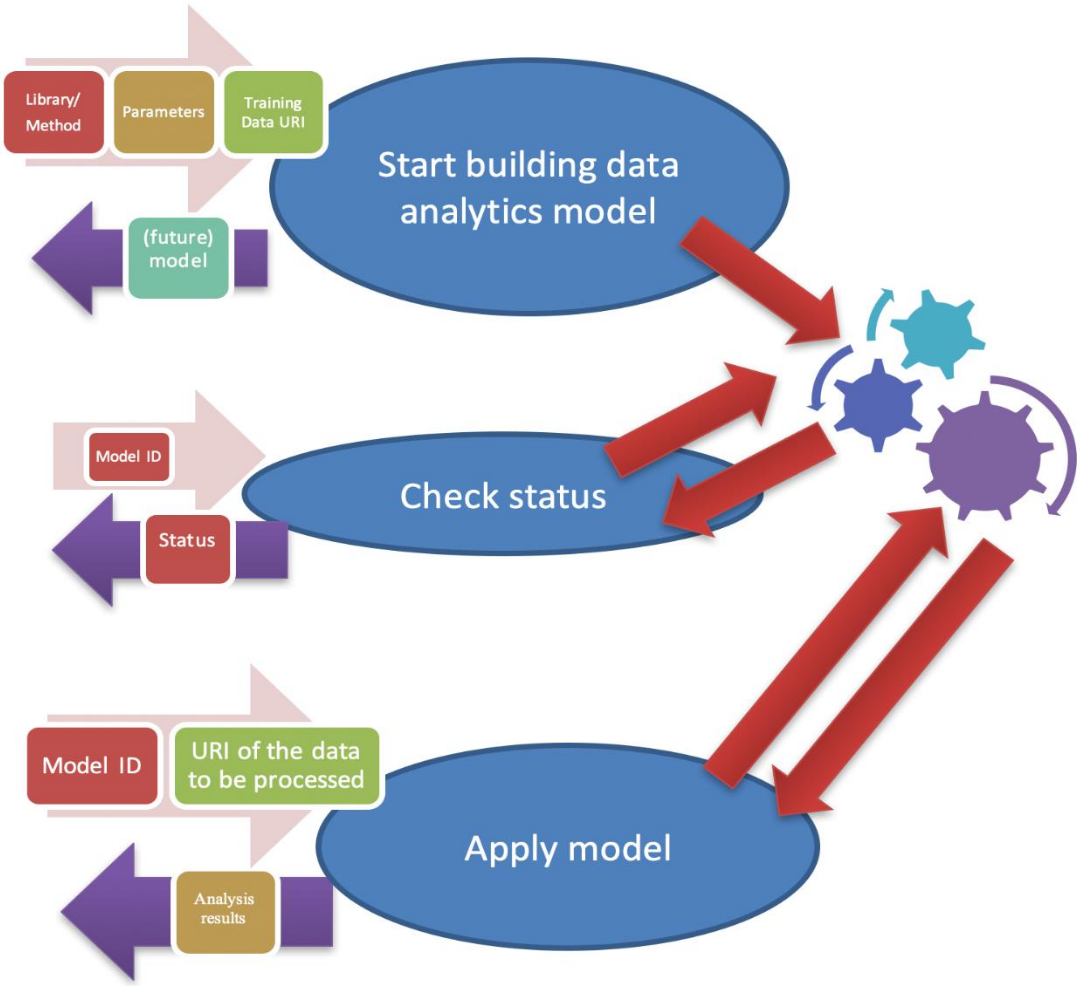


Figure 82. Machine Learning operation workflow

To link the scheme with the full workflow and functionality in the Marketplace, let us identify necessary components and actors in the process. A service provider interested in the use of the ML functionality identifies channels to be considered as inputs for the ML model first. In the example mentioned below, which classifies driver behavior patterns, the input channels consist of the speed, the location, and the rain intensity. The service provider defines a data view (see Section 6.4.1) and lists the selected channels in the call.

The machine learning functionality is employed to create a model from annotated data. The annotation of the data needs to be created in the next step. The service provider can add the expected category to each row (to each combination of input channel values stored) in the data view created in the previous step. This is done by invoking the endpoint /sei/category/create first and naming the category to be predicted by the future ML model (for example, driver_behaviour_ml_prediction_output in the mentioned example). Then, the actual category values for the data view rows (for example, normal/risky/dangerous) are assigned by the endpoint /sei/category/assign to as many input data as possible.

Having the annotated data in the form of the data view, the service provider invokes the ML model training (model creation/building) process by calling endpoint /ml_model/batch/build with the data view ID as a parameter. When the subsequent call of endpoint /ml_model/status indicates that the model is prepared and ready, the model can be used to classify new data.

The created model is used on new data (to be automatically classified) by invoking the /ml_model/batch/apply endpoint. The service provider can create a new data view with the same (input) channels as the data view used for the model training, which corresponds to the new data to be automatically classified. Alternatively, the original data view can be used and the endpoint ignores the manually added categories (the last column) in the view. The response to the /ml_model/batch/apply request contains the predicted category (e.g., normal/risky/dangerous) for the data view rows provided in the parameter.

Service providers can be also interested in evaluating the quality of the ML model in question. According to the machine learning methodology, this should be done on a separate set of annotated data – a test set containing data not used in the model training). In a standard setting, the user creates a new data view, manually adds categories (in the same way as described for the training data view discussed above) and invokes endpoint /ml_model/batch/evaluate. The response contains a score of the model on the given data view – an accuracy of the prediction compared to manually provided categories.

4.2.5.1. How to create a Machine Learning model

In order to create a Machine Learning model, the Service Provider needs an active Data View that has been assigned with a category, and that has at least 10 rows with an assigned value.

In the example the Data View with name "Temperature, humidity and wind speed to assess outside weather" is selected.

First we get the Select Data View step.

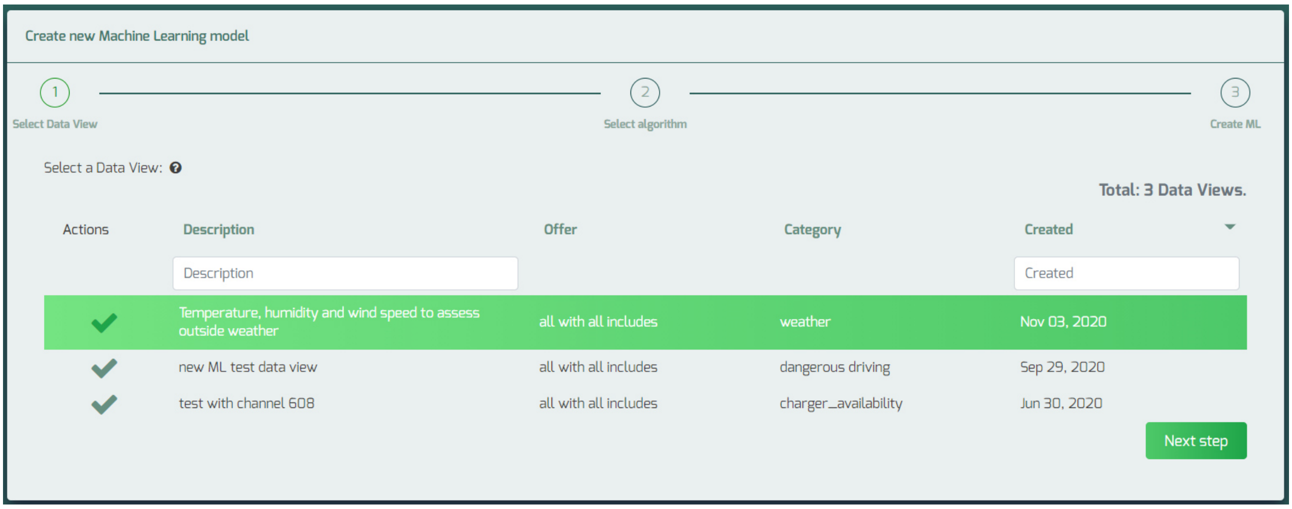


Figure 83. Create ML model - select Data View

Once a Data View is selected it is checked to know if it fulfil the necessary requirements mentioned before.

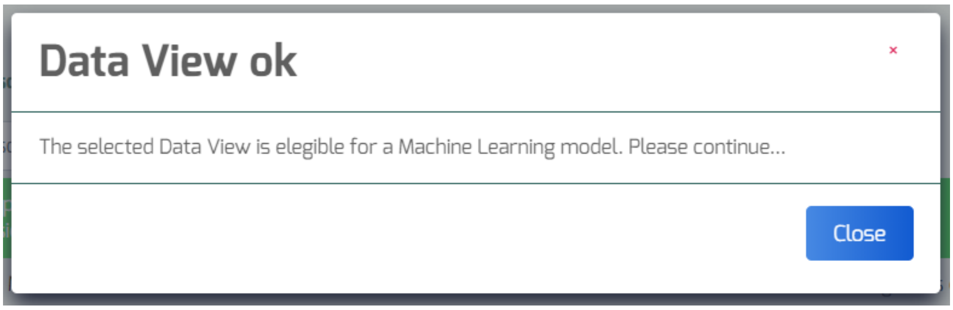


Figure 84. Create ML model - selected Data View ok

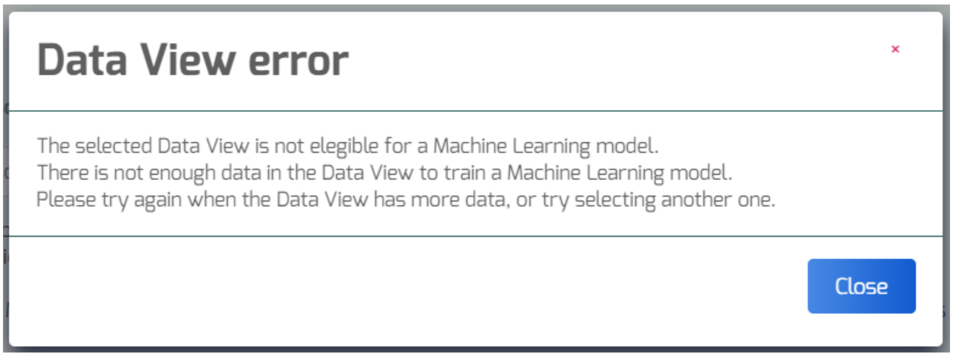


Figure 85. Create ML model - selected Data View not eligible

The next step is to select the type of algorithm to use when training the model with the selected Data View collected data.

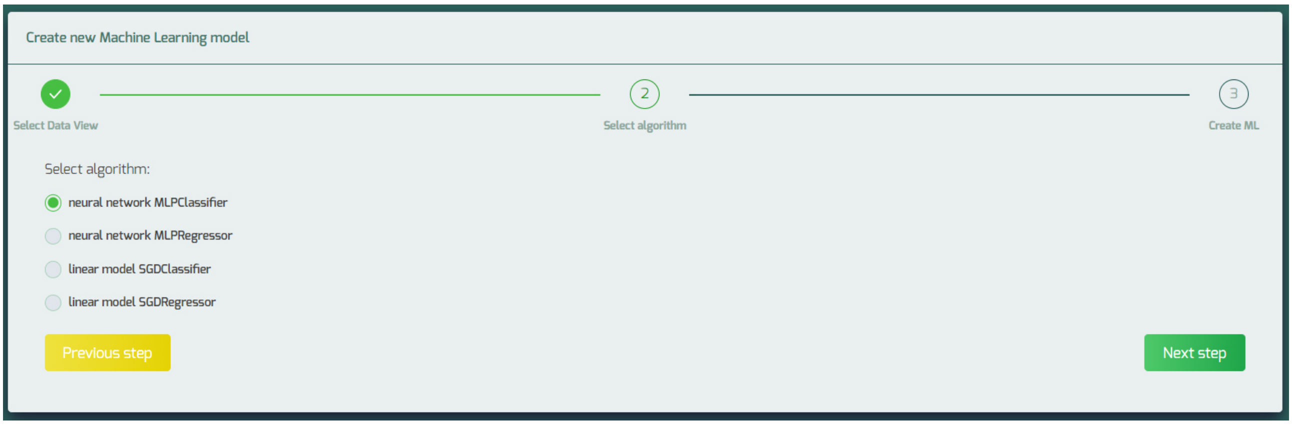


Figure 86. Create ML model – select algorithm

Finally, we get a quick review on the selections made and get the choice to create the model or return to change something.

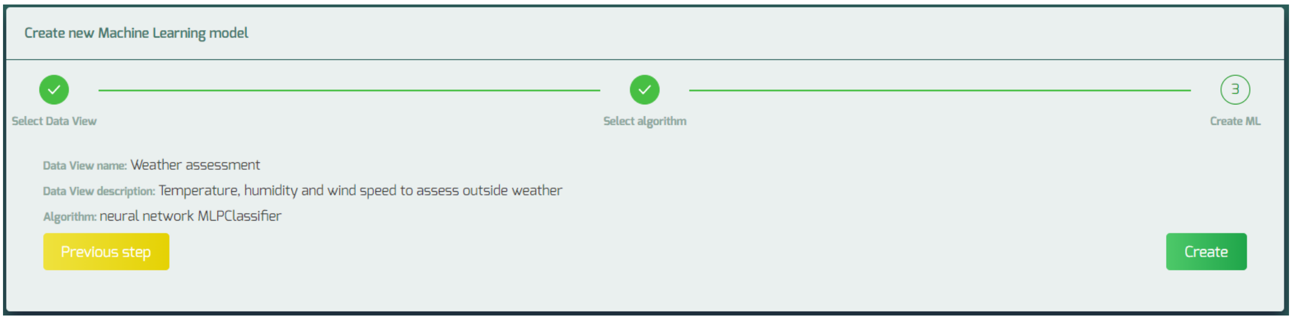


Figure 87. Create ML model – create step

Once created we get the choice to go directly to the details screen:

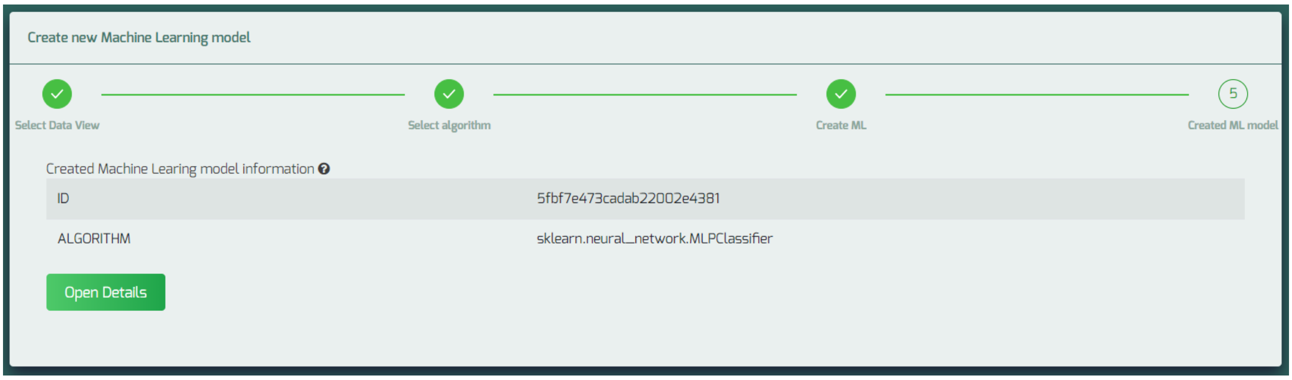


Figure 88. Create ML model – ML model created

4.2.5.2. Review Machine Learning model

Just clicking on the Machine Learning menu will lead to the list view, showing a list of all ML models created by the Service Provider.

In this list users can filter displayed results by algorithm of the model.

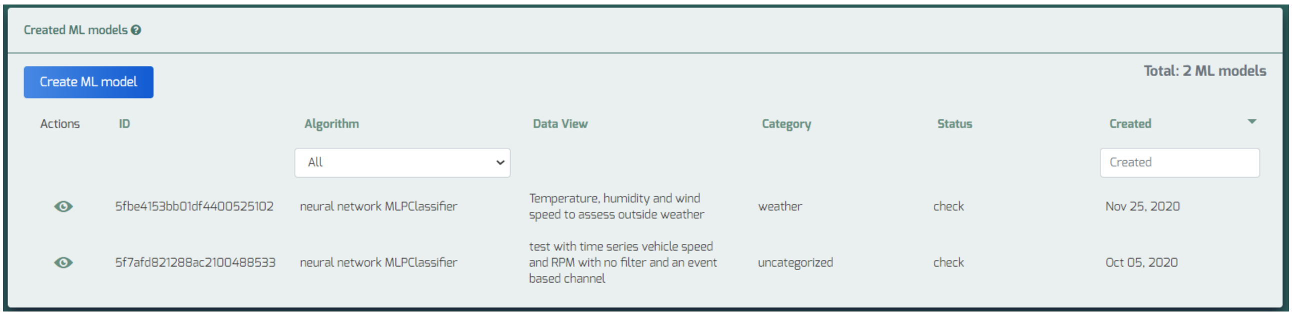


Figure 89. Machine Learning model list

Clicking on any of them will open the details view of that network, showing us the detailed configuration, including the ML model configuration along with the associated Data View and the currently known status of the model training.

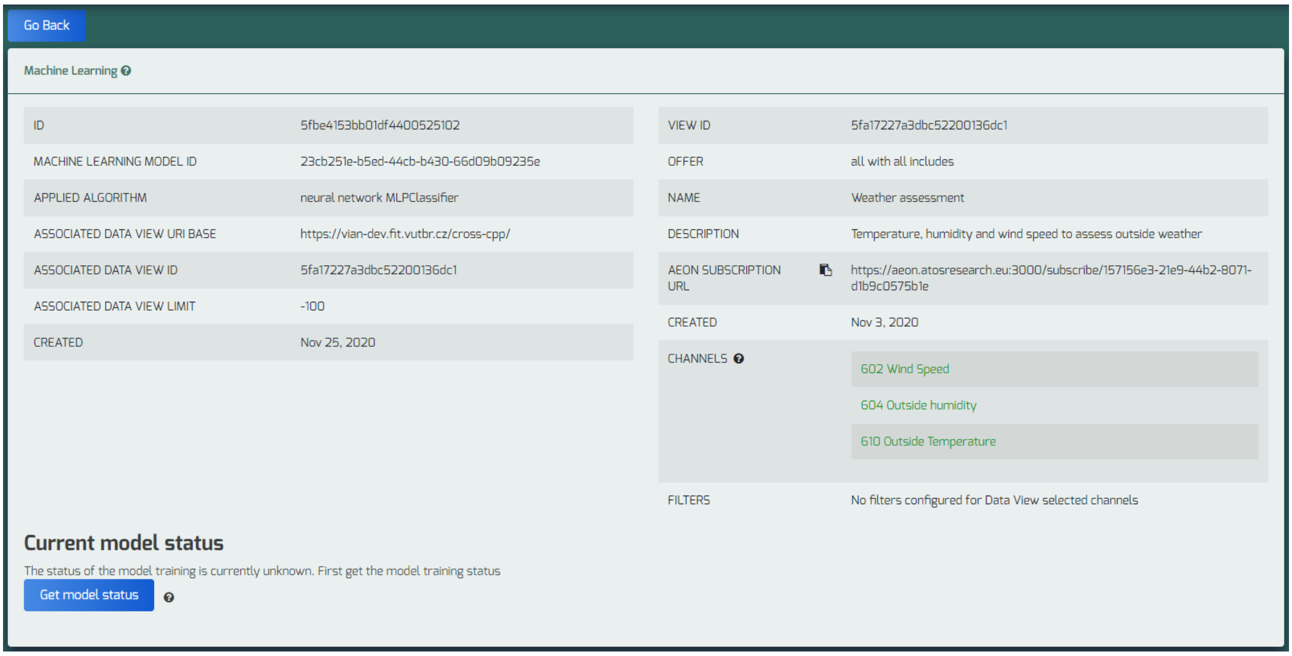


Figure 90. Machine Learning model details view

4.2.5.3. Operate with a Machine Learning model

At first every ML model will have the assigned status of “check”, which means the current training status of the model is unknown. To start operating with the model first is needed to know the status of the model training. To do that just click on the “get status” button.

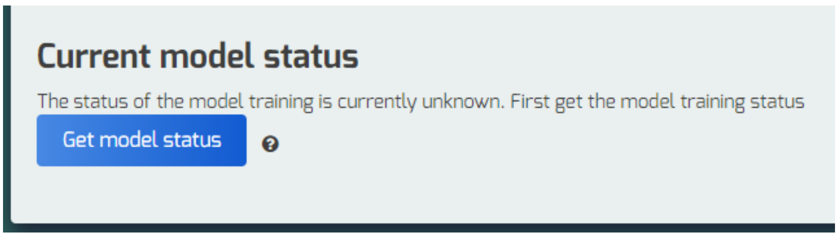


Figure 91. Get Machine Learning model status button

Once we have the status of the model, we have three possible outcomes that gives us different possibilities to operate with the model:

- The training has been initialized or is running at the moment. In this case we will have to wait for the training to be completed. We can also stop the training by killing the model, resulting in a failed training.
- The training has been completed successfully. This is the desired outcome and gives us the opportunity to actually work with the model.
- The training has failed. An error message is displayed, and we can delete the model.

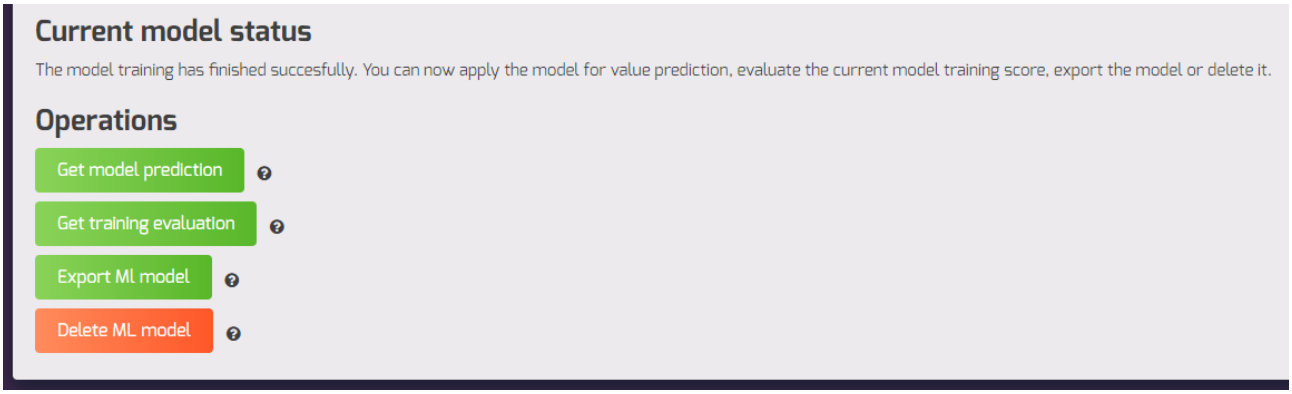


Figure 92. ML model status operations example

4.2.5.3.1. Get model prediction (apply ML model)

You can get a prediction of category values for the uncategorized rows of the Data View by clicking on the "get model prediction model" button.

A table will be displayed with the allocated values.

Machine Learning - model prediction results

602 Wind Speed	604 Outside humidity	610 Outside Temperature	#	vault_id	cpp_id	tstamp	weather	ml_prediction_output
2.5722222222 m/s	91.148163182253 %RH	3.1 ºC	2214634	8c1ce6b6-6846-4eca-92ee-eab3dc21c57c	VH0187	2020-11-23T23:35:00+01:00		cold
2.05777777776 m/s	89.223539950069 %RH	3.3 ºC	2214633	8c1ce6b6-6846-4eca-92ee-eab3dc21c57c	VH0187	2020-11-23T23:28:00+01:00		cold
1.02888888888 m/s	100 %RH	2.5 ºC	2214628	8c1ce6b6-6846-4eca-92ee-eab3dc21c57c	199928	2020-11-23T23:50:00+01:00	very cold	very cold
1.02888888888 m/s	100 %RH	2.3 ºC	2214627	8c1ce6b6-6846-4eca-92ee-eab3dc21c57c	199928	2020-11-23T23:40:00+01:00		very cold
1.54333333332 m/s	100 %RH	2.2 ºC	2214626	8c1ce6b6-6846-4eca-92ee-eab3dc21c57c	199928	2020-11-23T23:30:00+01:00		very cold
1.02888888888 m/s	100 %RH	2.1 ºC	2214625	8c1ce6b6-6846-4eca-92ee-eab3dc21c57c	199928	2020-11-23T23:20:00+01:00		very cold

Figure 93. ML model value prediction table

4.2.5.3.2. Get training evaluation (evaluate ML model)

You can get an evaluation score of the model training. Click the “Get training evaluation” button to get the numeric score.

Figure 94. ML model evaluation score

4.2.5.3.3. Export ML model

You can export the current ML model by clicking the “Export ML model” button and downloading the file in your device.

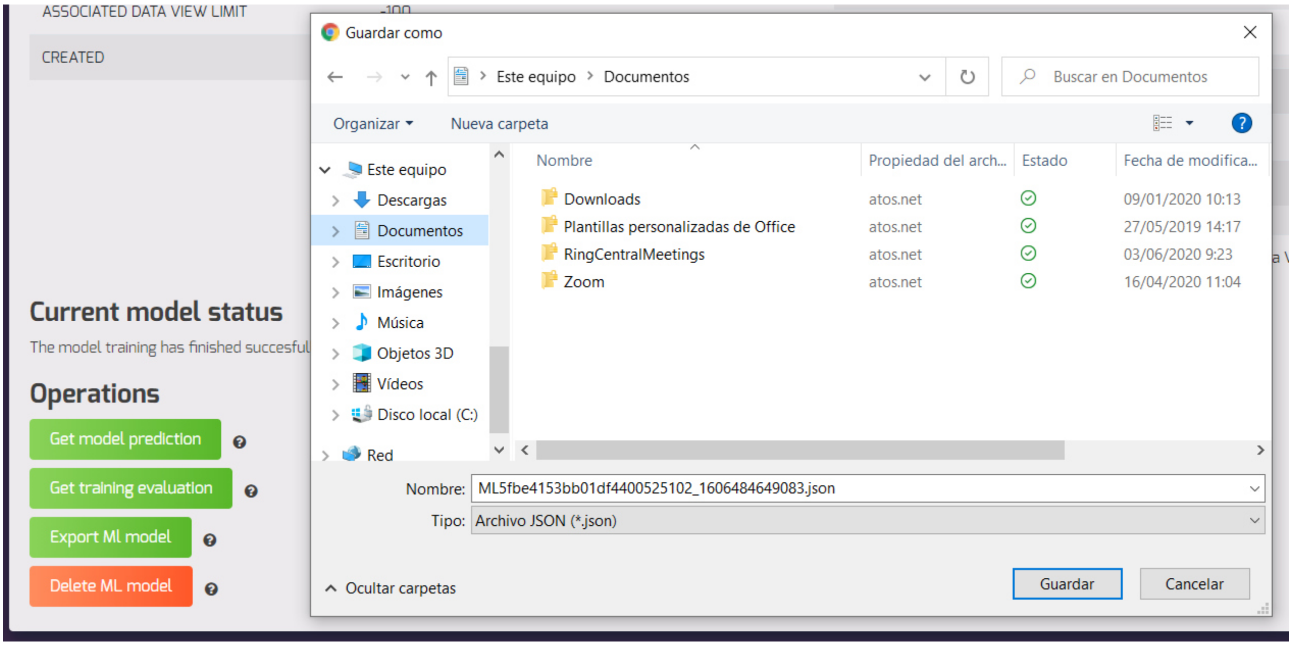


Figure 95. export ML model

4.2.5.3.4. Delete ML model

Once the training has finished, whether it has failed or has been completed, we can delete the model at any time by just clicking in the “Delete ML model” button and confirming. If the model is still running, we first have to kill it to stop the process, and then delete it.

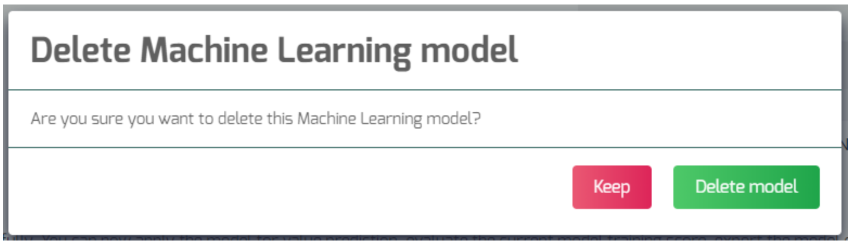


Figure 96. Delete ML model confirmation

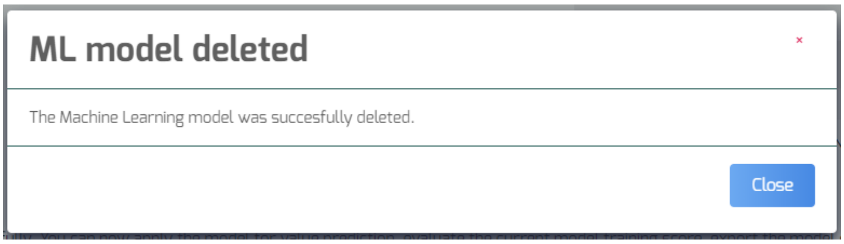


Figure 97. ML model deleted response

5. Security and authorization

5.1. Context Sensitive Security

The Cross-CPP project aims to provide a cross-sectorial Ecosystem, where unified quality of data, commercial confidentiality, privacy, IPR and ethical issues may arise in dynamically changing and varied combinations, requiring the services provided to be adapted to the specific needs of the users and of the providers of CPP data streams under the prevailing contextual conditions.

More detailed explanation of the Context Sensitive Security module can be read in the Cross-CPP Ecosystem guide (to be found under www.cross-cpp.eu).

5.1.1. How does the CSS module work?

To cope with the complexity of these needs, Cross-CPP has an innovative solution that employs a novel combination of two key technologies:

- A characterization for various information about the Cross-CPP Ecosystem environment, a means to extract that information from events occurring in the environment, and to make the corresponding conditions (such as time of day, calendar dates and events, location, user identity, and other specified conditions) available to other processes that can usefully apply that information to adapt services. This capability is embodied as the Context Monitoring and Extraction (CME) module.
- A flexible way to express confidentiality, privacy, IPR and ethical constraints in the form of precise rules that are subject to variable interpretation according to the current contextual conditions, or that are subject to modification in response to changes in the context. This capability is embodied as the Context Sensitive Security (CSS) module.

The CSS capabilities are used to represent rules corresponding to the contracts made between the Service Providers that make a Data Request, and the CPP owner (or a particular CPP, since a CPP owner can possess more than one CPP). Upon acceptance of the Data Request, an access control policy is constructed that codifies the access rights of Service Providers to data from various CPPs and this policy is used during runtime to grant or deny access to the data.

The CSS can function both in the mode of taking CPP extracted context into account or, in the case that the CPP owner does not allow the extraction of context data, a mode that is context independent, as reflected in the configuration of the CSS and CME modules.

5.2. Authorization

Authorization to access CPP data is achieved through an appropriate configuration in the CSS module, including the access control policy and declaration of the conditions and context variables upon which the policy is dependent. The Marketplace acts as the enforcer of the policy decisions rendered by the CSS module in response to access queries. An access query specifies the user requesting data access, the data to be accessed, and the operation requested on the data item. The CSS module interprets its current policy for the given parameters and provides a grant or deny response to the Marketplace, which the Marketplace in turn carries out.

The access control policy specified in the Declarative Policy Language extended for context sensitivity is contained in a file that the CSS Policy Server is instructed to load and use as the current policy. The CSS Policy Tool is used for the development and testing of a policy file. The declaration of conditions and context variables is done in the two files conditions.pl and context.pl respectively. These files are generated by the Context Sensitive Security Setup tool, which provides a simple graphical user interface for configuring context sensitivity.

F.A.Q.

Cross-CPP data-marketplace

Q: What is Cross-CPP data-marketplace?

A: Cross-CPP data-marketplace connects CPP Owners and Service Providers for selling and acquiring Connected Vehicle and Home Building data under the Common Industrial Data model (CIDM). It offers a secure and privacy preserving experience when selling or buying sharing big data, by having the full control over your data shared, to whom and for what purposes.

Cross-CPP offers to cross-sectorial Service Providers, the possibility to search for more than 200 sensor signals, display advance visualization representations (such as Histograms, Geo-Histograms, Time Series) and retrieve those datasets in a seamless experience thanks to the open SDK-API created.

Q: How do I, as Service Provider, register into Cross-CPP data-marketplace?

A: You can find the registration form by clicking the "Sign on!" button in the landing page. Select "Service Provider" role and fill the fields to request your registration. Once your registration is validated by a system administrator an email will be sent to you to confirm your access, Then, access the link in your email, login and accept the consent to start using the Cross-CPP Marketplace.

Q: What do I have to do in order to start working with Cross-CPP data-marketplace?

A: Once registered you must be familiar with the CIDM, as it is the format in which you will receive the data you request. You must also be familiar with AEON, as it is the communication channel used to send the data. You can find information for both in the Service Providers Developers Guide.

Cross-CPP data model

Q: What is the Common Industrial Data Model (CIDM)?

A: The CIDM is a standardized data model for industrial data-driven services. You can find extended information about the CIDM in the Service Providers Developers Guide

Q: Which are the benefits and advantages of using the CIDM model for data -driven services:

A: -The CIDM constitute a major business and technical advantage for Service Providers:

- The CIDM provides a brand-independent and transparent data model, which harmonizes proprietary data into generic datasets independently of any cross-sectorial Industry
- It is built on an open and highly scalable automotive big data format (JSON Schema).
- Active community of service providers increasing the number of signals available from vehicles and Smart Buildings to be recorded as well as the type of measurement channels can be modified or extended

- The Company Backend also provides an origin certification as a CIDM feature to support the validation and verification of origin, integrity and completeness of data. The intention is to protect the data inside the Data Package against manipulation.

Q: What is a signal?

A: A signal is the information provider of each CPP. They are the perception organs of CPPs, and it is their main duty to detect physical phenomenon and chemical quantities. They observe the environment and generate data in the CIDM format. An example could be “speed” or “latitude”

Q: What is a channel?

A: A channel is the way the physical signals and their sampled measurements are implemented and represented in the CIDM format. Some examples could be “Vehicle Speed” using the signal “Speed” in a time-series or in a histogram format, or “Position” using both “Latitude” and “Longitude” signals.

Q: Can I request a new signal or channel?

A: Cross-CPP data-marketplace offers a wide variety of signals provided by the manufacturers. The catalogue is extensive and can be filtered in many ways. If even then you can’t find the signal that you need and/or think can be provided by any of our company backends, please contact us in: support@datagora.eu

Cross-CPP marketplace components

Q: What is the Data Discovery component and how does it work?

A: The Data Discovery component is a tool that allows you to find what data you can access through the marketplace. There you can use the filters provided to narrow or enlarge your results and create Data Request based on the configured search. You can think of it like a test of what would you receive if you publish that request.

Q: What is the Context Monitoring and Extraction module and how can help my Organization?

A: The Context Monitoring and Extraction module allows Cross-CPP to suggest you signals to add to your current Data Discovery signals, based on the context model of the CPP and signals already selected. This might help you find data of interest that you would miss otherwise.

Q: How does the Suggestion component work?

A: The Context Monitoring component allows Cross-CPP to suggest you signals to add to your current Data Discovery filters, based on the context data of the signals already selected. This might help you find data of interest that you would miss otherwise.

Q: How can I use the Context Component in my Data Requests?

A: During the creation of a Data Request the Cross-CPP Marketplace gives the Service Provider the choice of making the Data Request context sensitive and configure the available context filters.

Q: What does it mean my Data Request is context sensitive?

A: The Data Request is context sensitive in case the Service Provider selects some of the CME module data filters to restrict the data packages that it wants to receive. For instance, if the Service Provider just wants to receive data that is provided by vehicles that are driving in the highway and this type of context data is extracted then one case say that the Data Request (and data packages that are produced from this request) is context sensitive.

Q: Which are the consequences of making my Data Request context sensitive?

A: Since you can do this by selecting several context information that could be used to filter the Data Request, the consequence of adding such a filter will be that you receive a smaller set of data packages (filtered by the selected context information) than you would usually receive in the case that no context information is selected.

Data Requests

Q: What is a Data Request?

A: A Data Request is a set of filters that defines which type of data would you like to receive. You would receive data from CPP owners that have accepted these requests through each request unique AEON channel.

Q: How do I create a Data Request?

A: You must get to the Data Discovery and define the data you are interested in through the filters given. Once set you give it a descriptive name, so CPP owners guess in a glance the nature of the request.

Q: How do I receive data from a Data Request?

A: In the very moment a Data Request is created an AEON channel is assigned. You can find the channel configuration in each data request details view. Data from CPP owners that have accepted the request will be sent through each AEON channel assigned to each eligible data request, meaning you can receive data from the same user and signal from more than one request.

Q: Can I modify my Data Request once created?

A: No. The acceptance of a request by a CPP owner implies a consent from its side. Modifying the request would make invalid such consent. Therefore, you can create another Data Request with the new desired configuration.

Q: How can I delete the Data Request published in the Marketplace

A: - You can delete any published Data Request in the details view of that request, or through the provided API endpoint (see Service Providers Developers Guide). This action cannot be undone, and all acceptances of the request will be automatically terminated.

Q: Can I use the data collected for other purposes not described in my Data Request?

A: No. The acceptance of a request by a CPP owner implies a consent from its side. That user allows certain usage of the data given and only for the purposes described in the request by the Service Provider (such as use for analytics or evaluate its context).

Toolbox

Q: What is the toolbox?

A: The Toolbox is a set of tools that offers Service Providers a way to generate analytics from the data obtained from Data Requests or further filter this data in order to get exact measurements or use the service as a notification system.

Q: How can I request analytics?

A: Any analytics uses data packages received from a Data Request, meaning that first the data Request must have been accepted by CPP Owners and started receiving data. Then, on the Cross-CPP Marketplace a Service Provider can create analytics based on those Data Requests.

Q: How do I get my analytics results?

A: Depending on the analytics type results can be instantly shown on the screen, as a diagram, chart, map and so on, or a new AEON channel is provided in order to subscribe to upcoming analytics results. Every analytics type explains the way to get the results on screen.

Q: How can I consult my analytics?

A: Any created analytics can be consulted on the Cross-CPP Marketplace under Analytics section. There a Service Provider can get the AEON channel for subscription, see results of one-time analytics, or even delete them.

Q: Which Time Series forecasting method should I use for my time series predictions?

A: Prior to attempting to predict future values of a time series, you should verify the assumptions your selected algorithm does on the input data. For example, ARIMA will assume that your data is autoregressive and Regression trees will not be able to detect trends on data (thus not being advisable for a time series in which trend is a significant feature). If you are unsure about the underlying patterns of your data, Neural Networks may work best for you, as it does not rely on any a-priori assumption.

Q: How does the embedding dimension parameter "m" affect the entropy metrics?

A: The time-series are defined by sequences of points. The embedding dimension refers to the number of points use to evaluate. So then, smaller embedding dimensions would yield more detailed information while larger numbers tend to be more general. Bear in mind if the embedding dimension is too low maybe the results may be chaotic.

Q: How do I choose between a Pearson and a Spearman's correlation coefficient?

A: This decision will depend on the nature of your data, as these coefficients measure different types of association between variables (Pearson quantifies linear relationship, while Spearman does so with non-linear patterns). Their use is not exclusive, so even both could possibly be valuable to your analysis.

Q: What are the units of the trajectories module?

A: For distances, the meter, and for times the second. Then the velocity is expressed in m/s.

Q: What is the interpretation of the results of clustering the trajectories?

A: Each trajectory included in the data is assigned to a numerical ID that refers to the cluster that is being assigned. By instance, all trajectories of similar length from east to west are grouped in the cluster number 1, so the results are 1 for each trajectory.

Q: What do the nodes represent in the networks module? And the links?

A: Each node is an object. It could represent a house, building, a car. The links are an abstract representation of some magnitude that is being measured. Links could be based on distances by instance.

Q: What is a Data View?

A: A Data View is a configuration to get data filtered by specific values constraints for one or more Measurement Channels included in a Data Request.

Q: How can I create a Data View?

A: Go to Data Views section under Toolbox in the Cross-CPP Marketplace. There a step by step guide will be offered.

Q: How do I get the data view results?

A: Service Providers can consult their generated Data Views through Data Views under Toolbox section. There the configuration can be consulted, as well as the AEON channel to subscribe to. Also, the options of retrieving the latest data or even deleting the Data View are offered there.

Q: How can I use a created ML model?

A: A machine learning model that was built in a previous step can be applied on new (unseen, unannotated) Data View rows by invoking the Apply function. The service estimates the category on the given data rows and outputs it as the response.

Q: My service processes a lot of data, is the ML component ready for big data application?

A: Yes, even a large neural network model loads in less than 480 ms and it can apply the category prediction with the speed of more than 5000 rows per second on the Cross-CPP testing infrastructure.

Q: What ML methods should I try first?

A: Although the ML components support all new and fancy neural network methods available in Scikit-learn, Google TensorFlow, and Facebook PyTorch libraries, we suggest to start with the simple linear method of Stochastic Gradient Descend or the basic Multi-Layer Perceptron neural network for your initial experiments. Often, the quality of results provided by these models is satisfactory and performance gains can be brought by additional annotations of the data.

AEON

Q: What is AEON?

A: AEON is a cloud platform to create applications with real time communications channels. You can find extended information about AEON in the Service Providers Developers Guide

Q: How can I use AEON to subscribe to my data requests and data analytics?

A: AEON provides an SDK (Node.js, JavaScript and Java) that encapsulates the complexity of connecting to a socket server. Please refer to section 6.4, document examples and online page for extended documentation.

Q: How do I create or configure an AEON channel?

A: You don't have to create or configure any channel. All needed AEON channels, such as for data requests or analytics, are created and assigned by Cross-CPP. You only have to use the channels given. The channels configuration can be found in the details view of each data request or analytics.

Glossary

Administrator: Cross-CPP marketplace system administrator

Autoregressive data: In a time series domain, it refers to data which values depend on prior data points from the same time series.

AEON: AEON application

AEON application: publication/subscription based communication application

AEON channel: set configuration for communication between two actors through AEON application

Analytics Toolbox: set of available analytics functions to be requested by the Service Provider

CIDM: Common Industrial Data Model

CIDM model: standardized data model for industrial data-driven services

CME: Context Monitoring and Extraction

Company Backend: system of an OEM that provides its users data to the Cross-CPP marketplace

Contract: entity that resumes the acceptance of a data request from a CPP owner

CPP: cyber-physical product

CPP Data: data created by a CCP and sent to the system by the Company Backend

CPP owner: CPP owner which CPP is registered in the Cross-CPP data-marketplace

Cross-CPP: System

CSS: Context Sensitive Security

Data Request: set of configurations that define a scope for CPP Data to be received by a Service Provider

Data View: set of configurated filters to receive specific values from a Data Request through a different notification channel

Entropy: It is usually explained as the order of a system. It is more accurate to understand the entropy as the lost information of a system. This definition for data classification problems implies the algorithms search for the variables that reduces the lost information of the system, those are the best classifiers.

Homoscedasticity: property of a multivariate domain in which the variance of each variable's error term is equal.

Machine Learning Analytics: A type of Analytics included in the toolbox

Marketplace: Marketplace Web Application

Measurement Channel: sampler of the data the signals process

Monotonic relationship: A type of association between variables that occurs when two variables tend to increase or decrease in the same direction, but not following a linear pattern (linear relationship).

MP: Marketplace

Multilayer Perceptron: A type of Artificial Neural Network with a varying number of hidden (processing) layers.

Networks Analytics: A type of Analytics included in the toolbox

Network Diameter: Value indicating the shortest distance between the two most distant nodes in a network.

Network Efficiency: Measure of how well information is exchanged between the nodes of a networks.

Node Eccentricity: Value representing the centrality of a network's node, or how close it is to the other nodes in the network.

OEM: Original Equipment Manufacturer

Rank Correlation: A type of correlation measure that quantifies ordinal association between two variables.

Service Provider: actor who receives the data created by owners to use it on the creation or improvement of services

Service Provider Wallet: group of MP functionalities for Service Providers

Signal: information provider of the data the CPP sensors generate

Stationarity: Property of a time series indicating that its statistical properties (e.g. mean, variance…) do not change over time.

System: the whole lot of applications that conforms Cross-CPP, including Marketplace Web Application and Marketplace Server.

Time Series Analytics: A type of Analytics included in the toolbox that analyses drifts in the data flow of time-series type channels.

Time Series Complexity: Measure of the presence of nonlinear patterns that explain the behaviour of a time series' data.

Trajectories Analysis: A type of Analytics included in the toolbox that uses trajectory related signals.

Figures

About Cross-CPP

The objective is to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research is to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project will develop: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to pro-vide enhanced local weather forecast and predict and optimise energy consumptions in households. The project will build upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, will be used as a basis for further development aiming to extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

Funded by the Horizon 2020 Framework Programme of the European Union

6 Annex 5: User Guide Administrator

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

CROSS-CPP

CROSS-CPP

CROSS-CPP MARKETPLACE
USER GUIDE

ADMINISTRATOR GUIDE
(PLATFORM MANAGEMENT)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780167

Introduction

Administrator User Guide describes management functionalities within CROSS-CPP Marketplace, such as listing, reviewing and managing data signals, measurement channels and platform users.

Purpose

This guide aims to help users from a n Administrator role on how to use the platform and give knowledge about the different functionalities available.

Audience

This guide is meant for and solely for users of CROSS-CPP Marketplace with Administrator role.

Scope

The contents of this guide are meant to be taken into consideration only when using the CROSS-CPP Marketplace and will only cover functionalities meant to be used by the role stated above.

CROSS-CPP team does not take responsibility on bad use of the application or the data provided when not following the instructions given in this guide.

Troubleshooting

For any questions or inquiries about the use of the CROSS-CPP Marketplace web application or the contents of it or this guide, or if you find there is no content in this guide for some functionality please forward it to: marketplace-support@cross-cpp.eu.

Contact

Cross-CPP Project website: https://cross-cpp.eu

Cross-CPP Marketplace: https://datagora.eu

Marketplace support: marketplace-support@cross-cpp.eu

Context Monitoring and Extraction Module (CME): context-support@cross-cpp.eu.

Contents

Guide

There are two main functionalities in the Cross-CPP Marketplace (Figure 1):

- **Catalogue**: list of all available registered signals and their assigned measurement channels. (2.1 Catalogue)
- **Data Discovery**: discover the data available in the Cross-CPP Marketplace through a set of filters to create your Data Requests for the data you are interested in. (2.2 Data Discovery)

CROSS-CPP Marketplace offers a section specifically for administrators apart from the main functionalities:

- **Management Panel** (3 Management Panel): is the management central point where administrator can take the control of the platform, including data signals, measurement channels and registered users.

At a glance, CROSS-CPP Marketplace main functionalities can be access directly through the side menu

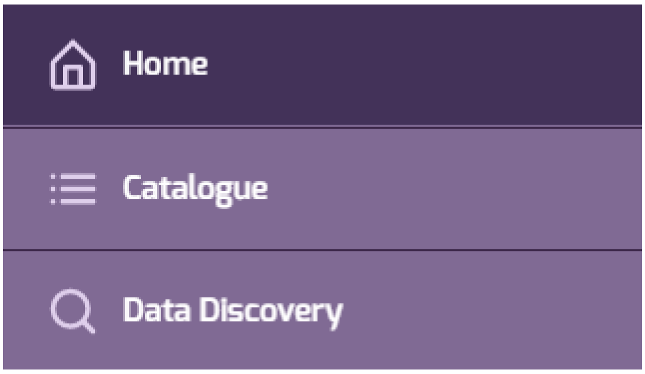


Figure 1. Main functionalities menu

Other functions can be found in the top right corner of the toolbar which shows the F.A.Q., a CPP selector and a menu including Marketplace Profile, User Settings and Log out.

The CPP shown in the toolbar is the one for which all data retrieved will belong to. In case the administrator has access to more than one CPP registered it can be changed through the toolbar or in the User Profile.

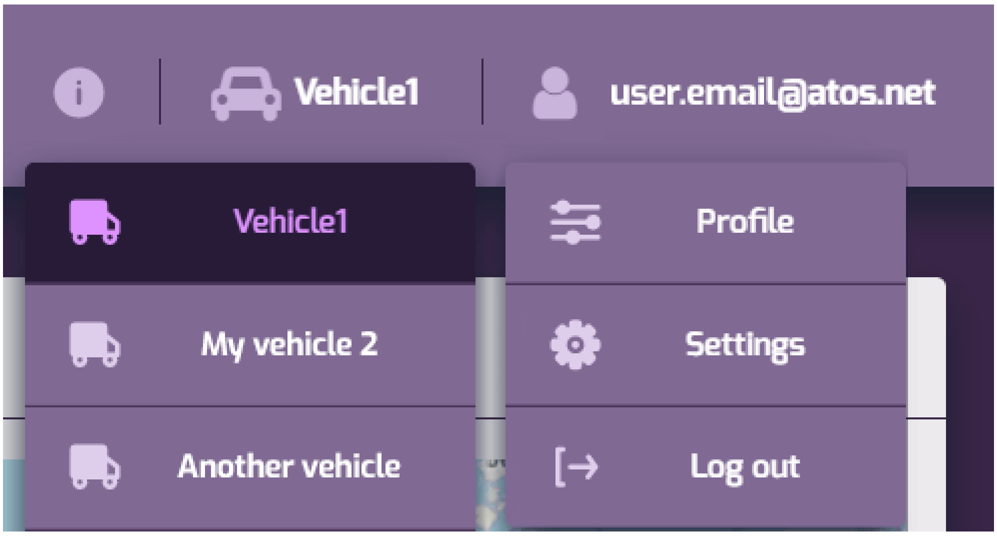


Figure 2. User menu

1. Common Industrial Data Model (CIDM)

Cross-CPP uses the CIDM as its data model. All data pushed into Cross-CPP Marketplace must follow this model.

As the administrator is responsible of the maintenance of the signals catalogue, it requires to understand the CIDM to be able to create, update and manage signals and measurement channels.

1.1. Model Architecture

The CIDM architecture consist in three layers:

- The Signal layer consists in the information providers within the CPP devices like vehicles or smart buildings. Signals observe the environment and generate data, they detect physical and chemical phenomenon, for example, speed, temperature, charge state level, etc.
- The Measurement Channel layer provides signals data aggregation. The data needs to be pre-processed since raw sensor data exceeds the available storage and transferring capacity, to reduce the size of data down-sampling and histograms methods are provided.
- The data layer aggregates data inside data packages to store and transfer. One data package contains data from exactly one signal measured with one Measurement Channel. In addition to the actual data, Data Packages contain header information ("meta data"). This header information provides ownership of the data and gives quality of signal indications by OEM signatures or describes parameters of the measurement (e.g. time, rough position estimate, etc.).

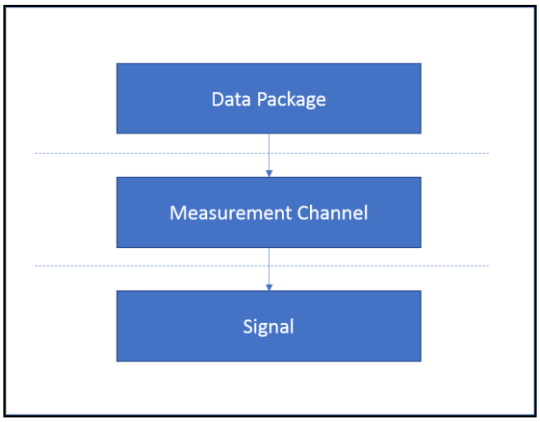


Figure 3: Layered High-level View of the Common Industrial Data Model (CIDM)

1.2. Signal Layer Specification

Sensors are the perception organs of CPP devices like vehicles and buildings. It is their main duty to detect physical phenomenon and chemical quantities by transferring them into electrical signals. The signal layers describe different types of signals and formats represented in the system. A new property is needed to group signals regarding the signal source type, cpp-type. **Fehler! Verweisquelle konnte nicht gefunden werden.** shows the UML modelling of the signals for CIDM.

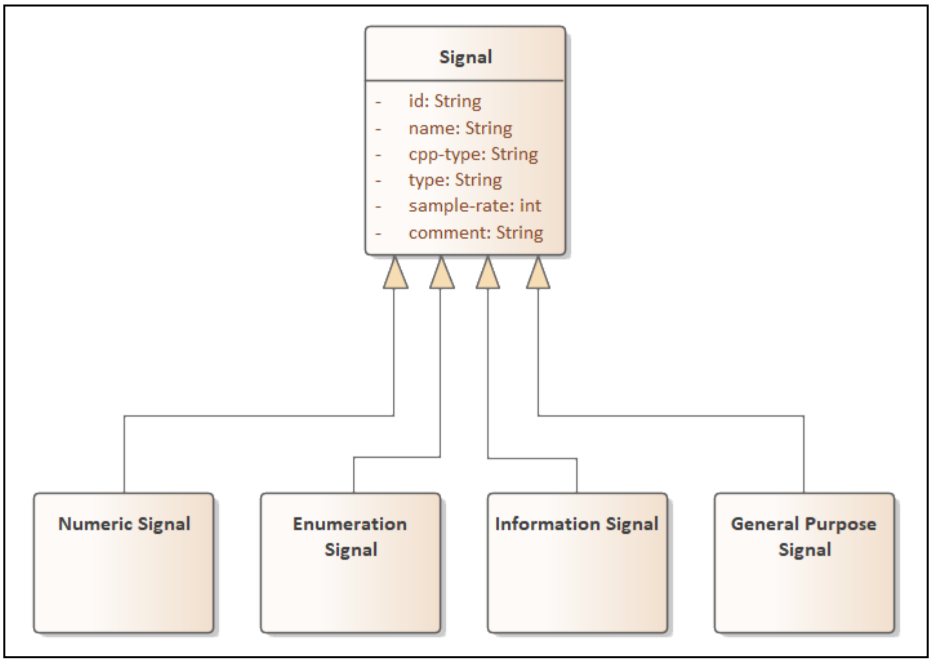


Figure 4. Signal UML Model.

The cpp-type is a required property that must be one of the two values, “vehicle” or “building”. The table below shows the complete definition of the Signal.

Property	Occurrence	Type	Format	Description
Common Properties				
id	Required	String		Unique Identifier of the Signal
name	Required	String		Name of the Signal
cpp-type	Required	String	one of: - vehicle - building	Type of the CPP
type	Required	String	one of: - numeric - enumeration - information - general-purpose	Type of the Signal
format	Optional	String		Signals representation format
sample-rate	Required	Numeric	double	Sample rate in Hz (Samples per Second). Must be larger than or equal to zero.
comment	Optional	String		Description of the signal
Numeric Signal				
type	Required	String	"numeric"	Type of the Signal needs to be numeric
format	Required	String	*<numeric formats>*	Signal's numeric representation (e.g. uint8, double, etc.)
min	Required	Number	<according to format>	Minimum Signal value
max	Required	Number	<according to format>	Maximal Signal value
resolution	Required	Number	<according to format>	Signals resolution
Unit	Required	String		Unit of the Signal (e.g. ºC)
Enumeration Signal				
type	Required	String	"enumeration"	Signal's type attribute needs to be "enumeration"
items	Required	Array	String	String array with possible Signal values
Information Signal				
type	Required	String	"information"	Signal's type attribute needs to be "information"

Property	Occurrence	Type	Format	Description
format	Required	String		Signals representation format (e.g. VIN, etc.)
General Purpose Signal				
type	Required	String	"general-purpose"	Signal's type attribute needs to be "general-purpose"
*	Optional	Any	No	May be extended with further attributes

Table 1. Signal property definition.

1.3. Measurement Channel Layer Specification

The measurement layer defines how sensor signals are captured and processed. One Measurement Channel describes how samples from one (or more - in the case of multidimensional histograms) sensor signal are aggregated and measured. Figure 5 shows the Measurement Channel UML model.

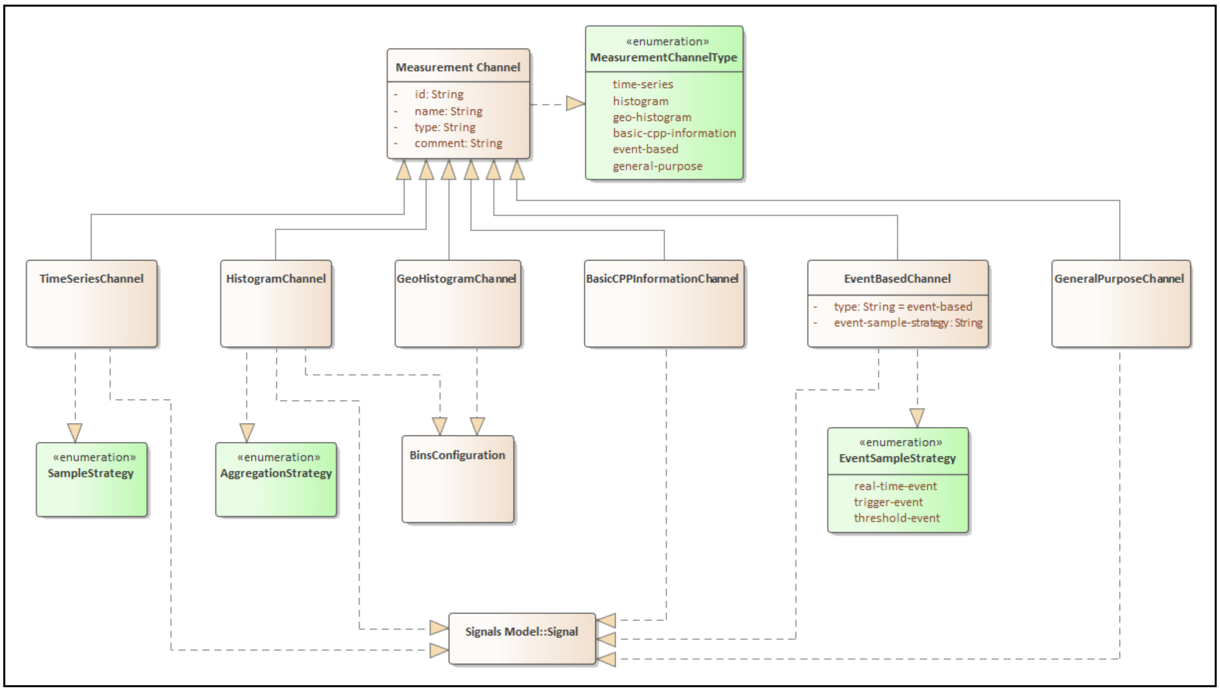


Figure 5. Measurement Channel UML Model.

The basic CPP information channel provides static information that is not measured by sensors but provides information about the CPP device, like the colour of a vehicle, the number of floors of a building, the identification number of a car, etc.

The event-based measurement channel provides information of events that occurs when the value of a measurement gets to a specific value (real-time-event) or when the value passes a specific threshold (threshold) and the last one that provides information about the event and the data-packages that have triggered the event. The table below details the measurement channel specification.

Property	Occurrence	Type	Format	Description
Common Properties				
id	Required	String	No	Unique Identifier of the Measurement Channel
name	Required	String	No	Name of the Measurement Channel
type	Required	String	one of: - time-series - histogram - geo-histogram - general-purpose - event-based - basic-cpp-information	Type of Measurement Channel
Comment	Optional	String	No	Description of the signal
Time Series Measurement Channel				
type	Required	String	"time-series"	Type of the Measurement Channel needs to be time-series
format	Required	String		Data type format of the samples
dimension	Optional	Number	uint32	Dimension of the Time-series. If dimension is not given, one-dimensional is assumed
capture-interval	Required when on-change is false	Number	double	Capture interval between two samples in seconds. Only required, when on-change is false.
on-change	Required	Boolean		Does Measurement-Channel only record changes in signal
sample-strategy	Required	String	one of: - min - max - average - last-known-value	Signal sampling strategy
signal	Required	Object	Array of Signal Object	See section 6.1.2 for Signal object definition

Property	Occurrence	Type	Format	Description
Histogram AND geo-Histogram Measurement Channel				
type	Required	String	one of: - histogram - geo-histogram	Type of the Measurement Channel needs to be histogram or geo-histogram
aggregation-strategy	Required	String	one of: - time - count - min - max	Histogram values aggregation strategy
capture-interval	Required	Number	double	Capture Interval of one Histogram. Needs to be larger than zero. +Infinity is valid (see IEEE 754).
dimensions	Required	Number	uint32	Dimensions of the Histogram
bins	Required	Array	Bin-Configuration Object	Array of bin configurations. Array needs to contain exactly one configuration for every dimension/axes of the histogram
Geo-Histogram Measurement Channel				
type	Required	String	"geo-histogram"	Type of the Measurement Channel needs to be geo-histogram
geo-resolution	Required	Numeric	double	Zoom level of the geo-histogram
Basic CPP Information Measurement Channel				
type	Required	String	"basic-cpp-information"	Type of the Measurement Channel needs to be basic-cpp-information
signal	Required	Object	Signal Object	See section for Signal object definition
Event Based Measurement Channel				
type	Required	String	"event-based"	Type of the Measurement Channel needs to be event-based

Property	Occurrence	Type	Format	Description
format	Required	String		Data type format of the samples
event-sample-strategy	Required	Event Sample	one of: - real-time-event - trigger-event - threshold-event	Event sampling strategy
comment	Optional	String	No	Description of Event Strategy
General Purpose Measurement Channel				
type	Required	String	“general-purpose”	Type of the Measurement Channel needs to be general-purpose
signal	Required	Any		See section 6.1.2 for Signal definition

Table 2: Measurement channel definition

1.4. Data Package Layer Specification

Data Packages contain the actual data of Signal measurements. As Signals are the information providers and Measurement Channels define the process of data acquisition from those Signals, Data Packages provide a structure for storing the data. In addition, they provide meta / header information containing time of recording, data ownership information, etc. Data Packages contain data from exactly one Measurement Channel. This leads to six different types of Data Packages that are defined similar as the Measurement Channels:

- Time Series Data Package
- Histogram Data Package
- Geo-Histogram Data Package
- Event based data Package
- Basics CPP information Data Package
- General Purpose Data Package

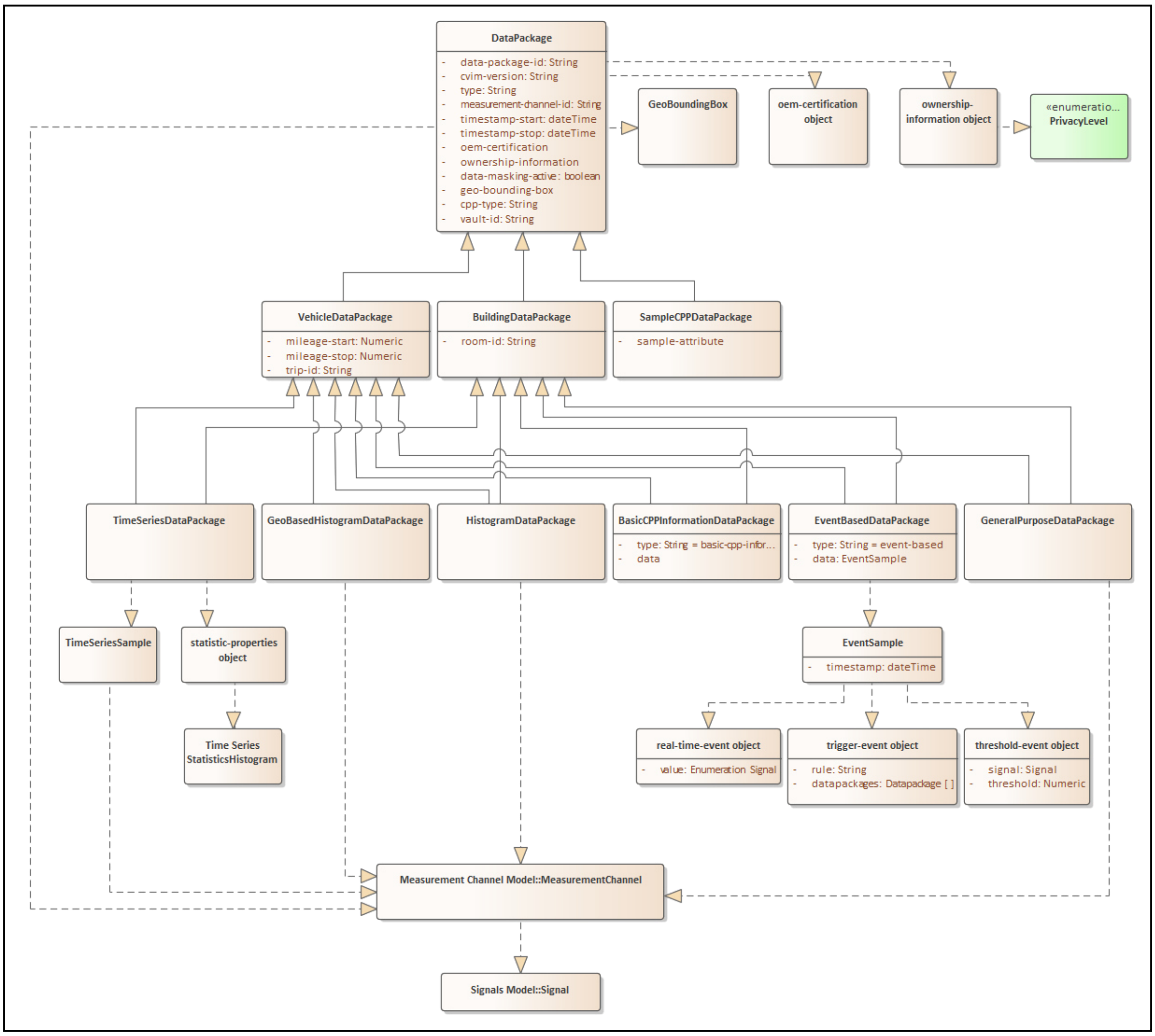


Figure 6. Data Package UML Model.

The data-package definition includes cpp-type to indicate is the data-package belongs to a "building" or a "vehicle" cpp and definition of data-package type, such as basics-cpp-information data-package and the event-based data-package.

The data of basics-cpp-information data-package depends of the signal type definition of the measurement channel - numeric, enumeration, information or general-purpose.

Event-based data-package has an additional property, "event-sample-strategy", to indicate three different type of event, real-time, trigger and threshold. According to the type of event, the "data" object has two mandatory properties, "timestamp" and "value", and an optional property named "datapackages" that is an array of the data-packages that triggers the "trigger-event".

The "building" CPP devices provides a set of sensors distributed along the rooms of the building, in order to identify the devices of the same room a new property has been included, "room-id".

Property	Occurrence	Type	Format	Description
Common Properties				
data-package-id	Required	String	UUID	Identifier of the Data Package. Unique per Cloud Storage Vault, Set by Cloud Storage Provider
cvim-version	Required	String	version	The name of the property is for backward compatibility with CVIM. Must be set to 1.2.1
type	Required	String	one of: - time-series - histogram - geo-histogram - general-purpose - event-based - basic-cpp-information	Type of the Data Package
vault-id	Required	String	UUID	ID of the Cloud Storage Vault, where the data is stored in.
cpp-id	Optional	String	any	ID of the CPP
cpp-type	Required	String	one of: - vehicle - building	Type of the CPP
trip-id	Optional	String	any	Trip-ID of the User
room-id	Optional	String	any	ID of the room in a building where the measurement data was collected
measurement-channel-id	Required	String		Identifier of the Measurement Channel whose data is inside this data package
mileage-start	Optional	Number	double	Mileage at the start of the measurement in kilometres (km)
mileage-stop	Optional	Number	double	Mileage at end of measurement (km)
geo-bounding-box	Optional	Object	Geo-Bounding-Object	Geographic bounding box
location	Optional	Object	Location-Object	Single location including latitude and longitude

Property	Occurrence	Type	Format	Description
oem-certification	Optional	Object	OEM-Certification-Object	OEM Certification
data-ownership-information	Optional	Object	Ownership-Information - Object	Data Ownership Information
expiration-date	Optional	String	date-time	Data expiration date
data-masking-active	Optional	Boolean		Indicates status of data-masking (true = active)
Time Series Data Package				
type	Required	String	"time-series"	Type of the Measurement Channel needs to be time-series
timestamp-start	Required	String	date-time	Measurement start time
timestamp-stop	Required	String	date-time	Measurement stop time
number-of-samples	Required	Number	uint32	Number of samples that are stored in data
statistic-properties	Optional	Object	statistic-properties-object	Provides statistic properties about the data
data	Required	Array	time-series key-value-pair - object	Array of time-series-data Objects. The size of the array mist equal number of samples
Histogram Data Package				
type	Required	String	histogram	Type of the Measurement Channel needs to be histogram
timestamp-start	Required	String	date-time	Measurement start time
timestamp-stop	Required	String	date-time	Measurement stop time
data	Required	(Multi-dimensional) Array	Number	Array containing he bin counts. Size of array must match the dimension and bin configuration of the related Measurement Channel. Number format depends on Histogram aggregation-stategy

Property	Occurrence	Type	Format	Description
Geo-Histogram Data Package				
type	Required	String	geo-histogram	Type of the Measurement Channel needs to be geo-histogram
timestamp-start	Required	String	date-time	Measurement start time
timestamp-stop	Required	String	date-time	Measurement stop time
data	Required	(Multi-dimensional) Array	Number	Array containing the bin counts. Size of array must match the dimension and bin configuration of the related Measurement Channel. Number format depends on Histogram aggregation-strategy. The outer most dimension is the geo-dimension. It must match in its size the size of the geo-tiles array.
geo-tiles	Required	Array	Geo-Tile Object	Array of geo-tile objects. Only visited tiles are included.
Basic CPP Information Data Package				
type	Required	String	"basic-cpp-information"	Type of the Measurement Channel needs to be basic-cpp-information
Timestamp	Required	String	date-time	Measurement date time
data	Required	Any		Data depends on the type of signal of the measurement channel
Event Based Data Package				
type	Required	String	"event-based"	Type of the Measurement Channel needs to be event-based
Timestamp	Required	String	date-time	Measurement date time
event-sample-strategy	Required		one of: - real-time-event - trigger-event - threshold-event	Event sampling strategy

Property	Occurrence	Type	Format	Description
data	Required	Object	Event Sample Object	event-based data Object indicating an event
General Purpose Data Package				
type	Required	String	"general-purpose"	Type of the Measurement Channel needs to be general-purpose
Timestamp	Required	String	date-time	Measurement date time
data	Required	Any	time	Datatype depends on Measurement Channel

Table 3. Data Package definition.

Property	Occurrence	Type	Format	Description
Timestamp	Required	String	date-time	Timestamp of the event
value	Required	String		e.g. "*Ignition On*", *"Wipers Off"*
datapackages	Optional	Array of data-package		

Table 4. Event Sample Object

2. Main Functionalities

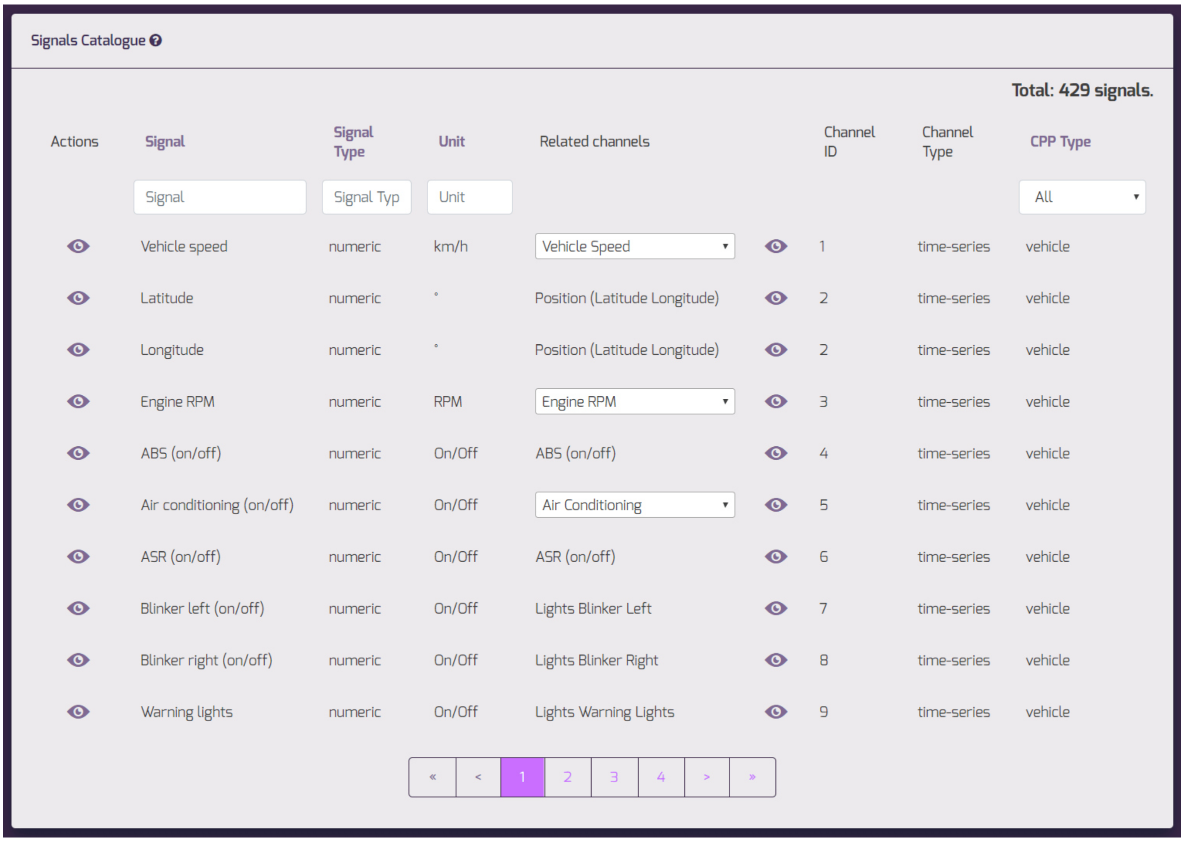
2.1. Catalogue

The Catalogue (see Figure 7) shows a list of all available signals under a harmonized data model called Common Industrial Data Model (from now on CIDM – see F.A.Q in section 8) which enables to retrieve data from Data Providers (CIDM has a dedicated guide to be found in cross-cpp.eu). The data in the Cloud Storage is stored following the CIDM which harmonizes the proprietary OEM formats into brand independent data formats (made then available in the catalogue of the marketplace).

In the Catalogue, a Service Provider can search and filter by different attributes and see the channels that use each signal. Below, the filters which appears in Signals Catalogue are explained (see Figure 8):

- **Signal**: the name of the signal describing what is measured.
- **Signal type**: type of the signal data. Following the CIDM this can be "numeric", "enumeration", "information" or "general-purpose".
- **Unit**: units of the data

- **Related Channels**: data channels that uses the data received from this signal. In case there is more than one that use the signal, a list is available to select which one is of interest.
- **Channel ID**: identification of the selected channel that receives data from the signal
- **Channel Type**: selected channel type of data receiving. Following the CIDM this can be "time-series", "histogram", "geo-histogram" or "general-purpose".
- **CPP Type**: from which type of CPP this signal receives data from. At the moment this can be from vehicles or buildings.


Signals Catalogue

Total: 429 signals.

Actions	Signal	Signal Type	Unit	Related channels		Channel ID	Channel Type	CPP Type
	Signal	Signal Typ	Unit					All
	Vehicle speed	numeric	km/h	Vehicle Speed		1	time-series	vehicle
	Latitude	numeric	°	Position (Latitude Longitude)		2	time-series	vehicle
	Longitude	numeric	°	Position (Latitude Longitude)		2	time-series	vehicle
	Engine RPM	numeric	RPM	Engine RPM		3	time-series	vehicle
	ABS (on/off)	numeric	On/Off	ABS (on/off)		4	time-series	vehicle
	Air conditioning (on/off)	numeric	On/Off	Air Conditioning		5	time-series	vehicle
	ASR (on/off)	numeric	On/Off	ASR (on/off)		6	time-series	vehicle
	Blinker left (on/off)	numeric	On/Off	Lights Blinker Left		7	time-series	vehicle
	Blinker right (on/off)	numeric	On/Off	Lights Blinker Right		8	time-series	vehicle
	Warning lights	numeric	On/Off	Lights Warning Lights		9	time-series	vehicle

« < 1 2 3 4 > »

Figure 7. Signals catalogue

Also, each signal and channel have its own detailed view with all available information, including direct links to each other.

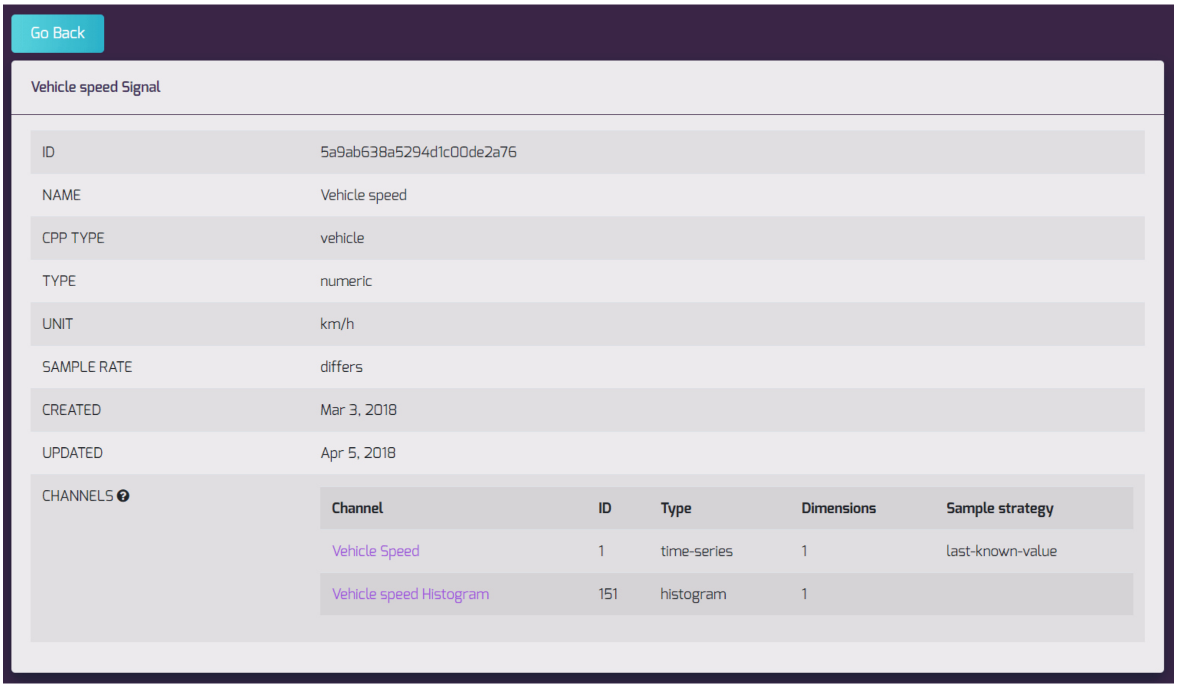


Figure 8. Signal details

2.2. Data Discovery

The Data Discovery offers CROSS-CPP users a tool to browse the data currently available to data consumers from all registered owners. A wide variety of filters are provided in order to help users to narrow the results.

In a first step, the user has to select the signals for which data is interested in. If none is selected, a search for all channels will be requested. One a signal is selected the channel that uses that signal will be added to the selection list. Selected channels can be removed from the list at any time.

If one or more channels are selected suggestions can be requested. This is possible thanks to the implementation of the Context Monitoring and Extraction module (CME) (check additional information in FAQs) within the Discovery process. In case suggestions are requested, the CME will provide a list of channels related to those already selected[1]. Channels of interest can be added directly from the suggestion list. Also, suggestions can be requested again each time a new channel is added.

Once the channels are selected, the user may apply several filters to the data packages (see Figure 9):

---

[1] Suggestions are provided based on a context model that is the result of analysis of the physical relation between signals (e.g. in case temperature is a selected signal, the CME would suggest the humidity and sun intensity

- **Date and duration** section filter the results by data package submission date, measurement recording date and measurement time duration.
- The **geographical selection** offers a way to narrow results in some geographical area. This can be set selecting a country in the list, or drawing a customized area entering the area bounds coordinates.

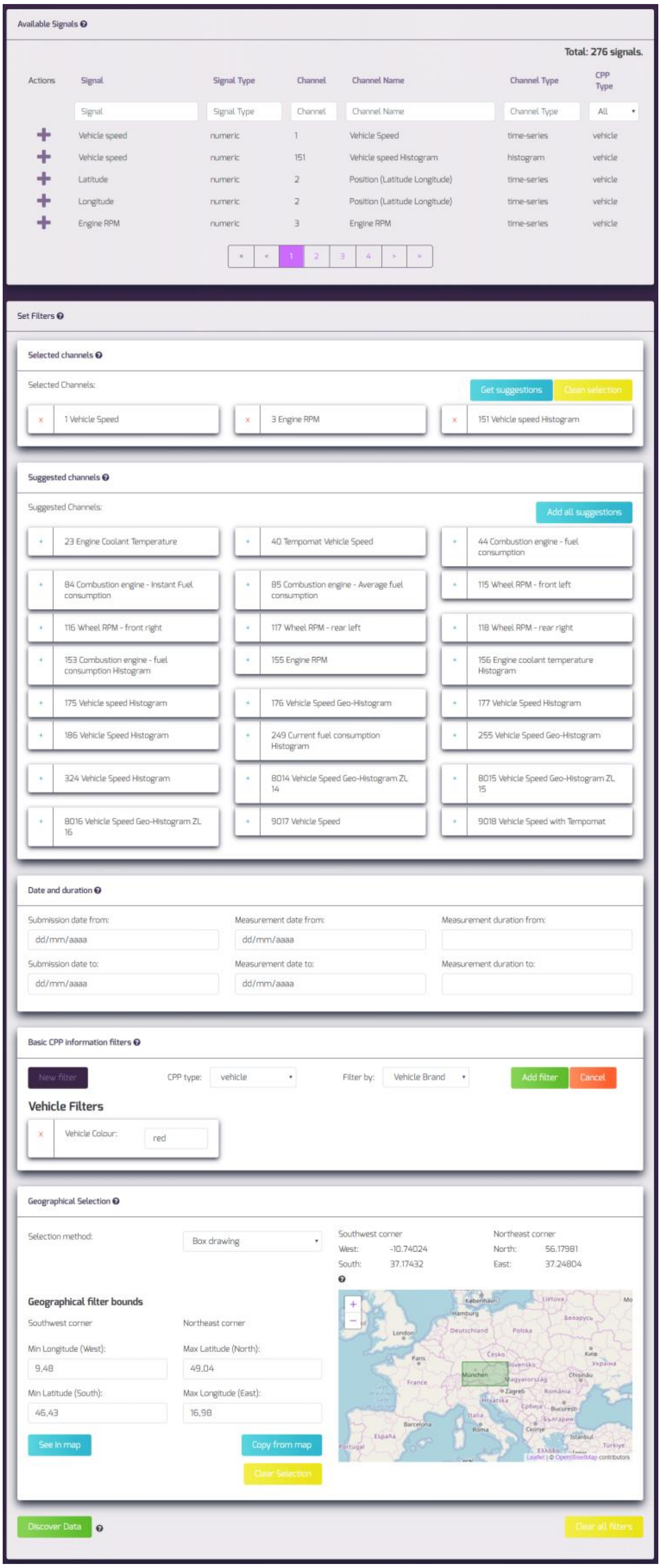


Figure 9. Data Discovery filters

Once the filters are set, Service Providers can see the data they would receive from a data request with selected configuration (See Figure 10). Only metadata is used in this process and no real data can be shown or retrieved through this functionality:

- A box with some **general statistics** shows the total of CPP owners and the number of selected channels that contains any data, and the number of data packages retrieved in the discovery.
- For each type of CPP selected with data a box is displayed showing the number of entities and channels with data, and the minimum, maximum and average duration of its measurements.
- The **heatmap** offers and overview of the geographical areas that provides data, in case it is desired to narrow the discovery to a specific area.
- The **pie charts** show how the data is distributed across the different channels. Sectors can be hidden if needed.
- The **line and bar charts** display the amount of data packages retrieved depending on its submission date and time duration.

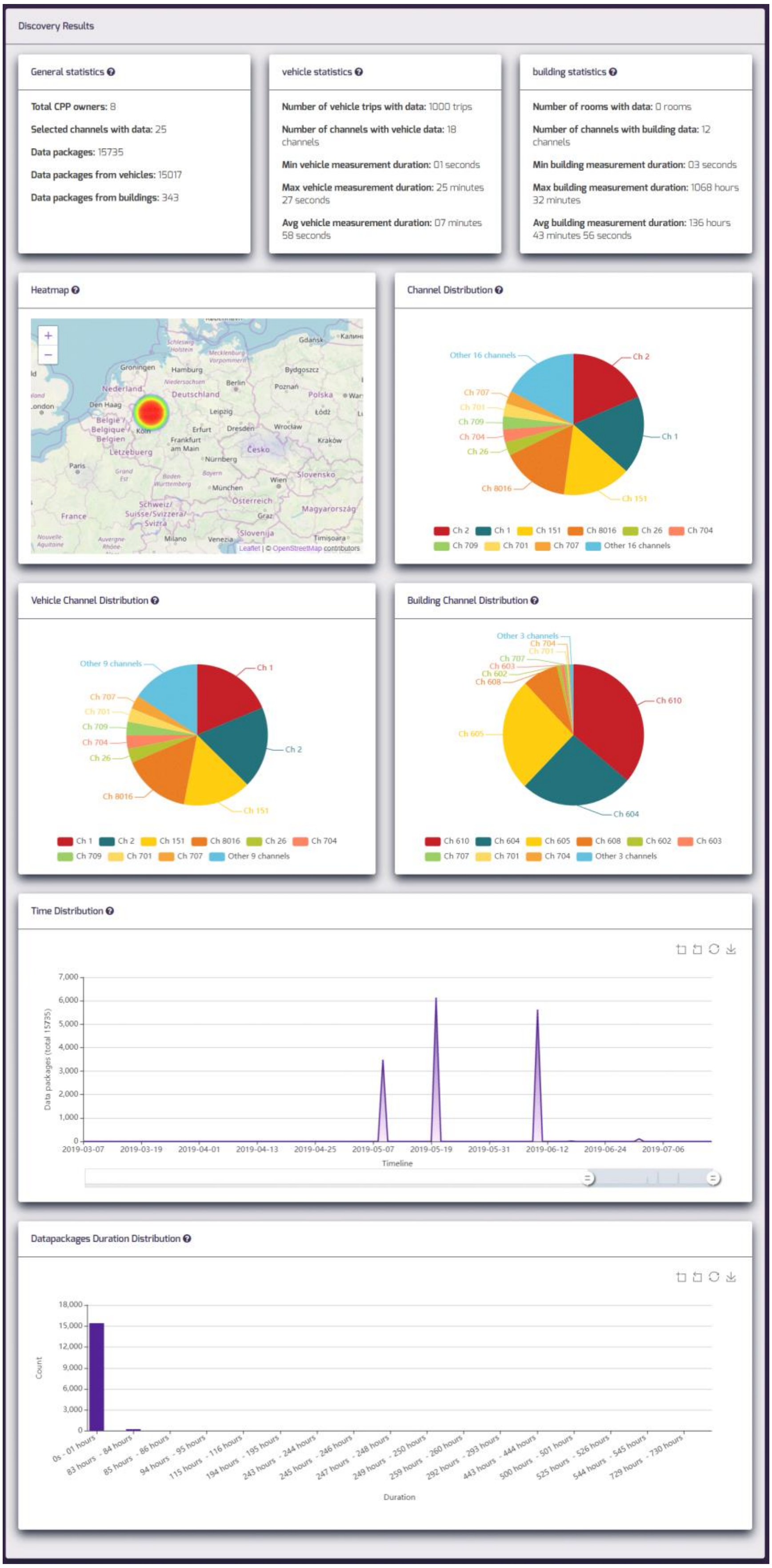


Figure 10. Data Discovery results

3. Management Panel

The Management Panel group the different functionalities for platform administration:

- Manage signals and channels
  - List available signals and channels
  - Review each signal and channel details
  - Create, update and delete signals and channels
- Manage users
  - List available users
  - Review each user details
  - Create and update users
- Check Data Requests, Analytics and Data Views
  - List existing Data Requests, analytics and Data Views
  - Check their configuration
  - Validate their integrity and functioning

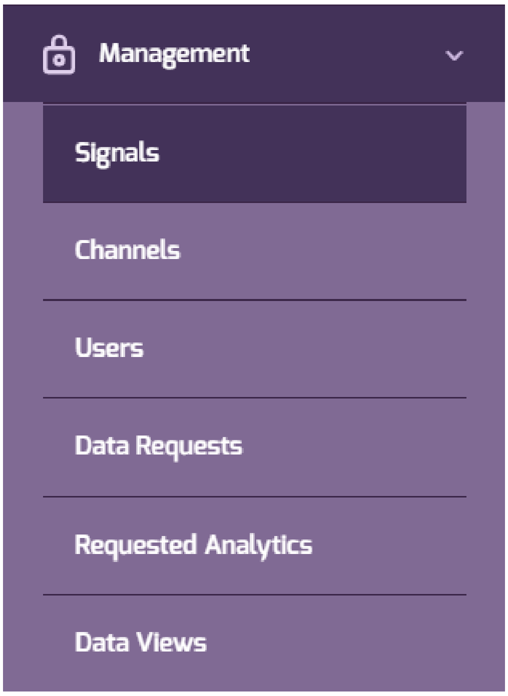
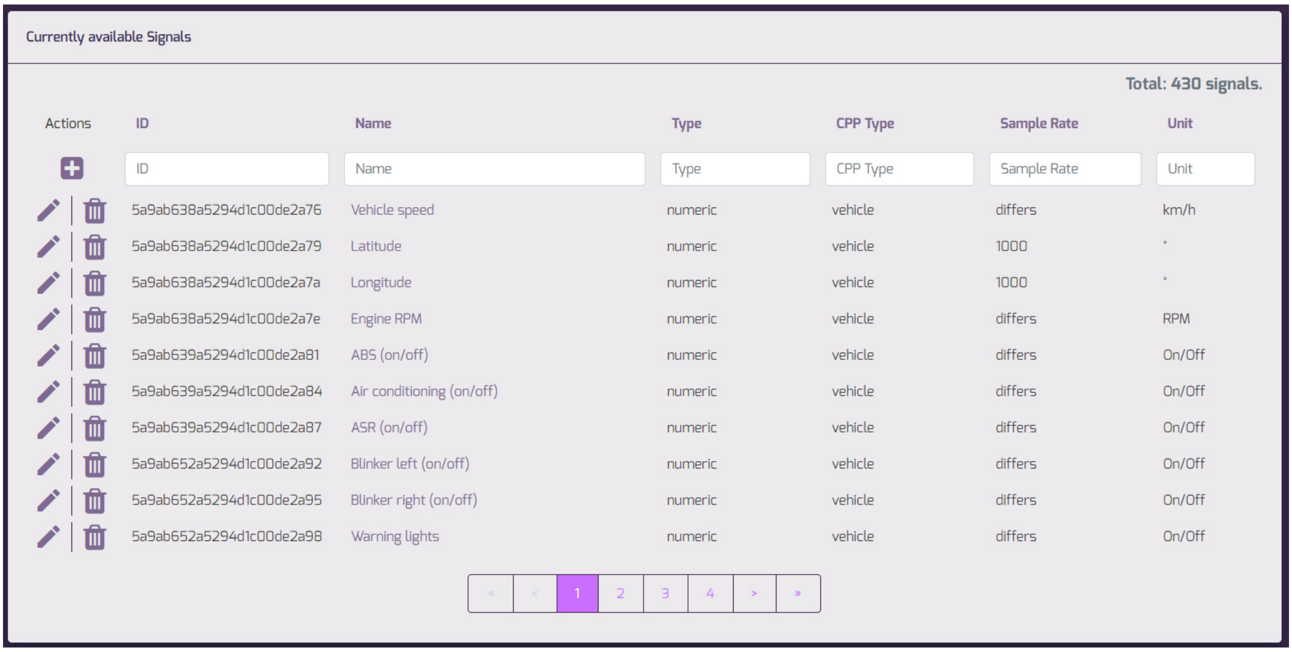


Figure 11. Management menu

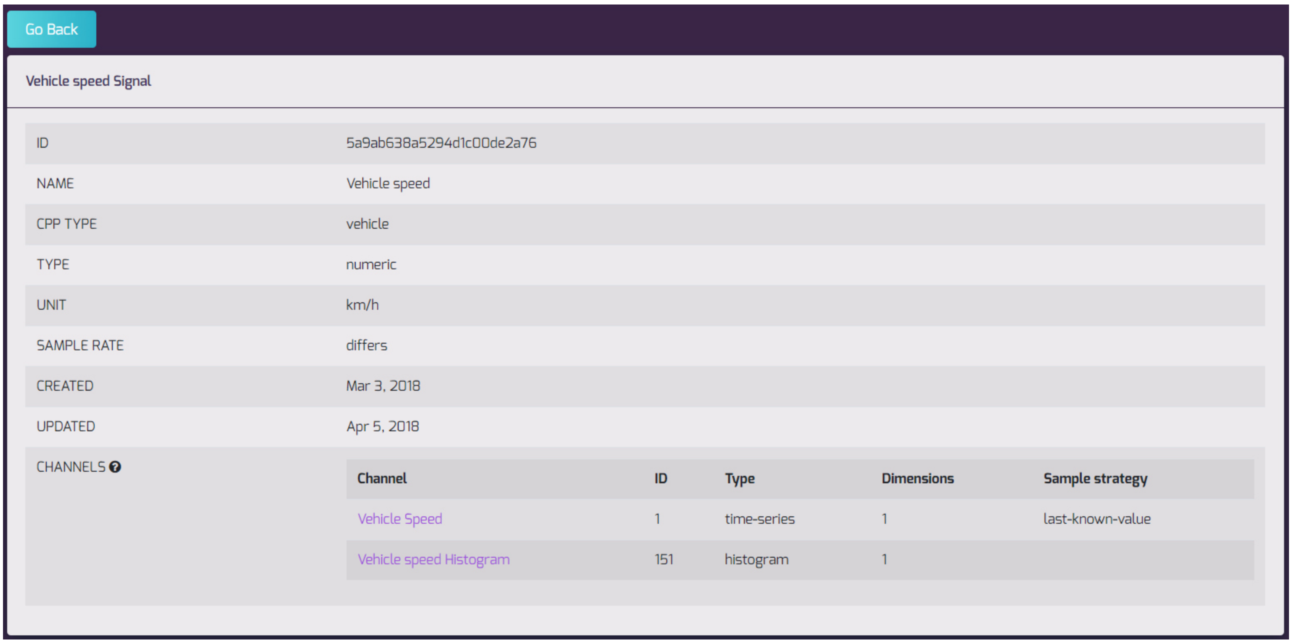
3.1. Data Signals

The administrator can find a list of the currently available signals in the Signals menu (Figure 12. Signals list). Clicking on a signal name will lead to that signal details view.



Actions	ID	Name	Type	CPP Type	Sample Rate	Unit
	ID	Name	Type	CPP Type	Sample Rate	Unit
	5a9ab638a5294d1c00de2a76	Vehicle speed	numeric	vehicle	differs	km/h
	5a9ab638a5294d1c00de2a79	Latitude	numeric	vehicle	1000	°
	5a9ab638a5294d1c00de2a7a	Longitude	numeric	vehicle	1000	°
	5a9ab638a5294d1c00de2a7e	Engine RPM	numeric	vehicle	differs	RPM
	5a9ab639a5294d1c00de2a81	ABS (on/off)	numeric	vehicle	differs	On/Off
	5a9ab639a5294d1c00de2a84	Air conditioning (on/off)	numeric	vehicle	differs	On/Off
	5a9ab639a5294d1c00de2a87	ASR (on/off)	numeric	vehicle	differs	On/Off
	5a9ab652a5294d1c00de2a92	Blinker left (on/off)	numeric	vehicle	differs	On/Off
	5a9ab652a5294d1c00de2a95	Blinker right (on/off)	numeric	vehicle	differs	On/Off
	5a9ab652a5294d1c00de2a98	Warning lights	numeric	vehicle	differs	On/Off

Figure 12. Signals list



ID	5a9ab638a5294d1c00de2a76
NAME	Vehicle speed
CPP TYPE	vehicle
TYPE	numeric
UNIT	km/h
SAMPLE RATE	differs
CREATED	Mar 3, 2018
UPDATED	Apr 5, 2018
CHANNELS	

Channel	ID	Type	Dimensions	Sample strategy
Vehicle Speed	1	time-series	1	last-known-value
Vehicle speed Histogram	151	histogram	1	

Figure 13. Signal details

3.1.1. How to create a signal?

To start the creation process, click on the + button in the top left corner of the list.

Signals can be created in two ways: form or json schema.

The creation form will lead the administrator through the whole process, showing the different necessary fields accordingly to selected options (such as signal type being numeric or for information).

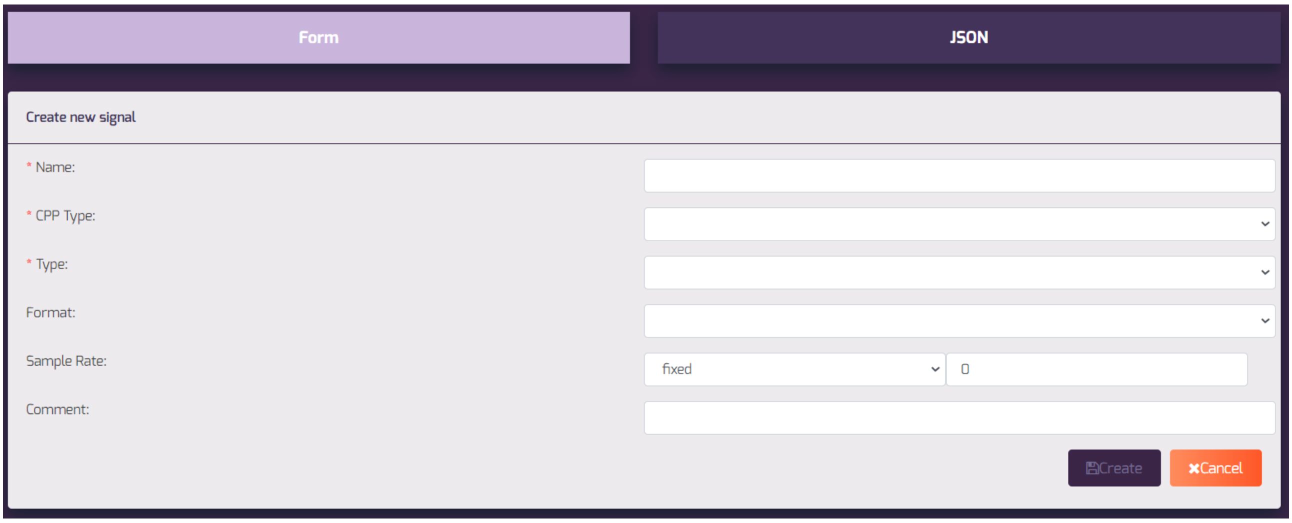


Figure 14. Create signal – empty creation form

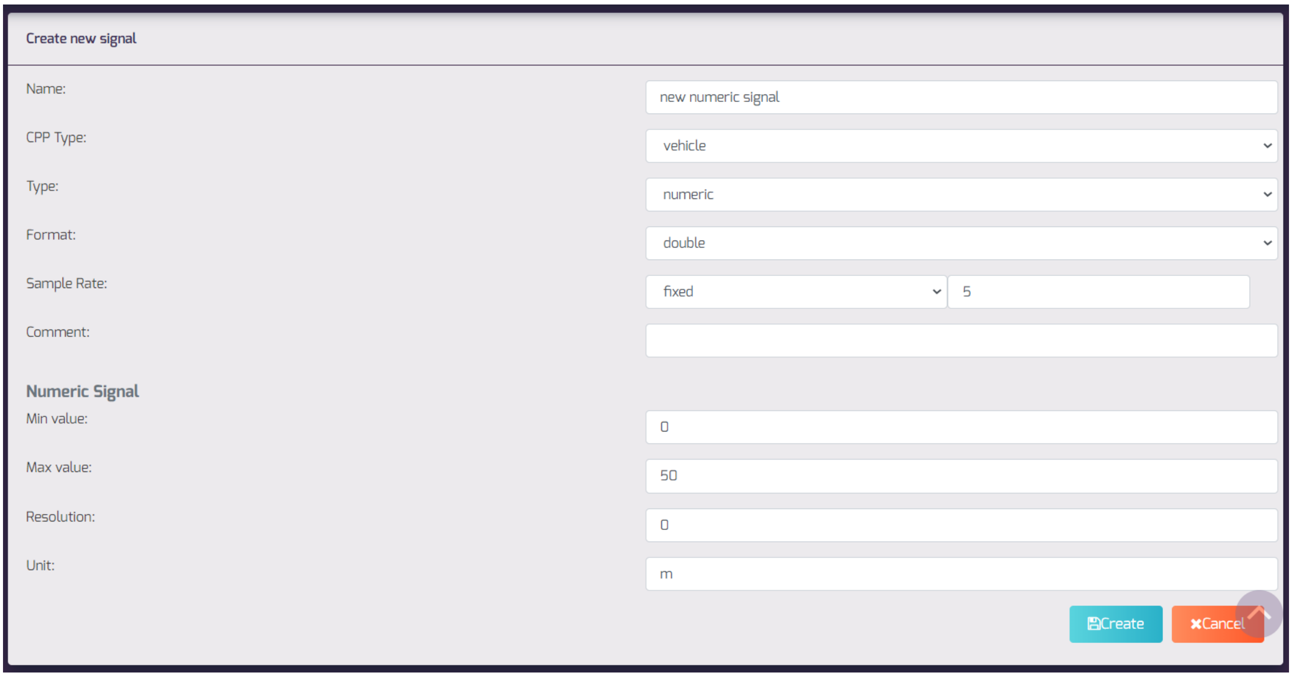


Figure 15. Create signal – new numeric signal form example

If the administrator has experience the signal can be created through the JSON form. There an empty text field is offered to paste a json object. This object has to be validated before being sent to be created.

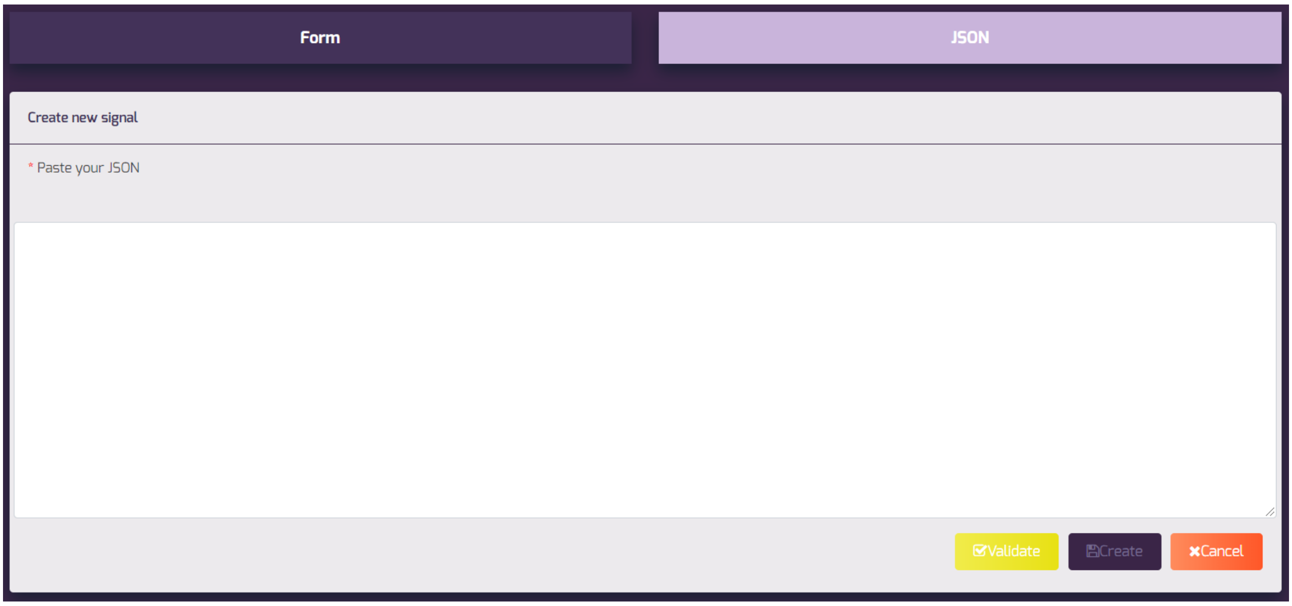


Figure 16. Create signal – empty creation json field

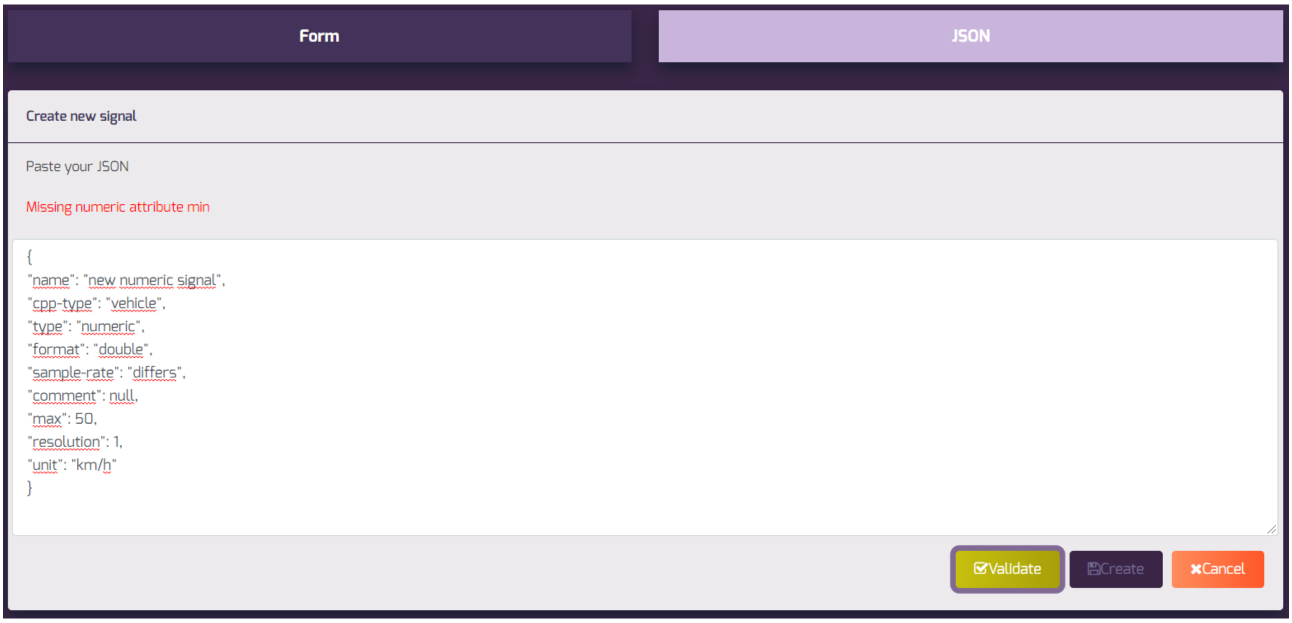


Figure 17. Create signal – new numeric signal bad json example

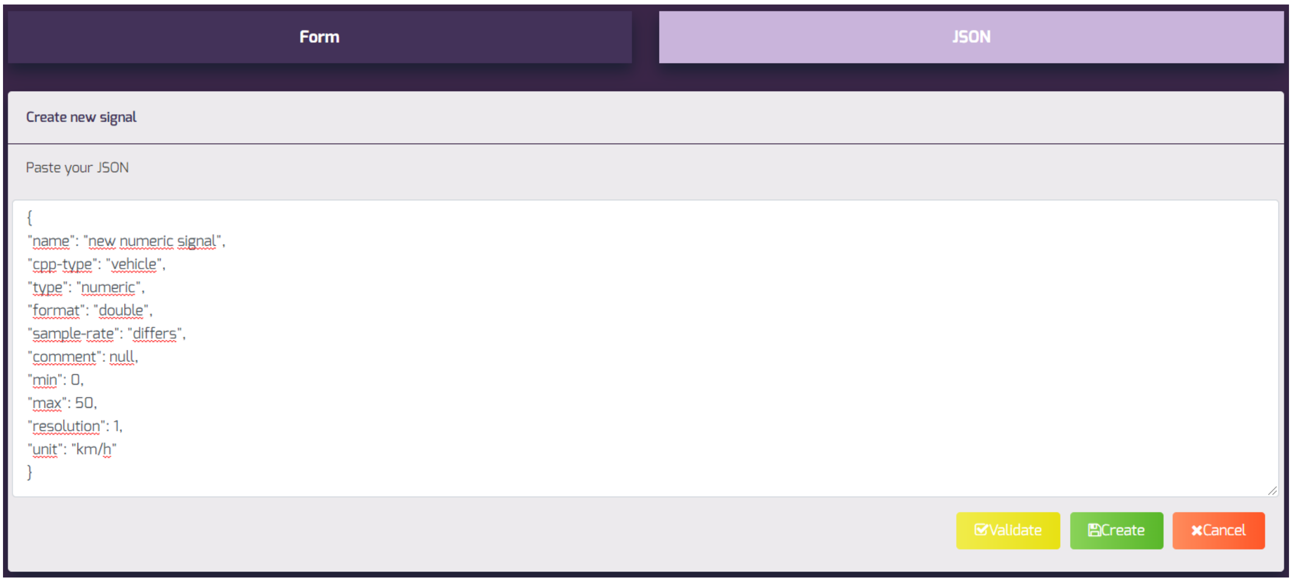


Figure 18. Create signal – new numeric signal validated json example

3.1.2. Update signals

Signals can be updated by clicking on the pencil icon at the left of the signal. A form like the creation one will be opened, with some changes:

- Signals can be updated only in form mode
- The type of the signal cannot be changed

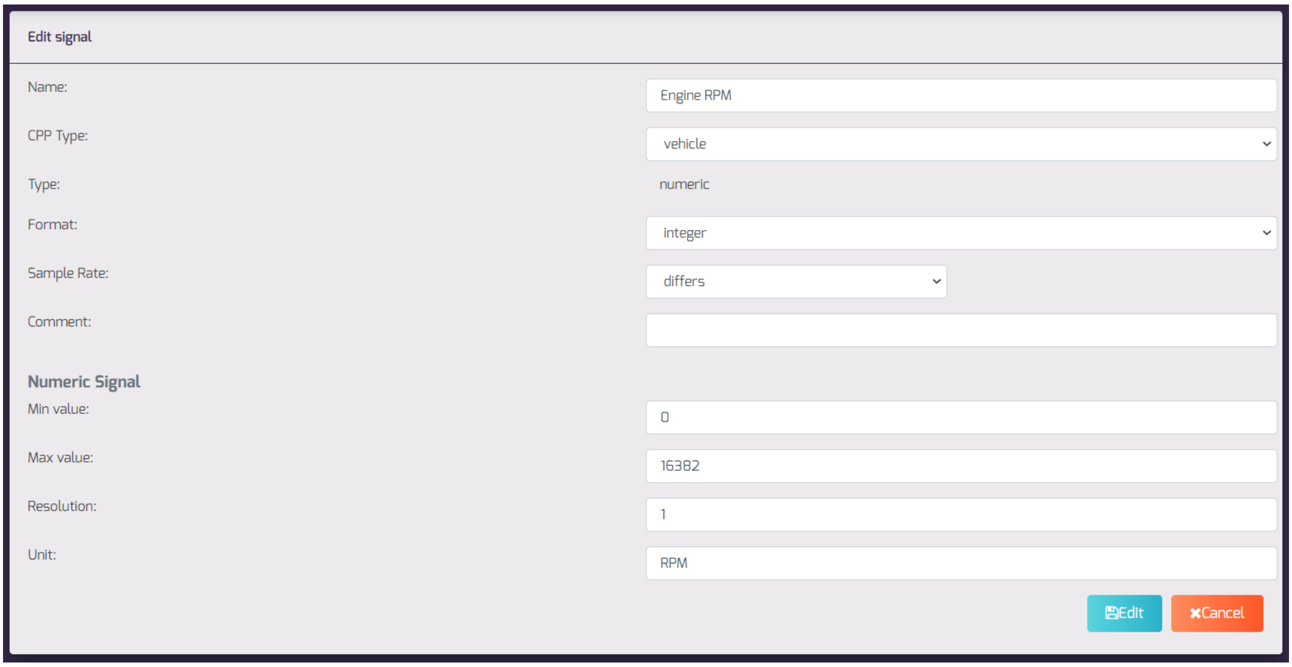


Figure 19. Update signal form

3.1.3. Delete Signals

Signals can be deleted from the system by clicking on the trash icon at the left of the signal. A signal can be deleted only if it is not being sampled by a measurement channel at the moment.

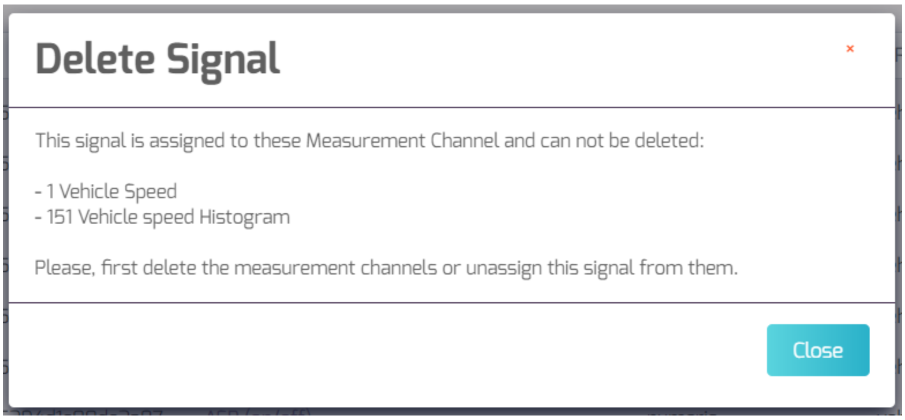


Figure 20. Delete signal used in measurement channels

If the signal is eligible to be deleted a message will be displayed asking for confirmation, as this operation cannot be undone.

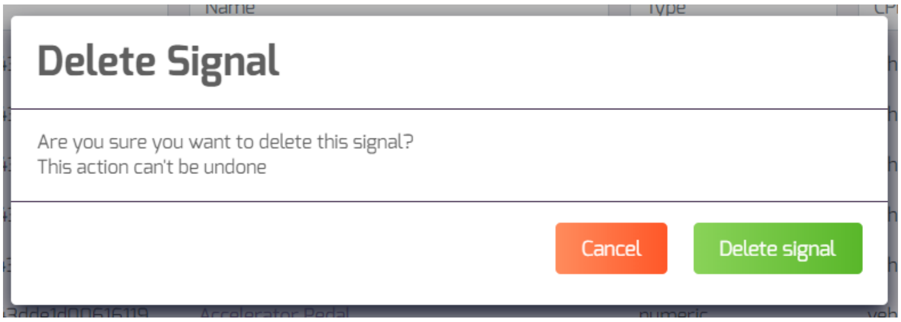
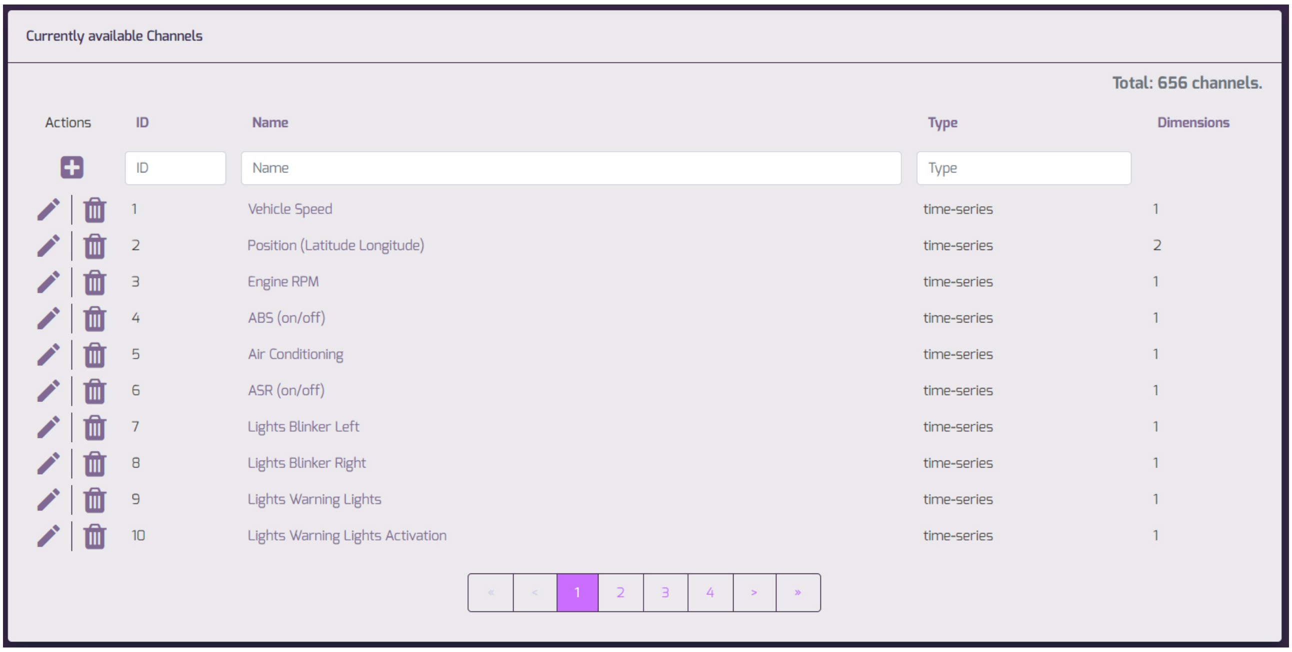


Figure 21. Delete signal confirmation

3.2. Measurement Channels

The administrator can find a list of the currently available Measurement Channels in the Channels menu (Figure 22. Measurement Channels list). Clicking on a measurement channel name will lead to that measurement channel details view.



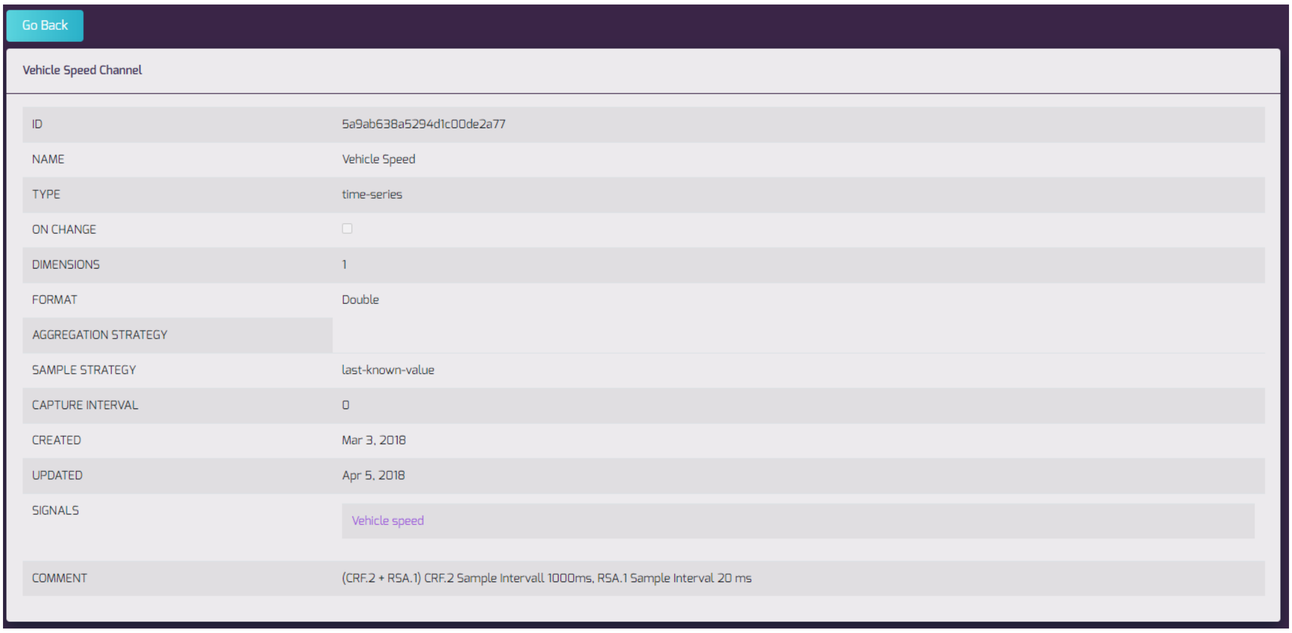
Currently available Channels

Total: 656 channels.

Actions	ID	Name	Type	Dimensions
	1	Vehicle Speed	time-series	1
	2	Position (Latitude Longitude)	time-series	2
	3	Engine RPM	time-series	1
	4	ABS (on/off)	time-series	1
	5	Air Conditioning	time-series	1
	6	ASR (on/off)	time-series	1
	7	Lights Blinker Left	time-series	1
	8	Lights Blinker Right	time-series	1
	9	Lights Warning Lights	time-series	1
	10	Lights Warning Lights Activation	time-series	1

« ‹ 1 2 3 4 › »

Figure 22. Measurement Channels list



Go Back

Vehicle Speed Channel

ID	5a9ab638a5294d1c00de2a77
NAME	Vehicle Speed
TYPE	time-series
ON CHANGE	
DIMENSIONS	1
FORMAT	Double
AGGREGATION STRATEGY	
SAMPLE STRATEGY	last-known-value
CAPTURE INTERVAL	0
CREATED	Mar 3, 2018
UPDATED	Apr 5, 2018
SIGNALS	Vehicle speed
COMMENT	(CRF.2 + RSA.1) CRF.2 Sample Intervall 1000ms, RSA.1 Sample Interval 20 ms

Figure 23. Measurement Channel details

3.2.1. How to create a channel?

To start the creation process, click on the + button in the top left corner of the list.

The Measurement creation process is very similar to the Signal creation one. Measurement Channels can also be created in two ways: form or json schema.

The creation form will lead the administrator through the whole process, showing the different necessary fields accordingly to selected options (such as measurement channel type being time-series or basic-cpp-information).

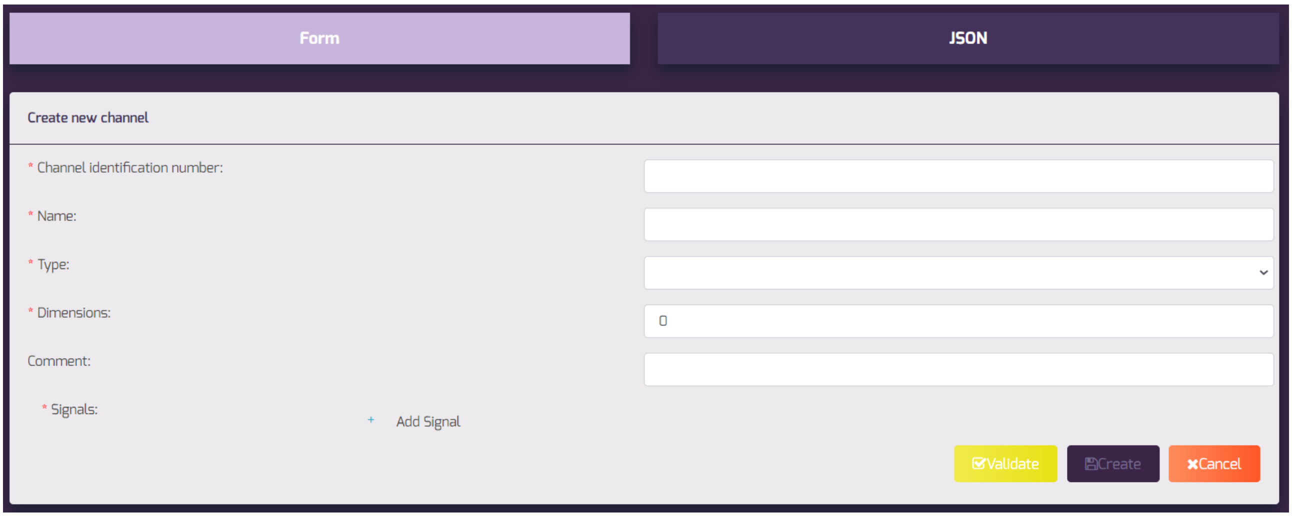


Figure 24. Create Measurement Channel – empty creation form

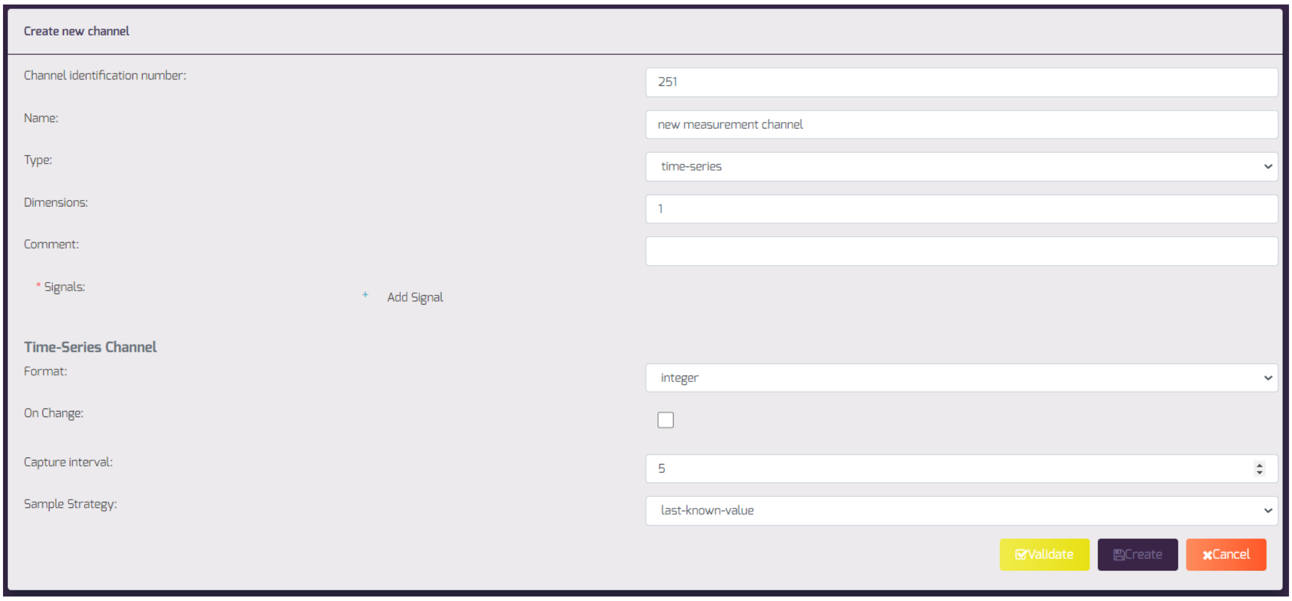


Figure 25. Create Measurement Channel – new time-series channel form example

There are two changes from to the signal creation:

- Measurement Channel form must be validated before being sent, much as the JSON creation form does
- When creating a Measurement Channel signals must be assigned to be sampled by the channel. The number of signals must be the value of the field "dimensions"

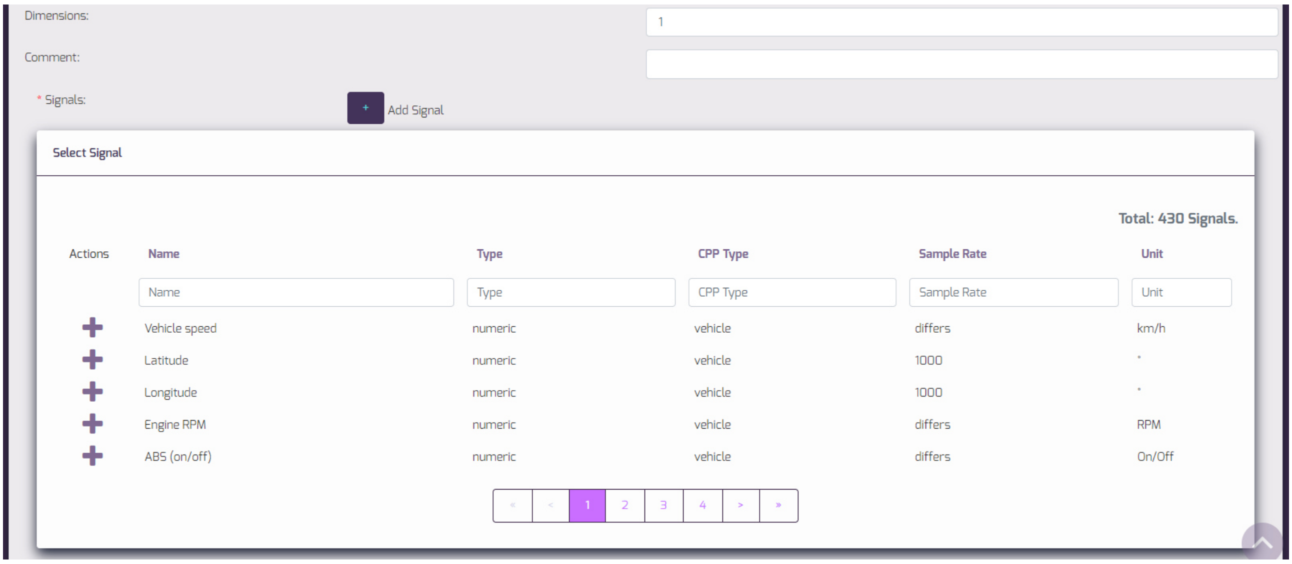


Figure 26. Create Measurement Channel – assign signal

If the administrator has experience the Measurement Channel can be created through the JSON form. There an empty text field is offered to paste a json object. This object has to be validated before being sent to be created.

There are some things to be considered when creating a Measurement Channel via JSON:

- Signal field must be an array of signal ids. Please first write down the ids of the signals to be included before starting the creation process.
- The attributes to be added accordingly to the Measurement Channel type must be included in an object which key is the type value

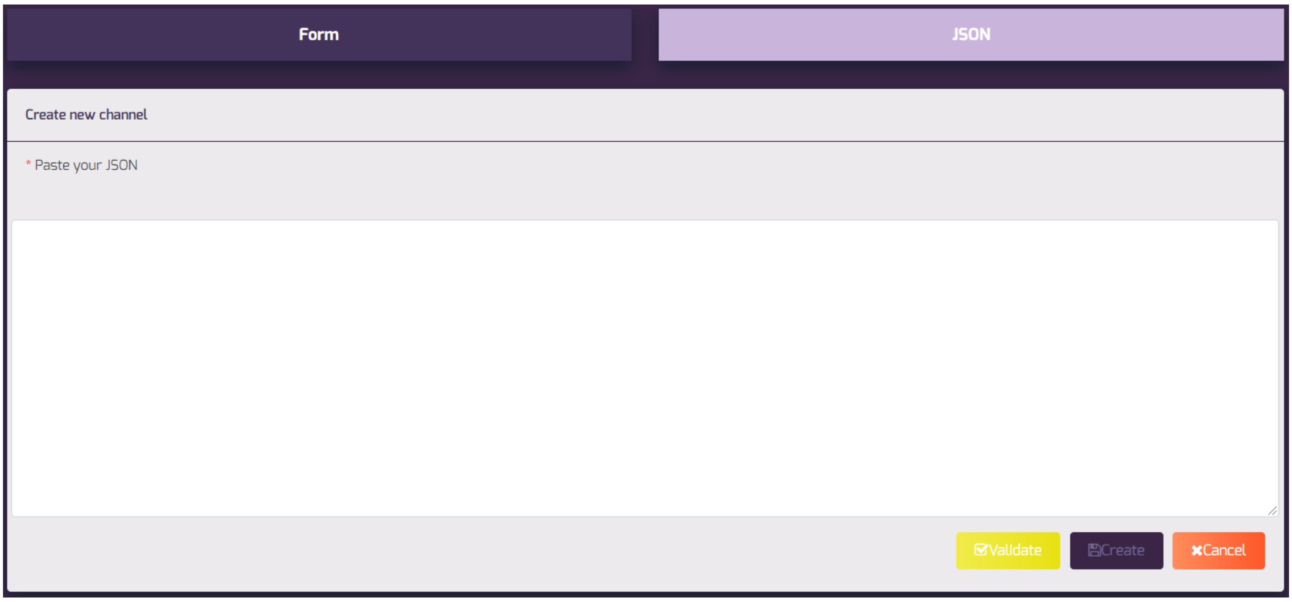


Figure 27. Create Measurement Channel – empty creation json field

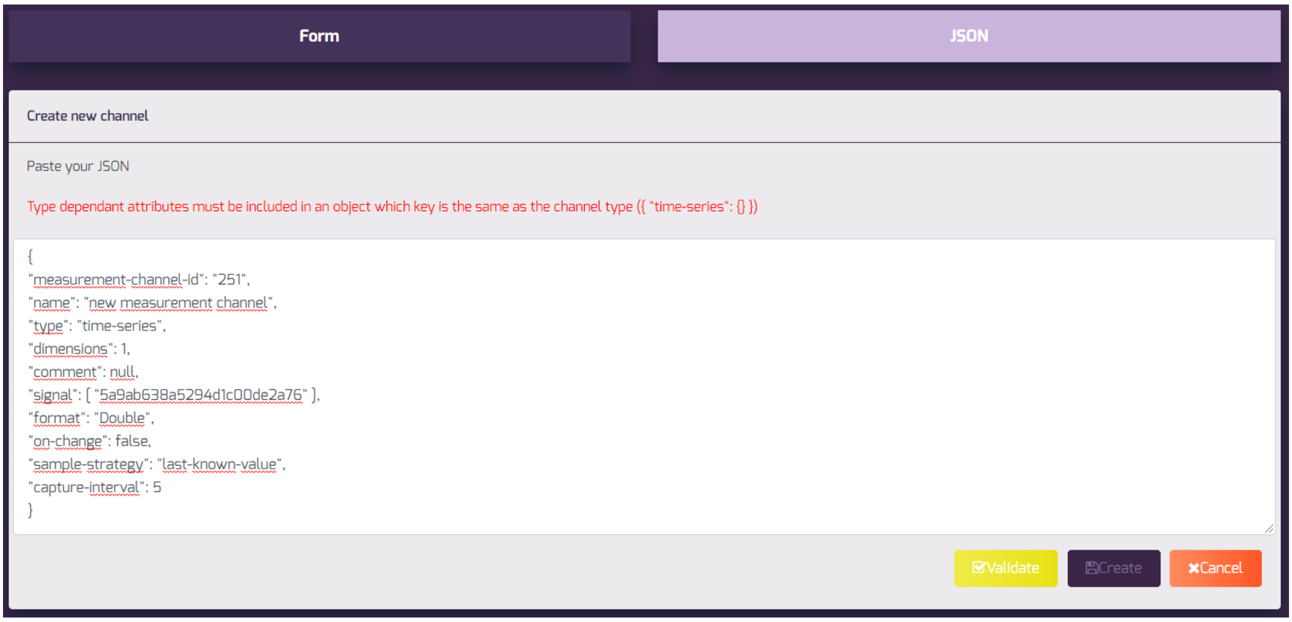


Figure 28. Create signal – new time-series channel bad json example

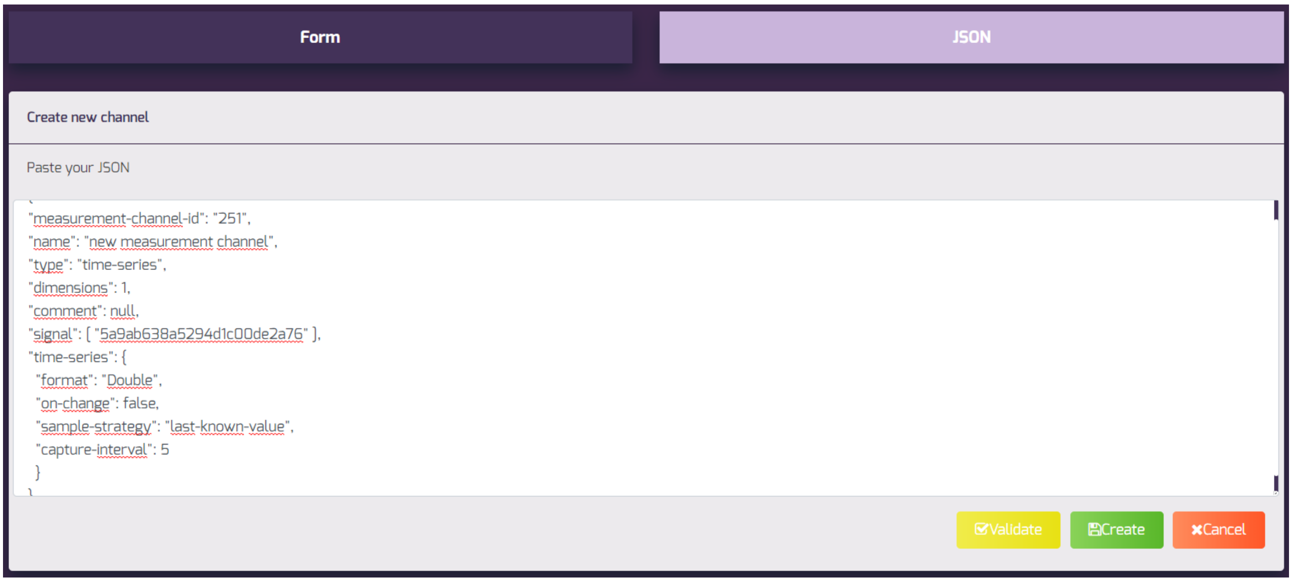


Figure 29. Create Measurement Channel – new time-series channel validated json example

3.2.2. Update channels

Measurement Channels can be updated by clicking on the pencil icon at the left of the channel. A form like the creation one will be opened, with some changes:

- Measurement Channels can be updated only in form mode
- The type of the Measurement Channel cannot be changed
- Signal number must be the same as the "dimensions" attribute value, but both can be changed
- Update form must be validated before being sent

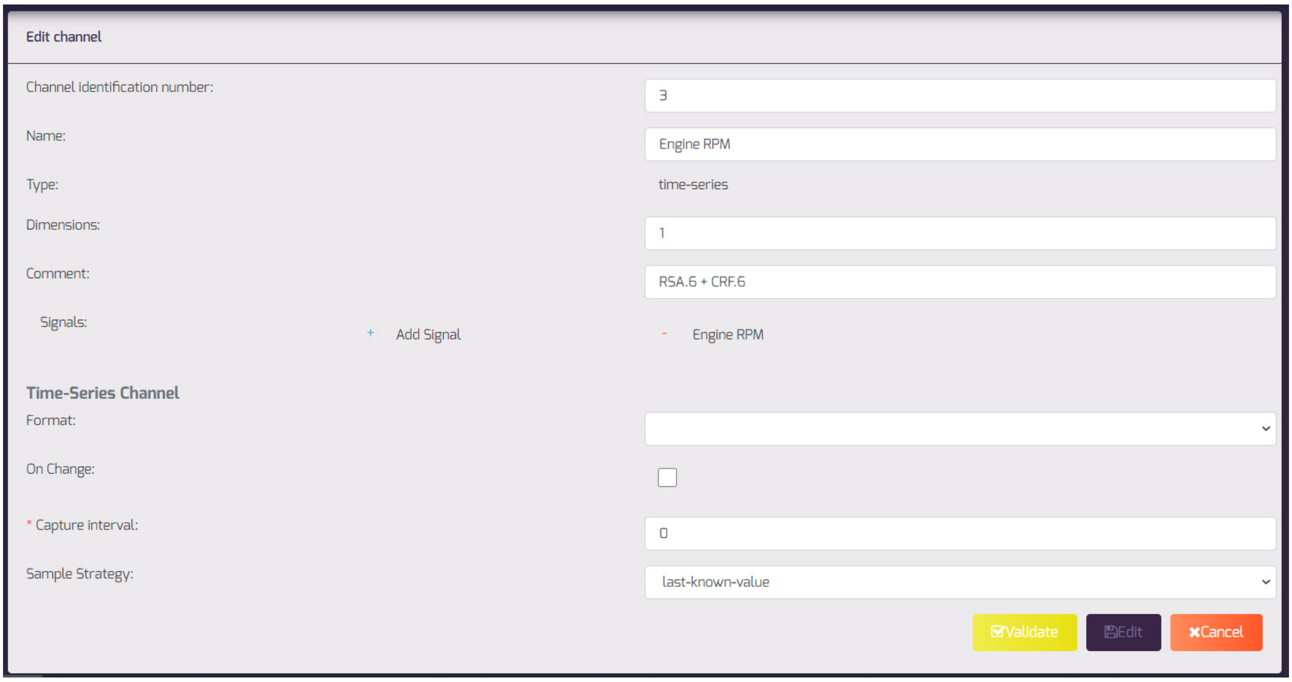


Figure 30. Update signal form

3.2.3. Delete channels

Measurement Channels can be deleted from the system by clicking on the trash icon at the left of the channel. A message will be displayed asking for confirmation, as this operation cannot be undone.

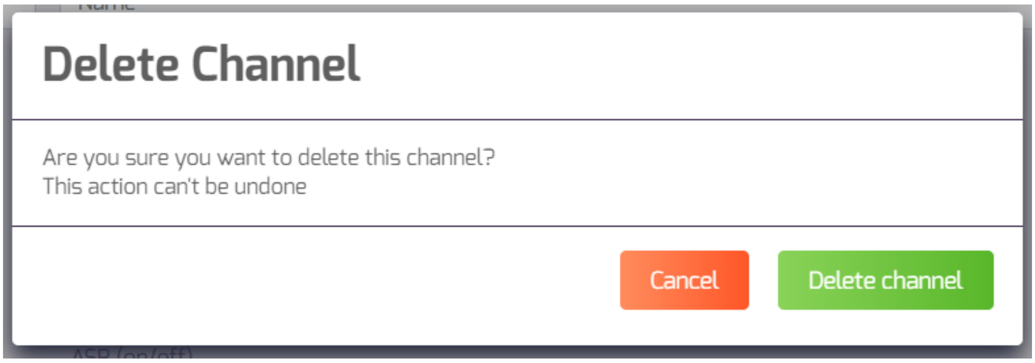
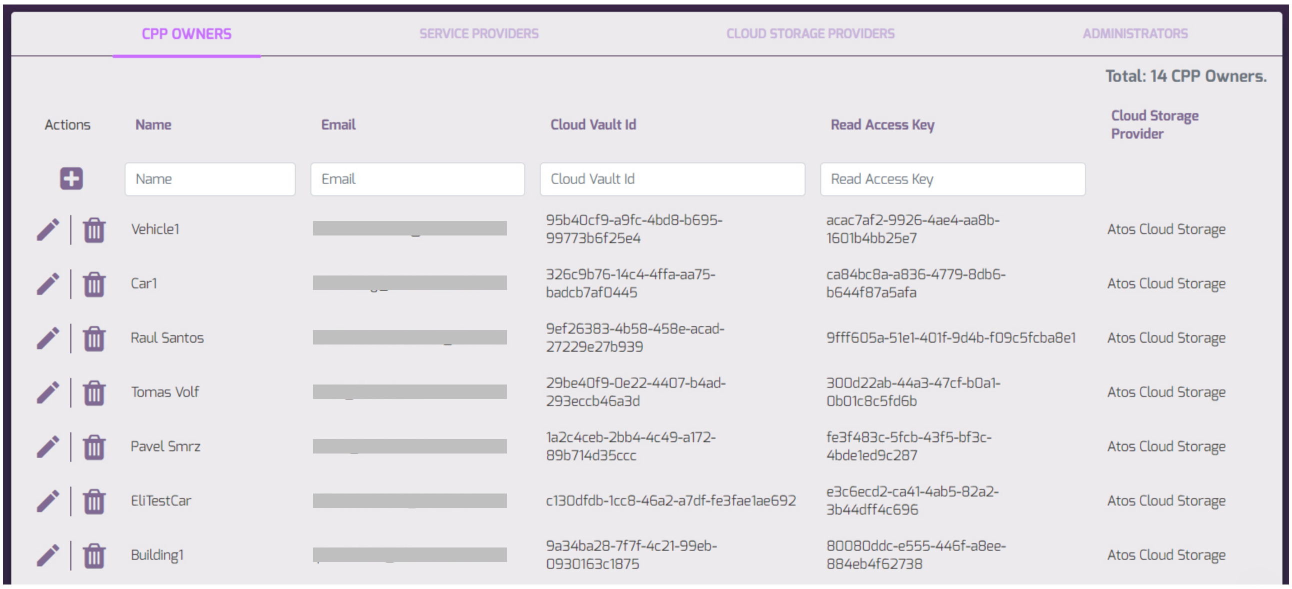


Figure 31. Delete Measurement Channel confirmation

3.3. Platform Users

The administrator can find a list of the currently registered users in the Users menu (Figure 32. Users list). The top bar of the list shows the different roles within the Cross-CPP Marketplace system. Selecting one will switch the list to display the users from that role.



CPP OWNERS | SERVICE PROVIDERS | CLOUD STORAGE PROVIDERS | ADMINISTRATORS

Total: 14 CPP Owners.

Actions	Name	Email	Cloud Vault Id	Read Access Key	Cloud Storage Provider
	Name	Email	Cloud Vault Id	Read Access Key	
	Vehicle1	[redacted]	95b40cf9-a9fc-4bd8-b695-99773b6f25e4	acac7af2-9926-4ae4-aa8b-1601b4bb25e7	Atos Cloud Storage
	Car1	[redacted]	326c9b76-14c4-4ffa-aa75-badcb7af0445	ca84bc8a-a836-4779-8db6-b644f87a5afa	Atos Cloud Storage
	Raul Santos	[redacted]	9ef26383-4b58-458e-acad-27229e27b939	9fff605a-51e1-401f-9d4b-f09c5fcba8e1	Atos Cloud Storage
	Tomas Volf	[redacted]	29be40f9-0e22-4407-b4ad-293eccb46a3d	300d22ab-44a3-47cf-b0a1-0b01c8c5fd6b	Atos Cloud Storage
	Pavel Smrz	[redacted]	1a2c4ceb-2bb4-4c49-a172-89b714d35ccc	fe3f483c-5fcb-43f5-bf3c-4bde1ed9c287	Atos Cloud Storage
	EliTestCar	[redacted]	c130dfdb-1cc8-46a2-a7df-fe3fae1ae692	e3c6ecd2-ca41-4ab5-82a2-3b44dff4c696	Atos Cloud Storage
	Building1	[redacted]	9a34ba28-7f7f-4c21-99eb-0930163c1875	80080ddc-e555-446f-a8ee-884eb4f62738	Atos Cloud Storage

Figure 32. Users list

3.3.1. General considerations about users

There are five types of user depending on the role they have:

- vehicleOwner (cpp owner / data owner)
- serviceProvider (service provider / data consumer)
- cloudStorageProvider (cloud storage owner)
- dataProvider (data provider / OEM)
- provider (administrator)

Each of them has its own functionalities within the platform. The role of platform administrator includes managing them, although each of them has its own actions to be performed.

3.3.2. How to create a new user?

Users can create their profile through the Sign-Up form of the Cross-CPP Marketplace landing page. However, this operation can also be done by a system administrator.

In any of the role lists, by clicking the + icon an especial row will be displayed letting the administrator to fill the fields of a new user. All fields are mandatory. Clicking on the check icon on the left will send the user to be created in the system. The list will be automatically updated.

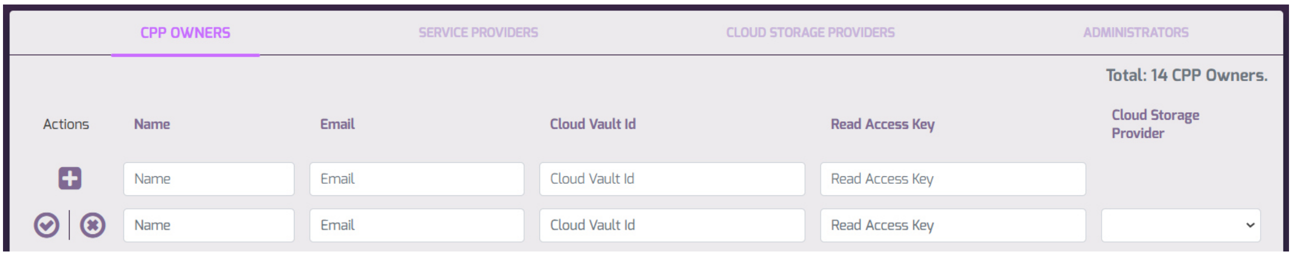


Figure 33. Create new user form

3.3.3. Validate users

Service Providers and Cloud Storage Providers creation require a validation from a system administrator. In each list a button displaying the currently awaiting requests is displayed. The button will not appear if there is no pending request.

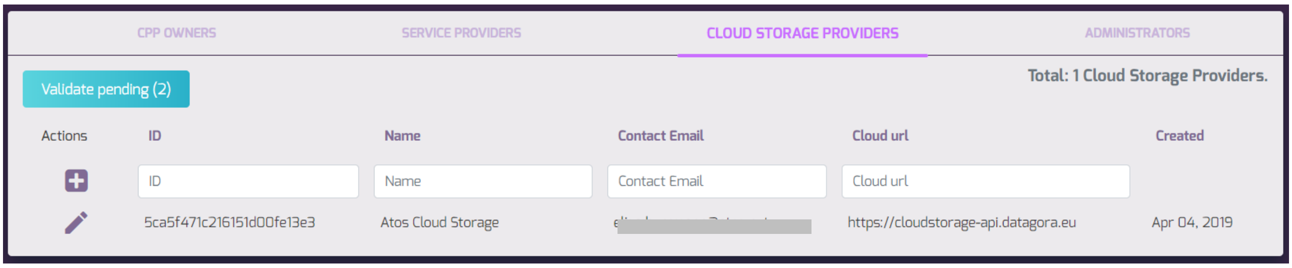


Figure 34. Validate pending user request button

If the administrator validates a user, it is sent to the IdM. Once the IdM responds with a valid user the platform notifies the server and the credentials are sent to the user's email. This means the only step necessary from the administrator point of view is validating pending users.

To validate a user request the administrator only need to click on the Validate button. On the contrary a request can also be declined by clicking on the Reject button. In either case a confirmation will be sent to the requesting user.

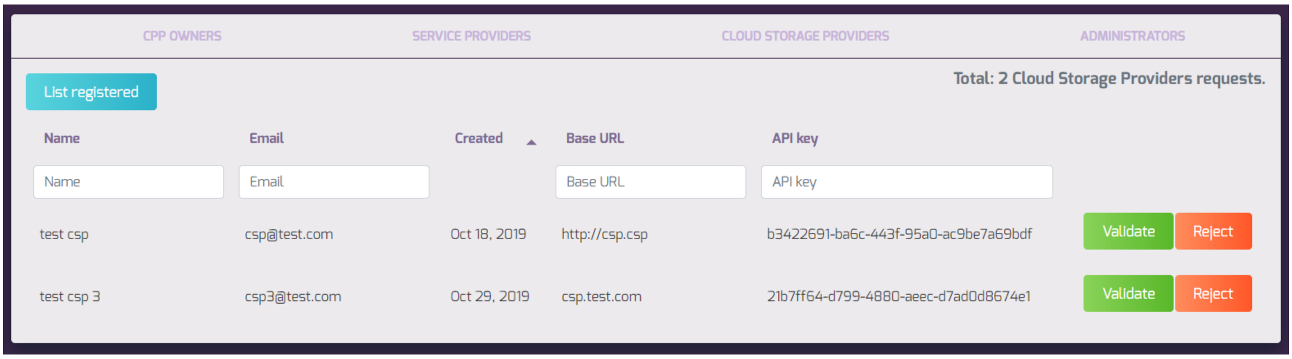


Figure 35. List of users awaiting validation

3.3.4. Update users

An user can be edited by clicking on the pencil icon at the left f the user and filling the fields.

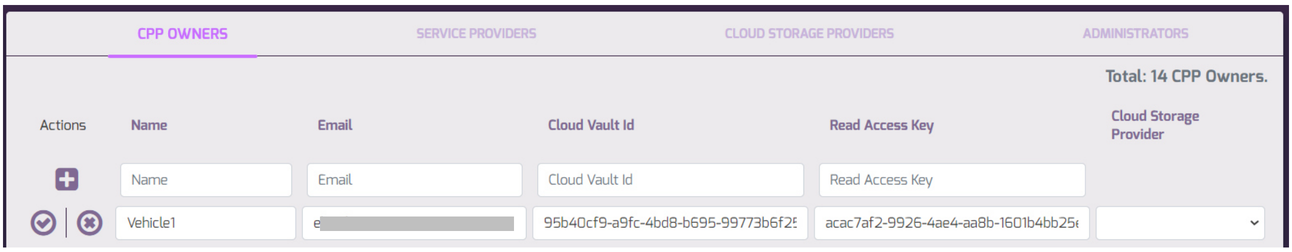


Figure 36. Update user form

3.3.5. Delete users

Users can be deleted from the system by clicking on the trash icon at the left of the user. A message will be displayed asking for confirmation, as this operation cannot be undone.

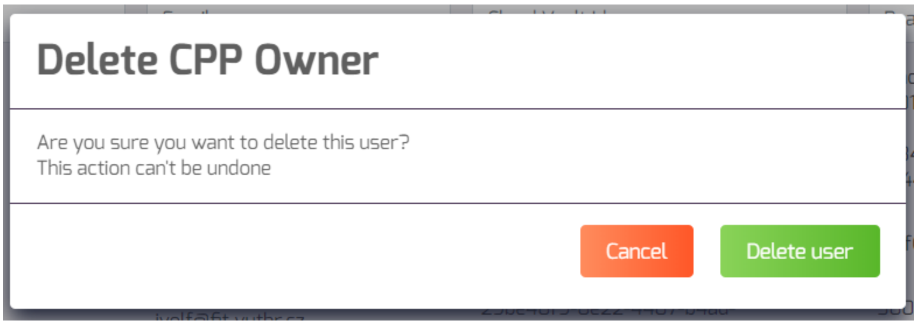
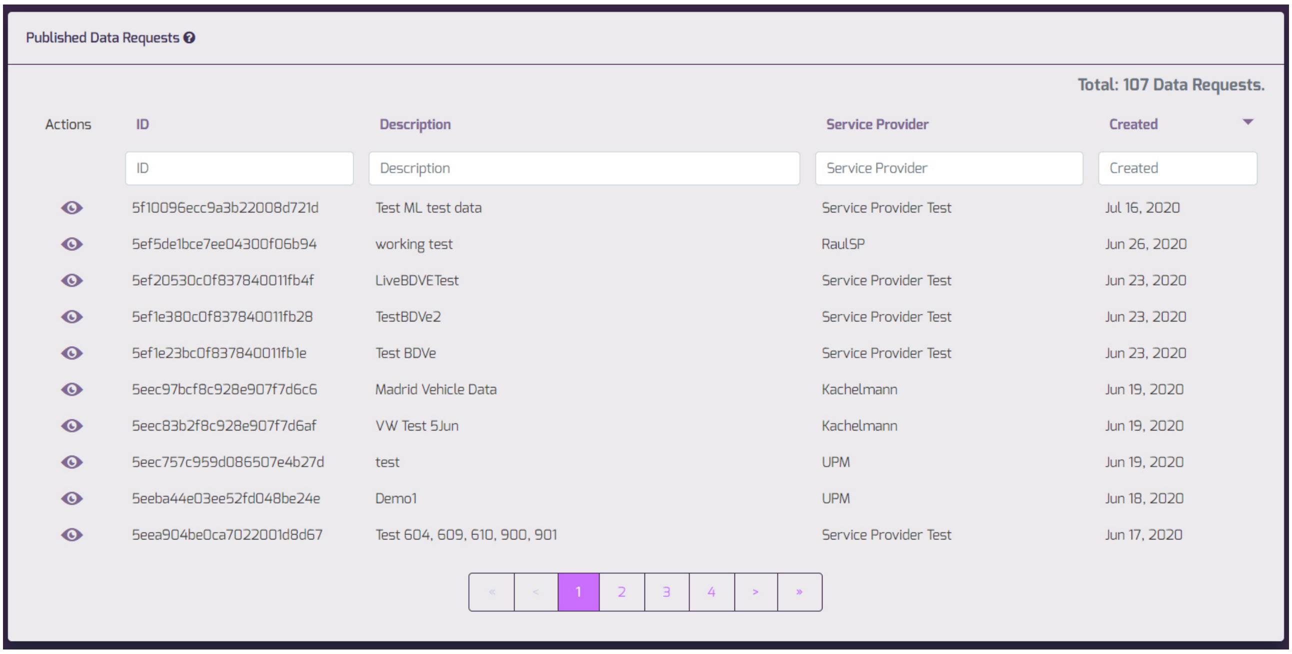


Figure 37. Delete user confirmation

3.4. Data Requests, Requested Analytics and Data Views

An administrator can use these views to review how many of them are created and in use in the system at any time. The main purpose of this set of views is to check the proper functioning of each of the different services offered by Cross-CPP Marketplace. In the Requested Analytics List can be found all requested analytics divided by the type of the analytic (time series, trajectories, network…).

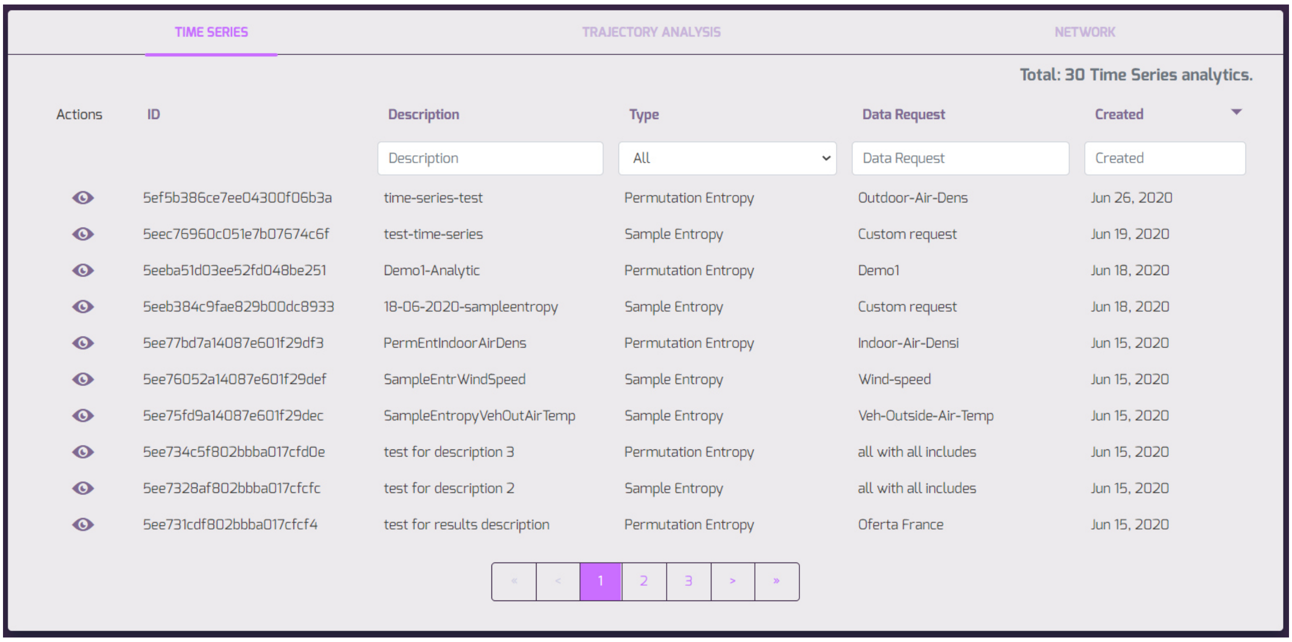


Published Data Requests

Total: 107 Data Requests.

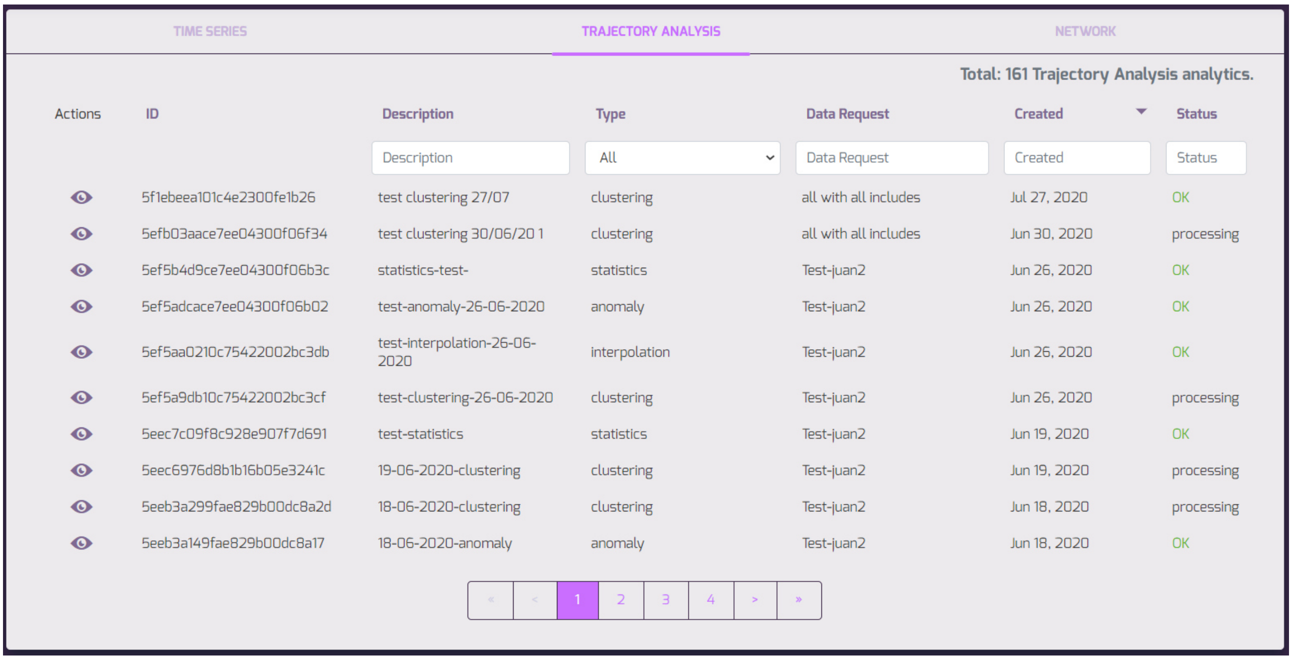
Actions	ID	Description	Service Provider	Created
	5f10096ecc9a3b22008d721d	Test ML test data	Service Provider Test	Jul 16, 2020
	5ef5de1bce7ee04300f06b94	working test	RaulSP	Jun 26, 2020
	5ef20530c0f837840011fb4f	LiveBDVETest	Service Provider Test	Jun 23, 2020
	5ef1e380c0f837840011fb28	TestBDVe2	Service Provider Test	Jun 23, 2020
	5ef1e23bc0f837840011fb1e	Test BDVe	Service Provider Test	Jun 23, 2020
	5eec97bcf8c928e907f7d6c6	Madrid Vehicle Data	Kachelmann	Jun 19, 2020
	5eec83b2f8c928e907f7d6af	VW Test 5Jun	Kachelmann	Jun 19, 2020
	5eec757c959d086507e4b27d	test	UPM	Jun 19, 2020
	5eeba44e03ee52fd048be24e	Demo1	UPM	Jun 18, 2020
	5eea904be0ca7022001d8d67	Test 604, 609, 610, 900, 901	Service Provider Test	Jun 17, 2020

Figure 38. Data Requests list



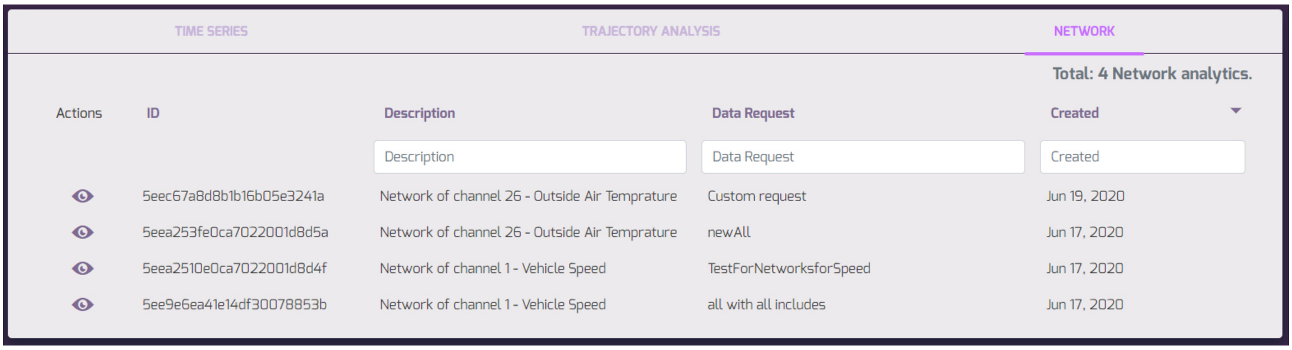
Actions	ID	Description	Type	Data Request	Created
		Description	All	Data Request	Created
	5ef5b386ce7ee04300f06b3a	time-series-test	Permutation Entropy	Outdoor-Air-Dens	Jun 26, 2020
	5eec76960c051e7b07674c6f	test-time-series	Sample Entropy	Custom request	Jun 19, 2020
	5eeba51d03ee52fd048be251	Demo1-Analytic	Permutation Entropy	Demo1	Jun 18, 2020
	5eeb384c9fae829b00dc8933	18-06-2020-sampleentropy	Sample Entropy	Custom request	Jun 18, 2020
	5ee77bd7a14087e601f29df3	PermEntIndoorAirDens	Permutation Entropy	Indoor-Air-Densi	Jun 15, 2020
	5ee76052a14087e601f29def	SampleEntrWindSpeed	Sample Entropy	Wind-speed	Jun 15, 2020
	5ee75fd9a14087e601f29dec	SampleEntropyVehOutAirTemp	Sample Entropy	Veh-Outside-Air-Temp	Jun 15, 2020
	5ee734c5f802bbba017cfd0e	test for description 3	Permutation Entropy	all with all includes	Jun 15, 2020
	5ee7328af802bbba017cfcfc	test for description 2	Sample Entropy	all with all includes	Jun 15, 2020
	5ee731cdf802bbba017cfcf4	test for results description	Permutation Entropy	Oferta France	Jun 15, 2020

Figure 39. Requested Time Series Analytics list



Actions	ID	Description	Type	Data Request	Created	Status
		Description	All	Data Request	Created	Status
	5f1ebeea101c4e2300fe1b26	test clustering 27/07	clustering	all with all includes	Jul 27, 2020	OK
	5efb03aace7ee04300f06f34	test clustering 30/06/20 1	clustering	all with all includes	Jun 30, 2020	processing
	5ef5b4d9ce7ee04300f06b3c	statistics-test-	statistics	Test-juan2	Jun 26, 2020	OK
	5ef5adcace7ee04300f06b02	test-anomaly-26-06-2020	anomaly	Test-juan2	Jun 26, 2020	OK
	5ef5aa0210c75422002bc3db	test-interpolation-26-06-2020	interpolation	Test-juan2	Jun 26, 2020	OK
	5ef5a9db10c75422002bc3cf	test-clustering-26-06-2020	clustering	Test-juan2	Jun 26, 2020	processing
	5eec7c09f8c928e907f7d691	test-statistics	statistics	Test-juan2	Jun 19, 2020	OK
	5eec6976d8b1b16b05e3241c	19-06-2020-clustering	clustering	Test-juan2	Jun 19, 2020	processing
	5eeb3a299fae829b00dc8a2d	18-06-2020-clustering	clustering	Test-juan2	Jun 18, 2020	processing
	5eeb3a149fae829b00dc8a17	18-06-2020-anomaly	anomaly	Test-juan2	Jun 18, 2020	OK

Figure 40. Requested Trajectories analysis list



Actions	ID	Description	Data Request	Created
		Description	Data Request	Created
	5eec67a8d8b1b16b05e3241a	Network of channel 26 - Outside Air Temprature	Custom request	Jun 19, 2020
	5eea253fe0ca7022001d8d5a	Network of channel 26 - Outside Air Temprature	newAll	Jun 17, 2020
	5eea2510e0ca7022001d8d4f	Network of channel 1 - Vehicle Speed	TestForNetworksforSpeed	Jun 17, 2020
	5ee9e6ea41e14df30078853b	Network of channel 1 - Vehicle Speed	all with all includes	Jun 17, 2020

Figure 41. Requested Networks list

Created Data Views

Total: 13 data views

Actions	ID	Description	Offer	Created
		Description		Created
	5efae582ce7ee04300f06f03	test with channel 608	all with all includes	Jun 30, 2020
	5ef5ad9dce7ee04300f06b00	test with time series vehicle speed and RPM with no filter and an event based channel	all with all includes	Jun 26, 2020
	5eec8c4df8c928e907f7d6b2	Testx		Jun 19, 2020
	5eea9121e0ca7022001d8d68	SELECT * FROM THIS WHERE "901" = 1 AND "609" > 120	Test 604, 609, 610, 900, 901	Jun 17, 2020
	5ee735b7f802bbba017cfd18	select	eChargers Health 2	Jun 15, 2020
	5ede4761c4160514100822f2	humidity >23	Siemens outside Humidity	Jun 08, 2020
	5ede4230dba44efe0f2865b5	test	Siemens outside Humidity	Jun 08, 2020
	5ede21f5dba44efe0f2865ae	Test6	newAll	Jun 08, 2020
	5edde7841eb3f7900f8f0dd9	Test3	newAll	Jun 08, 2020
	5edde5101eb3f7900f8f0dd7	Test2	MyOffer	Jun 08, 2020

« ‹ 1 2 › »

Figure 42. Data Views list

Details of each one can be reviewed by clicking on the eye icon at the left. The details view is the same offered to the Service Providers with some additional information:

- All IDs are displayed so the administrator can check the database or use testing tools
- All AEON channels URLs are displayed so the administrator can test the different connections

For more information on the details view of each service please check the Service Provider User Guide in their respective section.

4. AEON

4.1. What is AEON

AEON is a cloud platform to create applications with real time communications channels. The architecture is based on the strongly communication needs that we need to face nowadays, with billions of interconnected devices and short times of response. Thus, the technological solutions used for the implementation are based on strong requirements about performance, response and scalability, making use of the most advanced cutting-edge technologies.

AEON platform offers a shared cloud-based message queuing framework enabling messaging between various entities that wish to communicate with each other seamlessly and reliably using standard vendor neutral protocols

Benefits:

- Communicate applications and services through a real time network
- Easy to use, easy to integrate in developments: AEON provides an SDK to connect your services and devices over a globally scaled real-time network
- Performance, Scalability and Reliability: High performance for message delivery and data exchange between business processes and devices and from device to device. AEON is able to handle multiple types and priorities of messages, whilst at the same time providing the necessary Quality of Service. AEON provides reliable messaging with durability and persistence and needs to scale well for extremely large volumes.
- Big Data: AEON can take care of the cloud messaging of the data capture from M2M environments and big data flows.

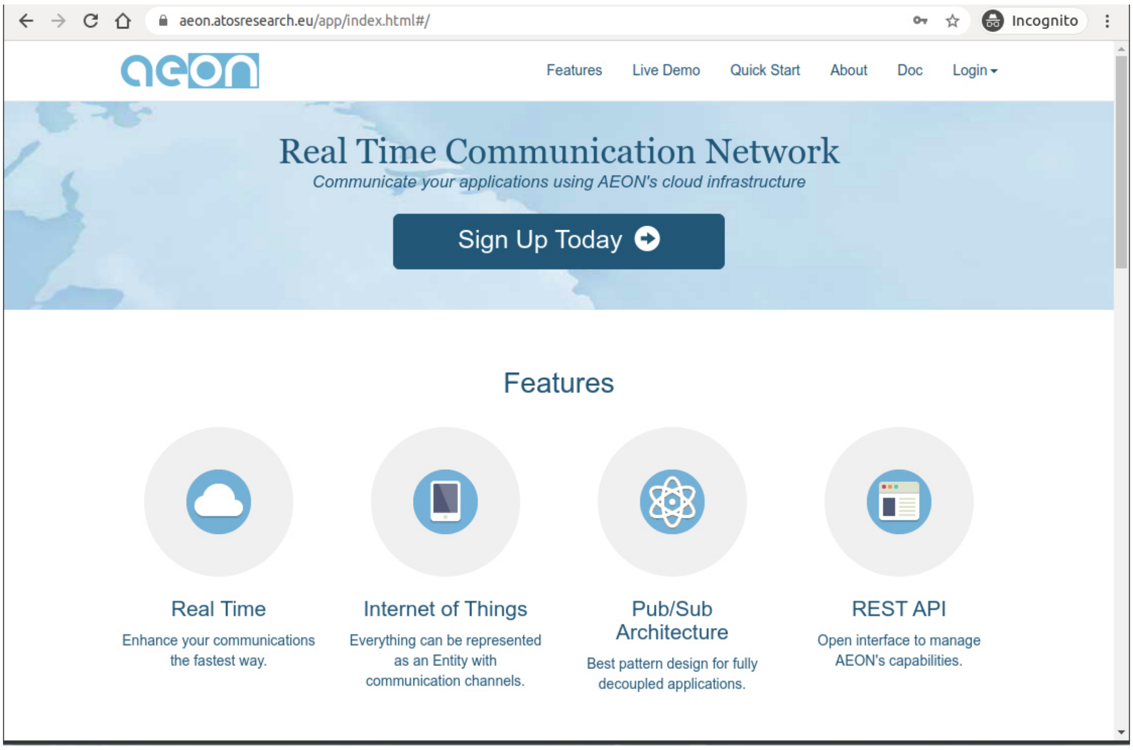


Figure 43. AEON dashboard.

4.2. Configuring AEON channels

AEON users can create AEON channels to publish/subscribe messages through their subscription and publications URLs. The Cross-CPP marketplace uses AEON with its own user to create communication channels for Data Requests (offers), active streaming analytics, Data Views, etc. to get data notifications in real-time.

Data consumers or Service providers will subscribe to the subscriptions URLs to receive real-time data from the Data Requests, analytics, Data Views etc. The configuration is provided in the detailed information view of each one in the Cross-CPP Marketplace frontend and API.

Administrators can access Data Requests, Analytics and Data Views AEON channels information and subscribe to them to check the proper functioning of these services.

4.3. Subscribing to AEON

AEON provides an SDK (Node.js, JavaScript and Java) that encapsulates the complexity of connecting to a socket server.

You can visit https://aeon.atosresearch.eu:3000/public/doc/html/quickstart/nodejs.html for further documentation.

FAQ

CROSS-CPP data-marketplace

Q: What is CROSS-CPP data-marketplace?

A: CROSS-CPP data-marketplace connects Data Providers and Data Consumers for selling and acquiring Connected Vehicle and Home Building data under the Common Industrial Data model (CIDM). It offers a secure and privacy preserving experience when selling or buying sharing big data, by having the full control over your data shared, to whom and for what purposes.

Cross-CPP offers to cross-sectorial Service Providers, the possibility to search for more than 200 sensor signals, display advance visualization representations (such as Histograms, Geo-Histograms, Time Series) and retrieve those datasets in a seamless experience thanks to the open SDK-API created.

Q: How do I, as administrator, register into AGORA data-marketplace?

A: You can only be registered by another administrator.

Q: What do I have to do in order to start working with CROSS-CPP data-marketplace?

A: Once registered you must be familiar with the CIDM, as it is the format in which Signals ad Measurement Channels are based. You must also be familiar with AEON, as it is the communication channel used to send the data.

CROSS-CPP data model

Q: What is the Common Industrial Data Model (CIDM)?

A: The CIDM is a standardized data model for industrial data-driven services. You can find extended information about the CIDM in the Annex.

Q: Which are the benefits and advantages of using the CIDM model for data -driven services:

A: -The CIDM constitute a major business and technical advantage for Data Consumers:

- The CIDM provides a brand-independent and transparent data model, which harmonizes proprietary data into generic datasets independently of any cross-sectorial Industry
- It is built on an open and highly scalable automotive big data format (JSON Schema).
- Active community of service providers increasing the number of signals available from vehicles and Smart Buildings to be recorded as well as the type of measurement channels can be modified or extended
- The Data Provider also provides an origin certification as a CIDM feature to support the validation and verification of origin, integrity and completeness of data. The intention is to protect the data inside the Data Package against manipulation.

Q: What is a signal?

A: A signal is the information provider of each CPP. They are the perception organs of CPPs and it is their main duty to detect physical phenomenon and chemical quantities. They observe the environment and generate data in the CIDM format. An example could be "speed" or "latitude"

Q: What is a channel?

A: A channel is the way the physical signals and their sampled measurements are implemented and represented in the CIDM format. Some examples could be "Vehicle Speed" using the signal "Speed" in a time-series or in a histogram format, or "Position" using both "Latitude" and "Longitude" signals.

CROSS-CPP marketplace components

Q: What is the Data Discovery component and how does it work?

A: The Data Discovery component is a tool that allows you to find what data a Service Provider can access through the marketplace. There you can use the filters provided to narrow or enlarge your results.

Q: What is the Context Monitoring Component and how can help my Organization?

A: The Context Monitoring component allows CROSS-CPP to suggest you signals to add to your current Data Discovery filters, based on the context data of the signals already selected. This might help you find data of interest that you would miss otherwise.

Q: What does it mean a Data Request is context sensitive?

A: The Data Request is context sensitive in case the Service Provider selects some of the CME module data filters to restrict the data packages that it wants to receive. For instance, if the Service Provider just wants to receive data that is provided by vehicles that are driving in the highway and this type of context data is extracted then one case say that the Data Request (and data packages that are produced from this request) is context sensitive.

Q: Which are the consequences of making a Data Request context sensitive?

A: Since a Service Provider can do this by selecting several context information that could be used to filter the Data Request, the consequence of adding such a filter will be that the Service Provider receive a smaller set of data packages (filtered by the selected context information) than would usually receive in the case that no context information is selected.

Data Requests

Q: What is a Data Request?

A: A Data Request is a set of filters that defines which type of data would a Service Provider like to receive. The Service Provider would receive data from CPP owners that have accepted these requests through each request unique AEON channel.

Q: How do a Service Provider create a Data Request?

A: The service Provider must get to the Data Discovery and define the data of interest in through the filters given. Once set and given it a descriptive name, so CPP owners guess in a glance the nature of the request.

Q: How do the Service Provider receive data from a Data Request?

A: In the very moment a Data Request is created an AEON channel is assigned. You can find the channel configuration in each data request details view. Data from CPP owners that have accepted the request will be sent through each AEON channel assigned to each eligible data request, meaning you can review data sent through that Data Request by listening to that AEON channel.

Q: Can a Data Request be modified once created?

A: No. The acceptance of a request by a CPP owner implies a consent from its side. Modifying the request would make invalid such consent.

Q: How can a Data Request published in the Marketplace be deleted

A: A Data Request can be deleted in the details view of that request, or through the provided API endpoint (see Service Providers Developers Guide). This action cannot be undone, and all acceptances of the request will be automatically terminated.

Q: Can the data collected be used for other purposes not described in the Data Request?

A: No. The acceptance of a request by a CPP owner implies a consent from its side. That user allows certain usage of the data given and only for the purposes described in the request by the Service Provider (such as use for analytics or evaluate its context).

Toolbox

Q: What is the toolbox?

A: The Toolbox is a set of tools that offers Service Providers a way to generate analytics from the data obtained from Data Requests or further filter this data in order to get exact measurements or use the service as a notification system.

Q: How are the analytics requested?

A: Any analytics uses data packages received from a Data Request, meaning that first the data Request must have been accepted by CPP Owners and started receiving data. Then, on the Cross-CPP Marketplace a Service Provider can create analytics based on those Data Requests.

Q: How are the analytics results given?

A: Depending on the analytics type results can be instantly shown on the screen, as a diagram, chart, map and so on, or a new AEON channel is provided in order to subscribe to upcoming analytics results. Every analytics type explains the way to get the results on screen.

Q: How the analytics be consulted?

A: Any created analytics can be consulted on the Cross-CPP Marketplace under Requested Analytics section. There the administrator can get the IDs and AEON channel for subscription, see results of one-time analytics, or even delete them.

Q: What are the units of the trajectories module?

A: For distances, the meter, and for times the second. Then the velocity is expressed in m/s.

Q: What is the interpretation of the results of clustering the trajectories?

A: Each trajectory included in the data is assigned to a numerical ID that refers to the cluster that is being assigned. By instance, all trajectories of similar length from east to west are grouped in the cluster number 1, so the results are 1 for each trajectory.

Q: What do the nodes represent in the networks module? And the links?

A: Each node is an object. It could represent a house, building, a car. The links are an abstract representation of some magnitude that is being measured. Links could be based on distances by instance.

Q: What is a Data View?

A: A Data View is a configuration to get data filtered by specific values constraints for one or more Measurement Channels included in a Data Request.

Q: How can a Data View be created?

A: Service Providers create them in the Data Views section under Toolbox in the Cross-CPP Marketplace. There a step by step guide will be offered.

Q: How do I check the data view results?

A: Administrators can consult all generated Data Views through Data Views under Management section. There the configuration can be consulted, as well as the AEON channel to subscribe to. Also, the options of retrieving the latest data or even deleting the Data View are offered there.

AEON

Q: What is AEON?

A: AEON is a cloud platform to create applications with real time communications channels

Q: How can I use AEON to subscribe to my data requests and data analytics?

A: AEON provides an SDK (Node.js, JavaScript and Java) that encapsulates the complexity of connecting to a socket server. Please refer to section 6.4, document examples and online page for extended documentation.

Q: How do I create or configure an AEON channel?

A: You don't have to create or configure any channel. All needed AEON channels, such as for data requests or analytics, are created and assigned by Cross-CPP. You only have to use the channels given. The channels configuration can be found in the details view of each data request or analytics.

Glossary

Administrator: CROSS-CPP marketplace system administrator

AEON: AEON application

AEON application: publication/subscription based communication application

AEON channel: set configuration for communication between two actors through AEON application

CROSS-CPP: System

Analytics Toolbox: set of available analytics functions to be requested by the Data Consumer

Channel: sampler of the data the signals process

CIDM: Common Industrial Data Model

CIDM model: standardized data model for industrial data-driven services

Contract: entity that resumes the acceptance of a data request from a data owner

CPP: cyber-physical product

CPP Data: data created by a CCP and sent to the system by the Data Provider

CPP Owner: Data Owner which CPP is registered in the CROSS-CPP data-marketplace

Data Consumer: actor who receives the data created by owners to use it on the creation or improvement of services

Data Consumer Wallet: group of MP functionalities for Data Consumers

Data Owner: owner of the CPP that sends data to the system

Data Provider: OEM that provides its users data to the CROSS-CPP marketplace

Data Request: set of configurations that define a scope for CPP Data to be received by a Data Consumer

Marketplace: Marketplace Web Application

My Data Wallet: group of MP functionalities for Data Owners

MP: Marketplace

OEM: Original Equipment Manufacturer

Offer: published and available Data Request

Provider: Data Provider

Service Provider: Data Consumer

Signal: information provider of the data the CPP sensors generate

System: the whole lot of applications that conforms CROSS-CPP, including Marketplace Web Application and Marketplace Server.

Figures

Tables

Annex

CIDM v1.2.1 jsonSchema

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "description": "Common Industrial Data Model",
    "type": "object",
    "properties": {
        "Signal": {
            "$ref": "#/definitions/Signal"
        },
        "MeasurementChannel": {
            "$ref": "#/definitions/MeasurementChannel"
        },
        "DataPackage": {
            "$ref": "#/definitions/DataPackage"
        }
    },
    "additionalProperties": false,
    "definitions": {
        "TimeSeriesMeasurementChannel": {
            "title": "TimeSeriesChannel",
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "time-series"
                    ]
                },
                "capture-interval": {
                    "type": "number"
                },
                "on-change": {
                    "type": "boolean"
                },
                "sample-strategy": {
                    "type": "string",
                    "enum": [
                        "min",
                        "max",
```

```
                "average",
                "last-known-value"
            ]
        },
        "signal": {
            "$ref": "#/definitions/Signal"
        },
        "format": {
            "type": "string"
        },
        "dimension": {
            "type": "number"
        }
    },
    "required": [
        "type",
        "capture-interval",
        "on-change",
        "sample-strategy",
        "signal"
    ]
},
"MeasurementChannel": {
    "type": "object",
    "properties": {
        "id": {
            "type": "string"
        },
        "name": {
            "type": "string"
        },
        "type": {
            "type": "string",
            "enum": [
                "time-series",
                "histogram",
                "geo-histogram",
                "general-purpose",
                "event-based",
                "basic-cpp-information"
            ]
        },
        "comment": {
            "type": "string"
        }
    },
    "required": [
        "id",
        "name",
        "type"
    ],
    "oneOf": [
        {
            "$ref": "#/definitions/TimeSeriesMeasurementChannel"
        },
        {
            "$ref": "#/definitions/HistogramMeasurementChannel"
```

```
        },
        {
            "$ref": "#/definitions/GeoBasedHistogramMeasurementChannel"
        },
        {
            "$ref": "#/definitions/GeneralPurposeMeasurementChannel"
        },
        {
            "$ref": "#/definitions/BasicCppInformationMeasurementChannel"
        },
        {
            "$ref": "#/definitions/EventBasedMeasurementChannel"
        }
    ]
},
"HistogramMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "histogram",
                "geo-histogram"
            ]
        },
        "aggregation-strategy": {
            "type": "string",
            "enum": [
                "time",
                "count",
                "min",
                "max"
            ]
        },
        "capture-interval": {
            "type": "number"
        },
        "dimensions": {
            "type": "integer",
            "minimum": 1
        },
        "bins": {
            "type": "array",
            "minItems": 1,
            "items": {
                "type": "object",
                "properties": {
                    "type": {
                        "type": "string",
                        "enum": [
                            "linear",
                            "logarithmic",
                            "custom"
                        ]
                    },
                    "lower-bound": {
                        "type": "number"
```

```
        },
        "upper-bound": {
            "type": "number"
        },
        "signal": {
            "$ref": "#/definitions/Signal"
        },
        "number-of-bins": {
            "type": "integer",
            "minimum": 0
        },
        "alternative-bin-labels": {
            "type": "array",
            "items": {
                "type": "string"
            }
        }
    },
    "required": [
        "type",
        "lower-bound",
        "upper-bound",
        "signal",
        "number-of-bins"
    ],
    "oneOf": [
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "linear",
                        "logarithmic"
                    ]
                }
            },
            "required": [
                "type"
            ]
        },
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "custom"
                    ]
                },
                "custom-bounds": {
                    "type": "array",
                    "items": {
                        "type": "number"
                    }
                }
            },
```

```
                        "required": [
                            "type",
                            "custom-bounds"
                        ]
                    }
                ]
            }
        }
    },
    "required": [
        "type",
        "aggregation-strategy",
        "capture-interval",
        "dimensions",
        "bins"
    ],
    "additionalProperties": false
},
"GeoBasedHistogramMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "geo-histogram"
            ]
        },
        "geo-resolution": {
            "type": "number"
        }
    },
    "required": [
        "type",
        "geo-resolution"
    ],
    "additionalProperties": false,
    "allOf": [
        {
            "$ref": "#/definitions/HistogramMeasurementChannel"
        }
    ]
},
"GeneralPurposeMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "general-purpose"
            ]
        },
        "signal": {
            "$ref": "#/definitions/Signal"
        }
    },
    "required": [
        "type",
```

```
        "signal"
    ],
    "additionalProperties": false
},
"BasicCppInformationMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "basic-cpp-information"
            ]
        },
        "signal": {
            "$ref": "#/definitions/Signal"
        }
    },
    "required": [
        "type",
        "signal"
    ],
    "additionalProperties": false
},
"EventBasedMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "event-based"
            ]
        },
        "format": {
            "type": "string"
        },
        "event-sample-strategy": {
            "type": "string",
            "enum": [
                "real-time-event",
                "trigger-event",
                "threshold-event"
            ]
        },
        "comment": {
            "type": "string"
        }
    },
    "required": [
        "type",
        "signal",
        "event-sample-strategy"
    ],
    "additionalProperties": false
},
"DataPackage": {
    "type": "object",
    "properties": {
```

```
"datapackage-id": {
    "type": "string"
},
"vault-id": {
    "type": "string"
},
"trip-id": {
    "type": "string"
},
"cpp-id": {
    "type": "string"
},
"cvim-version": {
    "type": "string",
    "enum": [
        "1.0.0",
        "1.0.1",
        "1.2.0",
        "1.2.1"
    ]
},
"cpp-type": {
    "type": "string",
    "enum": [
        "vehicle",
        "building"
    ]
},
"type": {
    "type": "string",
    "enum": [
        "time-series",
        "histogram",
        "geo-histogram",
        "general-purpose",
        "event-based",
        "basic-cpp-information"
    ]
},
"measurement-channel-id": {
    "type": "string"
},
"submit-time": {
    "type": "string",
    "format": "date-time"
},
"mileage-start": {
    "type": "number"
},
"room-id": {
    "type": "string"
},
"mileage-stop": {
    "type": "number"
},
"geo-bounding-box": {
    "type": "object",
```

```
        "properties": {
            "latitude-min": {
                "type": "number"
            },
            "latitude-max": {
                "type": "number"
            },
            "longitude-min": {
                "type": "number"
            },
            "longitude-max": {
                "type": "number"
            },
            "altitude-min": {
                "type": "number"
            },
            "altitude-max": {
                "type": "number"
            }
        },
        "additionalProperties": false
    },
    "location": {
        "type": "object",
        "properties": {
            "latitude": {
                "type": "number"
            },
            "longitude": {
                "type": "number"
            }
        },
        "additionalProperties": false
    },
    "oem-certification": {
        "type": "object",
        "properties": {
            "signature": {},
            "checksum": {},
            "sequence-number": {}
        },
        "additionalProperties": false
    },
    "expiration-date": {
        "type": "string",
        "format": "date-time"
    },
    "data-ownership-information": {
        "type": "object",
        "properties": {
            "privacy-veto-rights": {
                "type": "object",
                "properties": {
                    "consent-level": {
                        "type": "string",
                        "enum": [
                            "public",
```

```
                        "shared",
                        "private"
                    ]
                },
                "data-format": {
                    "type": "string",
                    "enum": [
                        "time-series",
                        "histogram"
                    ]
                },
                "jurisdiction": {
                    "type": "string",
                    "enum": [
                        "Europe",
                        "any"
                    ]
                },
                "storage-constraint": {
                    "type": "string",
                    "enum": [
                        "OEM storage",
                        "Personal storage"
                    ]
                }
            },
            "required": [
                "consent-level"
            ]
        },
        "copyright-stakeholders": {
            "type": "array",
            "items": [
                {
                    "type": "object",
                    "properties": {
                        "name": {
                            "type": "string"
                        },
                        "status": {
                            "type": "string"
                        }
                    },
                    "required": [
                        "name",
                        "status"
                    ],
                    "additionalProperties": false
                }
            ]
        },
        "data-stakeholders": {
            "type": "array",
            "items": [
                {
                    "type": "object",
                    "properties": {
```

```
                            "name": {
                                "type": "string"
                            },
                            "status": {
                                "type": "string"
                            }
                        },
                        "required": [
                            "name",
                            "status"
                        ],
                        "additionalProperties": false
                    }
                ]
            },
            "data-privacy-level": {
                "type": "string",
                "enum": [
                    "public",
                    "shared",
                    "private"
                ]
            }
        },
        "additionalProperties": false
    },
    "data-masking-active": {
        "type": "boolean"
    },
    "signatures": {
        "type": "array",
        "items": [
            {
                "type": "object",
                "properties": {
                    "signatory": {
                        "type": "string"
                    },
                    "checksum": {
                        "type": "string"
                    },
                    "signature": {
                        "type": "string"
                    }
                },
                "required": [
                    "signatory",
                    "checksum",
                    "signature"
                ],
                "additionalProperties": false
            }
        ]
    }
},
"required": [
    "cvim-version",
```

```
        "type",
        "measurement-channel-id",
        "vault-id",
        "cpp-type"
    ],
    "oneOf": [
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "time-series"
                    ]
                },
                "number-of-samples": {
                    "type": "integer",
                    "minimum": 1
                },
                "statistic-properties": {
                    "type": "object",
                    "properties": {
                        "min": {
                            "type": "number"
                        },
                        "max": {
                            "type": "number"
                        },
                        "average": {
                            "type": "number"
                        },
                        "histogram": {
                            "type": "object",
                            "properties": {
                                "measurement-channel-id": {
                                    "type": "string"
                                },
                                "data": {
                                    "type": "array",
                                    "minItems": 0,
                                    "maxItems": 10,
                                    "items": {
                                        "type": "number"
                                    }
                                }
                            },
                            "required": [
                                "measurement-channel-id",
                                "data"
                            ]
                        }
                    }
                },
                "data": {
                    "type": "array",
                    "minItems": 1,
                    "items": {
```

```
                        "type": "object",
                        "properties": {
                            "timestamp": {
                                "type": "string",
                                "format": "date-time"
                            },
                            "value": {
                                "type": [
                                    "array",
                                    "string",
                                    "number",
                                    "boolean"
                                ],
                                "items": {
                                    "type": [
                                        "string",
                                        "number",
                                        "boolean"
                                    ],
                                    "minLength": 1
                                }
                            }
                        }
                    }
                },
                "timestamp-start": {
                    "type": "string",
                    "format": "date-time"
                },
                "timestamp-stop": {
                    "type": "string",
                    "format": "date-time"
                }
            },
            "required": [
                "type",
                "number-of-samples",
                "data",
                "timestamp-start",
                "timestamp-stop"
            ]
        },
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "histogram",
                        "geo-histogram"
                    ]
                },
                "data": {
                    "type": "array",
                    "items": {
                        "type": [
                            "number",
```

```
                    "array"
                ],
                "minItems": 1,
                "items": {
                    "type": "number"
                }
            }
        },
        "timestamp-start": {
            "type": "string",
            "format": "date-time"
        },
        "timestamp-stop": {
            "type": "string",
            "format": "date-time"
        }
    },
    "required": [
        "type",
        "data",
        "timestamp-start",
        "timestamp-stop"
    ]
},
{
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "general-purpose",
                "basic-cpp-information"
            ]
        },
        "data": {},
        "timestamp": {
            "type": "string",
            "format": "date-time"
        }
    },
    "required": [
        "type",
        "data",
        "timestamp"
    ]
},
{
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "event-based"
            ]
        },
        "data": {
            "type": "object",
```

```
                            "properties": {
                                "event-type": {
                                    "type": "string",
                                    "enum": [
                                        "real-time-event",
                                        "trigger-event",
                                        "threshold-event"
                                    ]
                                },
                                "event-data": {
                                    "type": "object",
                                    "properties": {
                                        "value": {
                                            "type": "string"
                                        },
                                        "event-datapackages": {
                                            "type": "array",
                                            "minItems": 1,
                                            "items": {
                                                "$ref": "#/definitions/DataPackage"
                                            }
                                        }
                                    },
                                    "required": [
                                        "value"
                                    ]
                                }
                            },
                            "required": [
                                "event-type",
                                "event-data"
                            ]
                        },
                        "timestamp": {
                            "type": "string",
                            "format": "date-time"
                        }
                    },
                    "required": [
                        "type",
                        "data",
                        "timestamp"
                    ]
                }
            ]
        },
        "Signal": {
            "title": "CIDM Signal",
            "description": "Signals are the perception organs of vehicles. It is their ma
in duty to detect physical phenomenons and chemical quantities by transferring them into
electrical signals. They observe the environment and gener-
ate the data that is exchangeable at AutoMat’s marketplace. They are one of the core comp
onents of the AutoMat project. Figure 11 shows the UML modelling of the signals. Within A
utoMat all information pro-
viders are modelled as Signal. They can be classified as static signals or changing/non-
static signals, having a sample rate larger than zero. ",
            "type": "object",
```

```
        "properties": {
            "id": {
                "type": "string"
            },
            "name": {
                "type": "string"
            },
            "cpp-type": {
                "type": "string",
                "enum": [
                    "vehicle",
                    "building"
                ]
            },
            "type": {
                "type": "string",
                "enum": [
                    "numeric",
                    "information",
                    "enumeration",
                    "general-purpose"
                ]
            },
            "format": {
                "type": "string"
            },
            "sample-rate": {
                "type": "number",
                "minimum": 0
            },
            "comment": {
                "type": "string"
            }
        },
        "required": [
            "id",
            "name",
            "type",
            "sample-rate",
            "cpp-type"
        ],
        "oneOf": [
            {
                "$ref": "#/definitions/NumericSignal"
            },
            {
                "$ref": "#/definitions/EnumerationSignal"
            },
            {
                "$ref": "#/definitions/InformationSignal"
            },
            {
                "$ref": "#/definitions/GeneralPurposeSignal"
            }
        ]
    },
    "GeneralPurposeSignal": {
```

```
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "general-purpose"
                ]
            }
        },
        "required": [
            "type"
        ]
    },
    "InformationSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "information"
                ]
            },
            "format": {
                "type": "string"
            }
        },
        "required": [
            "type",
            "format"
        ]
    },
    "EnumerationSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "enumeration"
                ]
            },
            "items": {
                "type": "array",
                "items": {
                    "type": "string"
                }
            }
        },
        "required": [
            "type",
            "items"
        ]
    },
    "NumericSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
```

```
                    "enum": [
                        "numeric"
                    ]
                },
                "format": {
                    "type": "string",
                    "enum": [
                        "int",
                        "uint",
                        "float",
                        "double"
                    ]
                },
                "min": {
                    "type": "number"
                },
                "max": {
                    "type": "number"
                },
                "resolution": {
                    "type": "number"
                },
                "unit": {
                    "type": "string"
                }
            },
            "required": [
                "type",
                "format",
                "min",
                "max",
                "resolution",
                "unit"
            ]
        }
    }
}
```

About Cross-CPP

The objective is to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research is to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project will develop: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to pro-vide enhanced local weather forecast and predict and optimise energy consumptions in households. The project will build upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, will be used as a basis for further development aiming to extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

Funded by the Horizon 2020 Framework Programme of the European Union

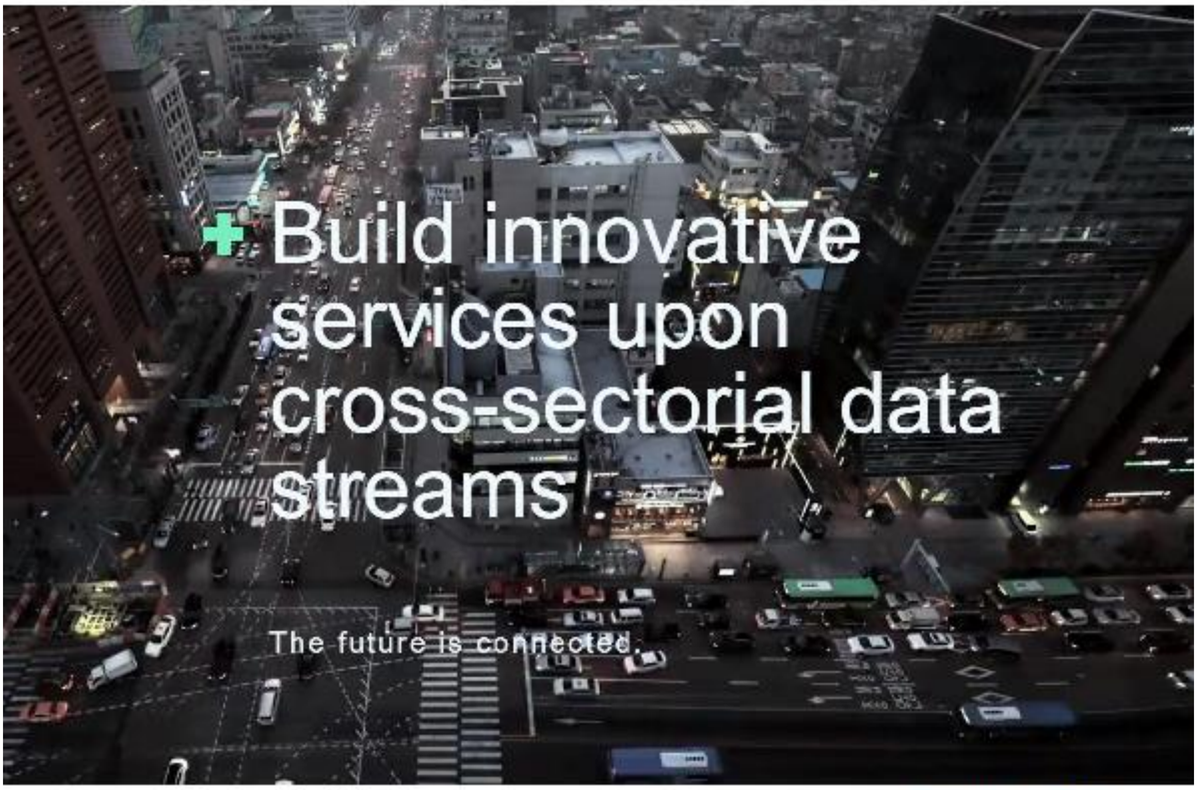
7 Annex 6: Ecosystem White Paper

CROSS-CPP

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

Cross-CPP Ecosystem

White Paper



November 2020

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780167

Foreword

Welcome to our Cross-CPP Ecosystem white paper. From the very first, we were absolutely convinced that Data Markets have to become more attractive for its key stakeholders to overcome existing obstacles, as e.g. the limited access to multiple data streams, or privacy concerns. Thus, we build up an Open Ecosystem to empower Data Owners to exploit and control their most valuable data assets from smart products and to give Data Customers access to this great spectrum of sensor data. All along, we have followed the maxim to think about the needs of Data Owners and Data Customers, but also to win smart product manufacturers (e.g. car makers) to open up their products, by designing a convincing trustworthy Ecosystem.

Recently we have finalized the implementation and integration of the Cross-CPP Ecosystem and have started its validation and assessment by data providers and data customers. Several public presentations of our Data Marketplace solution and Analytic Toolbox have been presented a short time ago in the BDVe Webinar ‘How to monetize your data in an open data Marketplace’, which is still accessible via the BDVe website.

In this White Paper you will find some more details about the Cross-CPP ecosystem concept as a whole, leading from today’s constraints in cross-industrial data access to how Cross-CPP will face those challenges with its innovative ecosystem concept.

If you got curious about how all that is made possible, just continue on the following pages, enjoy the reading, and please contact us with your feedback or questions!

Cross-CPP consortium partners

Executive Summary

With the increasing number of connected sensors and actuators within cyber physical mass products (CPP) there is an enormous amount of data continuously generated, representing on the one side a new information resource to create new value, allowing the improvement of existing services or the establishment of diverse new cross-sectorial services, on the other side a major big data-driven business potential - not only for the manufacturers of CPP, but in particular also for cross-sectorial industries and various organisations with interdisciplinary application services.

We currently see, however, that these business potentials are currently still locked since manufacturing industry producing CPP are driven by CPP specific business approaches. This situation is mainly characterised by today's sporadic proprietary CPP data access restricted to CPP manufacturer specific products and services and limited access to CPP data caused by missing or distributed access to CPP data as well as by diverse brand specific data formats.

In this White Paper, we present to you the key innovations of the Cross-CPP ecosystem concept to face those challenges. These key innovative features concern the *Standard Data Model* fundamental for the brand independent data exchange, the *one-stop shop* concept represented by the Cross-CPP Marketplace acting as the central mediator for the data exchange between data provider and data consumer and the basic data *privacy and security* concept granting full control to the data owner over his data.

The following key points will refer to those main innovative features and show up the high future potential of the Cross-CPP ecosystem concept:

Once brought to life**,** *all* involved stakeholders - service providers as well as data providers and data owners - are in a win-win situation and will profit from the whole Cross-CPP ecosystem.

1 Challenges on the way to build innovative services upon cross-sectorial data streams

Key ***motivation of Cross-CPP*** project is to give cross-sectorial industries access to the great spectrum of sensor data coming from high volume products from various industrial sectors (vehicles, smart home devices, etc.). With the increasing number of connected sensors and actuators within such mass products, this number will rise in short-term. This enormous amount of data continuously generated by mass products will represent:

- ***a NEW information resource to create new value***, allowing the improvement of existing services or the establishment of diverse new cross-sectorial services, by combining data streams from various sources
- ***a major big data-driven business potential***, not only for the manufacturers of Cyber Physical Products (CPP), but in particular also for cross-sectorial industries and various organisations with interdisciplinary applications.

Existing CPP Data Access Constraints

However, these business potentials are currently still locked since manufacturing industry producing mass products has not established clear models and tools for such cross sectorial collaborations. Current solutions and offerings in the CPP data domain are driven by OEM specific business approaches that are almost solely focused on their own products and are realized by proprietary solutions. The resulting brand-specific solutions mostly do not provide CPP data to the outside world, hindering long-term value creation by service providers due to fragmented environments and the lack of brand-independent representation of CPP data. Due to this high level of market fragmentation, the situation today is characterized by far too complex and individual value chains resulting in economic inefficiency.

This situation is mainly characterised by the following major difficulties for both, the service providers (data customers) and for the CPP manufactures / owners (data providers):

- **No or limited access to CPP data** caused by missing or distributed access to CPP data as well as by diverse brand specific data formats. This situation is forcing service providers interested in CPP data to build up and maintain interfaces to diverse ecosystems with different data models, causing high efforts and costs for data collection and processing.
- **Limited possibilities to use cross-sectorial CPP big data streams,** due to missing functionalities for an easy access and detection of needed data, as well as of easy to use big data analytics functionalities for Service Providers with low big data expertise and knowledge.
- **Missing preconditions to establish such cross-sectorial data market**, as for e.g. the absence of an agreed common data model for CPP data coming from various industrial sectors, or mechanisms for an optimal management of commercial confidentiality, privacy, IPR and ethical aspects applicable to various cross CPP data streams.
- **Uneconomical brand-specific service platform solutions**, causing high costs for the implementation of proprietary data platforms, which have to be financed and justified by just a few CPP manufacturer specific services (no single service can bear the cost for such system solution -> ROI is not achievable). Thus, today services will be often offered for high price not acceptable for the majority of customers.
- **Wasted Innovation Potentials** by thousands of external experts due to closed brand-specific data platforms. Even very big CPP Manufactures will by far not have the expertise and innovation potentials of the world-wide network of service providers. By that reams of new innovative services remain closed.

The Cross-CPP Ecosystem Approach to Overcome the CPP Data Access Constraints

On the contrary to today's sporadic proprietary CPP ecosystems, which are in most cases restricted to CPP manufacturer specific services and which are not open for third parties interested in these CPP data, the Cross-CPP Ecosystem targets to bring sensor data from CPPs of various industrial sectors to the outside world. Therefore, as key challenges, Cross-CPP has to overcome several obstacles by establishing a CPP Big Data Ecosystem, which has to face the following key challenges:

- **Brand independent concept**, open for integration of diverse CPP data providers coming from different industrial areas, also providing a brand independent standardized cross industrial CPP data model which needs to be flexible enough to incorporate data coming from various industrial sectors.
- A Mediator Platform (called the **CPP Big Data Marketplace)** providing to Service Providers a single CPP data access point with just one interface (one-stop-shop), as well as support functionalities for easy data mining/analytics. By these means, data customers (Service Providers) just need to set-up and maintain one interface to gather diverse CPP data from different CPP providers.
- **Controlled access to diverse CPP data streams** and optimal management of data ownership and data rights, applicable to various cross CPP data streams.
- **Win-Win value chain** for all ecosystem partners, due to the fact that the costs for the ecosystem in place can be shared by a large amount of data customers, which will make a single service much more economical.

2 The Cross-CPP Ecosystem Solution to Meet the Challenges

How a system solution has to looks like, which can satisfy the stated challenges and which provides the stated features to the stakeholder of the Cross-CPP Ecosystem?

Starting Point of the Cross-CPP System Development

The Cross-CPP Ecosystem development had not to reinvent the wheel and could build upon the results of the past and current projects ProSEco, AutoMat, IASIS, Resilience 2050, and Juniper.

Specifically, results from the project AutoMat, which has established a novel and open Ecosystem in the form of a cross-border Vehicle Big Data Marketplace that leverages currently unused information gathered from connected vehicles, was used as a basis for further development. The validity of the developed brand independent AutoMat concept and standardized common vehicle data model has been proven in the scope of project for the automotive industry by three leading car manufactures (Volkswagen, Renault and FIAT).

> *In contrast to AutoMat project, which was focused exclusively upon automotive industry, a key mission of Cross-CPP is to analyse and extend the AutoMat concept with regard to the integration of data streams coming from other CPPs than vehicles.*

To achieve this transition from AutoMat to a "Cross-Sectorial Ecosystem" the following challenges have to be faced:

- Enlarge the big data pool by providing data streams from various industrial sectors such as automotive, home automation systems and other industries.
- Extend Cross-CPP Data Model, for data streams coming from multiple industrial sources, applicable for various industrial sectors.
- Empower data owners to exploit their most valuable CPP data (vehicle data, smart home device data etc.), simultaneously ensuring that the owner can fully control which data should be harvested and provided to which Service Provider, always under consideration of their commercial confidentiality, privacy, IPR and ethical aspects.
- To offer advanced one-stop-shopping support for Service Providers, especially for SMEs or start-ups, by providing instruments enabling an easy access, detection and selection of required data, as well as flexible cross-stream analysis tools for large data volumes.

The Three Pillars of the Cross-CPP Ecosystem Structure

The Cross-CPP Ecosystem solution is characterised by an open system approach, enabling that any Data Owner can bring his sensor data to the outside world, to be used by any Service Provider. Thereby, the Cross-CPP System Solution can be grouped into three main pillars, as presented in Figure 1:

- The **Left Pillar** represents Data Providers (CPP Manufacturers/Data Owner) side, comprising the data harvesting and making CPP data available to the Data Customers in a standardised format.
- The **Right Pillar** represents Data Customer/Service Provider side from Cross-sectorial industries or manufacturers of CPP using the provided CPP data from various products to create new value out of that data by improving services or establishment of diverse new cross-sectorial services.
- The **Middle Pillar** represents the mediator between the Data Provider and the Data Customer, responsible for the data handling between both parties. This comprises the storage

of the data in a Data Owner's storage vault. Furthermore, the so called Big Data Marketplace handles the controlled access to the stored CPP data from the various Data Owners and forwards the data to the respective Service Providers.

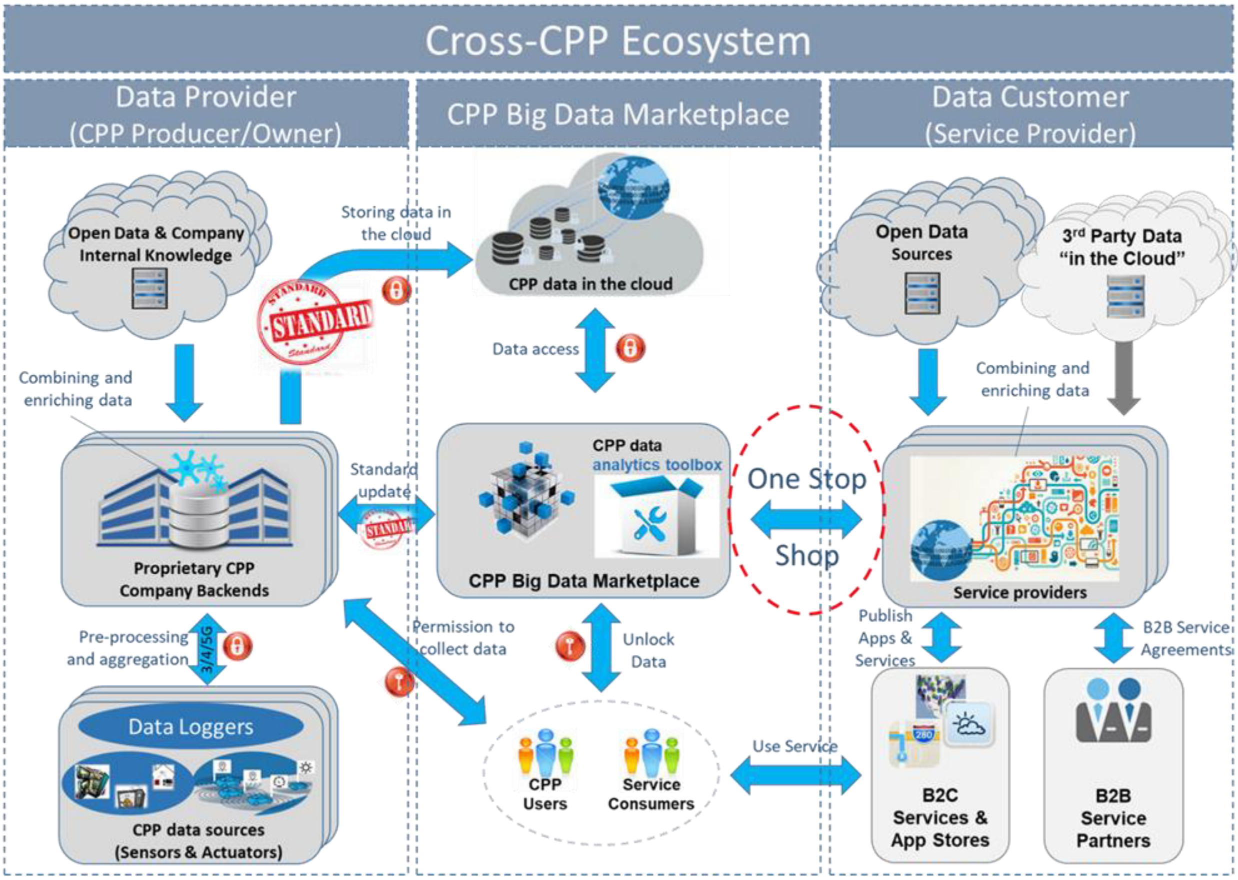


Figure 1: The three Pillars of the Cross-CPP System Structure

For a better understanding of the Cross-CPP Ecosystem Structure in reference to Figure 1 the three pillars are explained in more detailed:

Left Pillar - Data Harvesting and CPP Company Backend

Data Harvesting:

The main role of the Proprietary CPP Data Harvesting is to acquire the data from a proprietary CPP data source and send them to CPP Company Backend. To ensure that the data harvesting process is in consent with end user needs, a data acquisition configuration is downloaded from the CPP Company Backend and deployed in the CPP data logger. CPP data will be measured and stored as CPP data packages. In order to reduce the amount of information to be transmitted the CPP signals can be pre-processed and aggregated. These CPP data will be transmitted to the CPP Company Backend module according to the transmission strategy defined by the CPP manufacturer.

Company Backend:

It represents the central data access point to the CPPs for thousands to millions of CPPs of a brand. It will be based on a proprietary brand specific solution, interpreting and transforming proprietary manufacture-specific CPP data into physical information in reference to agreed owner permissions. Furthermore, the information will be validated and can be masked to enforce privacy. Finally, the information is converted into the required quasi standard data format, the Unified

Cross-Industrial Data Model (CIDM) and is published to the owner's CPP Cloud Storage. The CPP Company Backend has to handle also the consent of the data owner for data harvesting.

Middle Pillar - Cloud Storage and Big Data Marketplace

Cloud Storage:

Cross-CPP targets a cloud based concept for the storage of CPP Big Data. Any Data Owner holds a data vaults explicitly for his data. The storage infrastructure enables the data reception from the CPP Manufacture Backend, as well as data access by the CPP Big Data Marketplace in the standard CIDM format. Furthermore, storage management has to prevent data from unauthorized manipulation, ensure data completeness and the protection of the privacy of data owners, etc. The storage management is also able to handle a CPP Data Model updates.

Big Data Marketplace:

The Big Data Marketplace embodies the central element of the Cross-CPP Ecosystem approach representing the mediator for all the actors involved in the data flow of the Cross-CPP Ecosystem.

For the data customer (service provider) the CPP Big Data Marketplace represents a "One-Stop-Shop", the single point of access to data streams from multiple mass products. Therefore, the marketplace offers instruments enabling an easy access and detection of needed data, as well as a data analytics toolbox and context features, which will provide easy to use big data analytic functionalities for Service Providers with low big data expertise and knowledge. Furthermore, in reference to the agreements between the Data Owners and the relevant stakeholders during run-time the Marketplace will control the data exchange between Data Owner and Service Provider.

Right Pillar - Data customers & Service providers

Right pillar covers the Data Customer/Service Provider representing cross-sectorial industries or manufacturers of CPP using CPP data to improve services or to establish diverse new cross-sectorial services. The Marketplace supports the Service Provider by enabling access to a data catalogue and tools for the selection of relevant data, by setting up data order, as well as by the access right management with the data Owner.

Once all contractual agreements for the data access by service providers have been arranged, for the run-time phase the Service Providers have to enable to bring the cross-sectorial data streams in the CIDM format to their worlds. Based on specialized algorithms for the various services, also addressing features such as data cleaning, filtering, etc. and transforming the CPP information into service relevant input information, the final service products are generated.

3 The Key Innovations of the Cross-CPP Ecosystem

In the following the basic essentials of the Cross-CPP solution concept are presented mandatory to meet the challenges stated above. These key innovative features concern the Standard Data Model fundamental for the brand independent data exchange, the one-stop shop concept represented by the Cross-CPP Marketplace acting as the central mediator for the data exchange between Data Provider and Data Consumer and the basic data privacy and security concept granting full control to the Data Owner over his data.

3.1 One common standard for all kind of data

The key innovation of the Cross-CPP solution to realise a brand independent data exchange between the Data Providers and Data Consumers embodies the Common Industrial Data Model (CIDM). Except the CPP Manufacturer specific data formats along the data harvesting, the data are transformed into the standard CIDM format before they are transferred to the storage area. In the CIDM format the data will be retrieved from the Cloud Storage.by the Marketplace and in this format forwarded to the Service Providers.

The CIDM represents an open and highly scalable big data format, designed to harmonize proprietary data harvested by the CPP Manufacturer into generic datasets. The CIDM is flexible enough to incorporate data coming from multiple industrial sources, applicable for various industrial sectors.

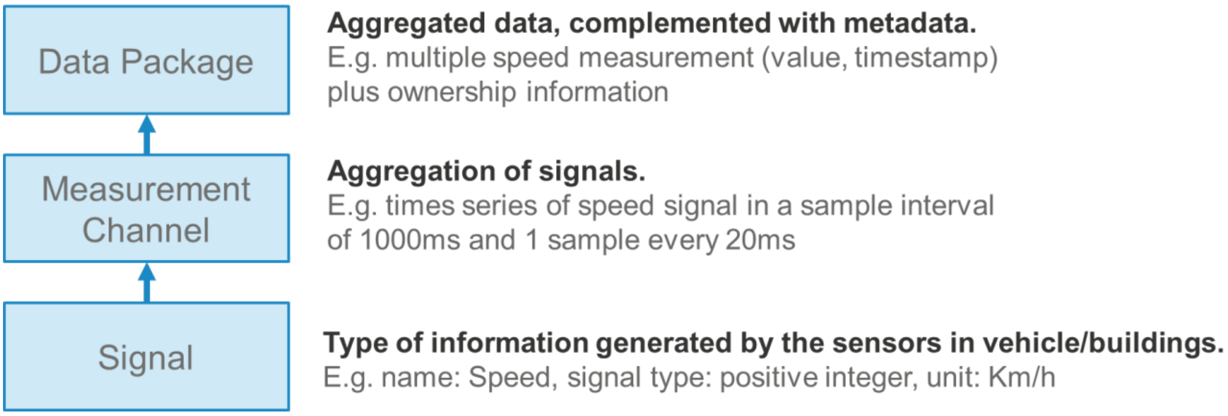


Figure 2: Basic Level Structure of the CIDM

The structure of the CIDM consists of three layers (see Figure 2):

- Starting from the bottom part, Signals describe the type of physical phenomena and chemical quantities of vehicles and buildings etc. including the name of the signal, the format and unit.
- As measurements of the phenomena may far exceed the available transmission bandwidth or the full resolution may not be required in most applications, the data from the CPP need to be pre-processed and aggregated. The result constitutes a “measurement channel” that include the signals to aggregate, the aggregation type (time series, histograms, etc.) and the configuration of the aggregation.
- Finally, at the highest level, data packages provide the actual data coming from the CPP, aggregated according to a measurement channel selected. In addition to the data, data packages also contain metadata with support information like ownership and quality assessment. The data packages are forwarded to the cloud store vault of the Data Owner.

However, the CPP Data Model is not rigid, rather representing a living data structure. In reference to the needs of the Service Provider community the number of signals to be recorded, as well as the type of measurement channels can be modified or extended. Based on the decision of a Standardization Board a required update of the actual version of the CIDM by additional signals/channels might be concluded.

3.2 Empower data owners to exploit and control their most valuable data assets

A key innovation of vital importance for the Cross-CPP Ecosystem solution is to follow the General Data Protection Regulation (GDPR) approved by the European Parliament, the Council of the European Union and the European Commission. The GDPR protects people's personal data throughout the European Union (EU). The decree also affects data exports from the EU. It enforces the right for people to lawfully agree with companies to use their private information. It also enforces the right for people to have their private information no longer accessible by a company.

- Data owners may take advantage of **data masking** features that reflect their privacy requirements
- The Cross-CPP system provides also technical measures capable of **enforcing data controls** corresponding to the nuances specified by the contracts

In reference to the GDPR, Cross-CPP is aiming to give citizens full control over their data so that the Data Owner always is aware about which of his data are accessed, where they are stored and who is using which of his data for what purpose. Moreover, the Data Owner is able to deactivate specific data accesses at any time. This control of the Owner’s personal data causes the necessity to formulate contractual regulations for the interaction between Data Owner and the various stakeholders of the Cross-CPP Ecosystem guaranteeing that the citizens has full control over their personal data. This is implemented by the following introduced key contractual regulations between Data Owner and respective stakeholders:

- CPP Data Owner and CPP Manufacturer: The CPP Manufacturer gets the CPP owner’s permission to gather, pre-process and forward data to his storage vault in the cloud.
- CPP Data Owner and Storage Provider: The CPP Owner rents a cloud storage vault explicitly to store his CPP data.
- CPP Data Owner and Service Provider: The Data Owner permits each Service Provider to use his data required to generate the services.

These contractual agreements represent the reference for the Marketplace to control which data of the respective Data Owner are allowed to be retrieved from his storage vault and which of these data can be forwarded to a specific Service Provider. These permissions can be withdrawn by the Data Owner at any time.

The implemented Security and Policy Enforcement system of the Cross-CPP solution provides the functionality to define CPP privacy rules and contract privacy rules. The system enforces the privacy rules when the service provider consumes the shared data. In order to integrate the functionality provided, the elements created for the integration are the API Client, the sails service and the sails models and controllers.

A Context Sensitive Security Framework of the Cross-CPP Ecosystem ensures that specific security policies are enforced so that access by client applications to data, objects, web services or other protected resources are mediated by policy enforcement points that determine whether access should be granted according to access control policies maintained in a policy server and subject to interpretation according to monitored context. Policy enforcement points may be distributed around a Cross-CPP deployment and the framework is scalable, capable of supporting multiple security policies simultaneously, and adaptable to changing technology, organisational restructuring, and increasing data volumes. Furthermore, a secure data communication between the various stakeholders of the Cross-CPP Ecosystem is guaranteed by the implementation of respective encryption/decryption commercial tools.

3.3 One-Stop-shop for cross-sectorial data

In the Cross-CPP Ecosystem the **CPP Big Data Marketplace** represents the "One-Stop-Shop", the single point of entry for a brand-independent access for Service Providers to create new B2B and B2C data-based products and services. The Marketplace is the meditator for all the actors involved in the flow of the data in the Cross-CPP Ecosystem. Thereby, the Marketplace knows which stakeholders are participating in the Cross-CPP Ecosystem, which data from which data owner are harvested by which CPP Manufacturer and used by which Data Consumer.

In this respect, the Marketplace, in turn, will maintain the mapping between CPP Owners (e.g. CPP_ID, which signals/channels), CPP Manufacturer (e.g. ID, data forwarded storage vaults), Storage Providers (e.g. ID, CPP Owner ID, data available) and Data Consumer: (e.g. registration, data request, data Owner access right). This central stakeholder interaction management by the Marketplace represents the basic concept of the Cross-CPP Ecosystem in order to carry out the access control of the CPP Owner's data by the various data consumers at run-time, as well as being able to answer accurately future requests by any Service Provider.

CPP Big Data Marketplace offers to cross-sectorial Data Consumers, the possibility to search for more than 200 sensor signals, display advance visualization representations (Histograms, Geo-Histograms, Time Series etc.) and retrieve those datasets in a seamless experience thanks to the open SDK-API created.

CPP Big Data-Marketplace functionalities include:

- Provision of a wide data catalogue and delivery of statistics for Service Providers.
- Discovery service of requested data and identification of data owners.
- Management of permissions (future smart contract provision service)
- Secure and reliable end-to-end communication (from the cloud to the Service Providers).
- Transactions accounting.

In the scope of the Data Discovery the Marketplace offers Service Providers a tool to select which data they want to receive from CPP owners. A wide variety of filters are provided in order to help Service Providers to narrow their desired results. Furthermore, a context service is provided. When the Service Provider has selected one or more channels he can also request suggestions of in this context commonly used channels. This is possible thanks to the implementation of the Context Monitoring and Extraction module (CME) within the Data Discovery process. In case suggestions are requested, the CME will provide a list of channels related to those already selected[1]. Channels of interest can be added directly from the suggestion list.

In respect to a stated service provider request to access specific CPP data sets the CPP Big Data Marketplace identifies the relevant Data Owners and supports to manage their permission for the data access by the Service Provider.

During runtime in respect to the data request of a Service Provider the access authorisation will be checked and the relevant data storages identified. The retrieved CPP data from the storage cloud will be decrypted, verified, anonymised where needed and delivered to the Service Providers. In the scope of these activities also an accounting of the data access will be realised as reference for any billing process.

3.4 Find the right data for You!

It may be complicated to get access to huge amount of data. Yet, is this the only challenge? Surely not. As it is well-known in the machine learning community, having data is not tantamount to having knowledge. The Analytics Toolbox simplifies the extraction of the latter, by providing a set of libraries and modules designed to satisfy most data-related needs, and based on the most recent concepts and algorithms developed by the scientific community. It is buttressed by a modular structure, in which new analytics services can be added to fulfil new requirements; and in which multiple algorithms can be chained together, to give answer to even more complex questions.

But how will this Analytics Toolbox help you?

- By enabling fast prototyping. No data to download, no library to develop and deploy in-house. The Analytics Toolbox enables performing a first feasibility evaluation of a new business idea at essentially no cost.
- By unleashing the power of advanced algorithms. The Toolbox includes modules not easily available in other all-purpose analytics solutions, and specifically designed with CPP data in mind. These span from the analysis of thousands of trajectories, to the representation of network relationships. Again, no in-house development is required: the Toolbox includes everything is needed for a first evaluation.
- By minimising overheads. Filter your data prior to download, for instance through averaging, clustering, or through event-driven triggers. Only download what you need, and when you need it.

The data analytics toolbox, will provide easy to use big data analytic functionalities for Service Providers, supporting an easy access, search/detection and selection of required data. Furthermore, the Analytics Toolbox simplifies the extraction of data, by providing a set of libraries and modules designed to satisfy most data-related needs, and based on the most recent concepts and algorithms developed by the scientific community. It is buttressed by a modular structure, in which new analytics services can be added to fulfil new requirements; and in which multiple algorithms can be chained together, to give answer to even more complex questions. The Analytics toolbox provides services such as:

- Basic statistics and data aggregation: from statistical metrics of location and dispersion, to analysis of distributions and entropies.
- Time series analyses, including the detection of drifts, or sudden changes, through the application of statistical and data mining models.

[1] Suggestions are provided based on a context model that is the result of analysis of the physical relation between signals (e.g. in case temperature is a selected signal, the CME would suggest the humidity and sun intensity

- Trajectory analyses. From the processing of individual trajectories, including interpolations and error detections; to multivariate scenarios, as in the detection of clusters of similar trajectories - see the image for an example.
- Network analyses, for understanding the structures created by interacting elements.

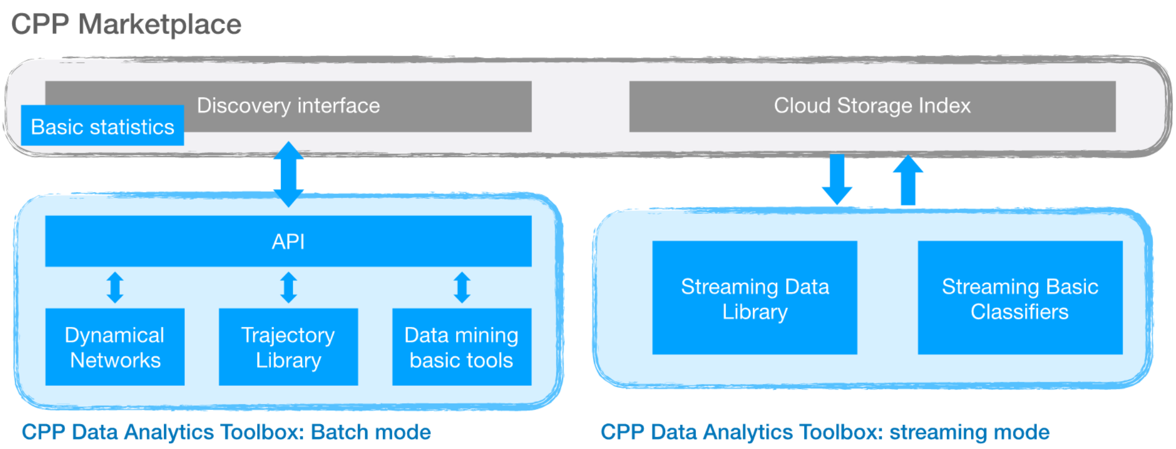


Figure 3: Structure of the Data Analytics Toolbox

During runtime in respect to the data request of a Service Provider the access authorisation will be checked and the relevant data storages identified. The retrieved CPP data from the storage cloud will be decrypted, verified, anonymised where needed and delivered to the Service Provider. In the scope of these activities also an accounting of the data access will be realised as reference for any billing process.

About Cross-CPP

The objective was to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research was to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project developed: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to provide enhanced local weather forecast and predict and optimise energy consumptions in households. The project has built upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, were used as a basis for Cross-CPP development extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

CROSS-CPP

Funded by the Horizon 2020 Framework Programme of the European Union

8 Annex 7: CIDM White Paper

CROSS-CPP

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

CIDM

Common Industrial Data Model

CIDM
Common Industrial Data Model

White Paper

November 2020

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780167

Foreword

Welcome to our Cross-CPP CIDM white paper. From the very first, we were absolutely convinced that Data Markets have to become more attractive for its key stakeholders to overcome existing obstacles, as e.g. the limited access to multiple data streams, or privacy concerns. Thus, we build up an Open Ecosystem to empower Data Owners to exploit and control their most valuable data assets from smart products and to give Data Customers access to this great spectrum of sensor data. All along, we have followed the maxim to think about the needs of Data Owners and Data Customers, but also to win smart product manufacturer (e.g. car makers) to open up their products, by designing a convincing trustworthy Ecosystem.

In this white paper, we will present you the Common Industrial Data Model (CIDM), which - being the central standardization concept - is one of the most crucial parts of our Cross-CPP ecosystem. It features the essential link between Manufacturers of cyber-physical mass products (CPP) and Service Providers, making a huge step in simplifying the processes of data exchange and data provision, at the same time allowing data owners full control over their data at any time.

If you got curious about how all that is made possible, just continue on the following pages, enjoy the reading, and please contact us with your feedback or questions!

Cross-CPP consortium partners

Executive Summary

With the increasing number of connected sensors and actuators within cyber physical mass products (CPP) there is an enormous amount of data continuously generated, representing on the one side a new information resource to create new value, allowing the improvement of existing services or the establishment of diverse new cross-sectorial services, on the other side a major big data-driven business potential - not only for the manufacturers of CPP, but in particular also for cross-sectorial industries and various organisations with interdisciplinary application services.

We currently see, however, that these business potentials are currently still locked since manufacturing industry producing CPP are driven by CPP specific business approaches. This situation is mainly characterised by today's sporadic proprietary CPP data access restricted to CPP manufacturer specific products and services and limited access to CPP data caused by missing or distributed access to CPP data as well as by diverse brand specific data formats.

The CIDM Standard presents the crucial link between CPP manufacturers and Service Providers, featuring a flexible and transparent cross industrial CPP data model, enabling a brand independent data exchange between the Data Providers and Data Consumers for data coming from multiple industrial sources, applicable for various industrial sectors. It is an open and highly scalable big data format, flexible enough to harmonize proprietary data harvested by the CPP Manufacturer into datasets tailored to the needs of the data consumers coming from different industrial sectors. It is satisfying the large range of data and data representations needed by the data customer, at the same time enabling the data owners to have full control about their data.

Thus, the CIDM standard is not an immutable model, but rather represents a *living standard*. Whenever it is required, with respect to the market needs of the service provider community the CIDM standard can be adapted to match these new requirements.

In this White Paper, we present to you the following key points that show up the high potential of the CIDM for all Cross-CPP stakeholders:

Having the CIDM in place, both sides - service providers as well as data providers - are in a win-win situation and profit from the standardisation.

1 Challenges on the Way to a Brand Independent Data Exchange

The Key ***motivation to establish the Common Industrial Data Model (CIDM)*** *standard* is to give cross-sectorial industries access to the great spectrum of sensor data coming from high volume products from various industrial sectors (vehicles, buildings, etc.). With the increasing number of connected sensors and actuators within such mass products there is an enormous amount of data continuously generated by mass products representing

- ***a NEW information resource to create new value***, allowing the improvement of existing services or the establishment of diverse new cross-sectorial services, by combining data streams from various sources and
- ***a major big data-driven business potential***, not only for the manufacturers of Cyber Physical Products (CPP), but in particular also for cross-sectorial industries and various organisations with interdisciplinary application services.

A Missing Interface Standard Constraints the CPP Data Access

However, these business potentials are currently still locked since manufacturing industry producing Cyber Physical Products (CPP) are driven by CPP specific business approaches. This situation is mainly characterised by the following major difficulties for both, the CPP manufactures/owners as data providers and the Service Providers as data consumers:

- Today's sporadic proprietary CPP data access is in most cases restricted to CPP manufacture specific products and services and not open for third parties interested in these CPP data.
- No or limited access to CPP data caused by missing or distributed access to CPP data as well as by diverse brand specific data formats. This situation is forcing Service Providers interested in CPP data to build up and maintain interfaces to diverse CPP manufacturers with different data models, causing high efforts and costs for data collection and processing.

These constraints of brand-specific solutions, not providing brand-independent CPP data to the outside world, are hindering on the long-term the value creation by service providers. Furthermore, such situation is characterized by far too complex and individual value chains resulting in economic inefficiency.

The Cross-CPP CIDM Approach to Overcome the CPP Data Access Constraints

The Common Industrial Data Model (CIDM) represents a unique standard data model, removing brand proprietaries and dependencies and creating a common data format. Figure 1 demonstrates the CIDM as essential link between CPP Manufacturers and Service Providers. On the left side CPP Manufacturers transfer their proprietary data into the CIDM data format. By this, brand-dependent information is removed and aligned to fit the CIDM format. Service Providers, on the right side, can access these data in a standard, brand independent format and can receive generic data sets of various brands. Both sides are in a win-win situation and profit from the standardisation.

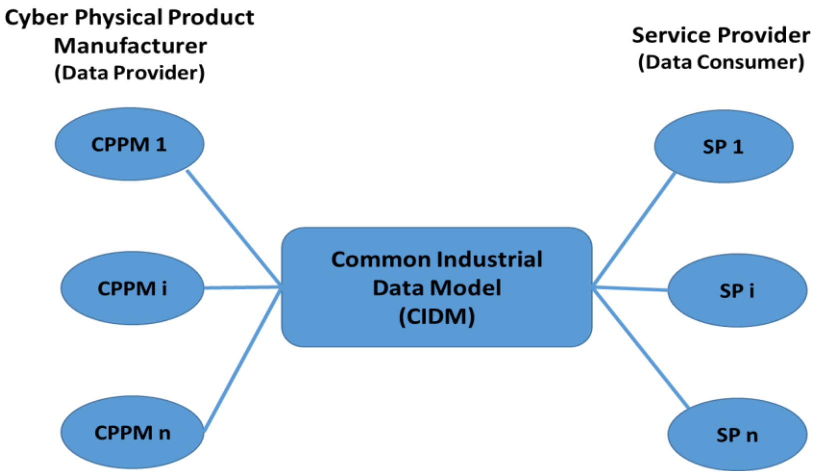


Figure 1: CIDM Standard link between CPP Manufacturers and Service Providers

The CIDM Standard, presenting the crucial link between CPP manufacturer and Service Provider, has to overcome several obstacles and facing the following basic requirements:

- To be a flexible and transparent cross industrial CPP data model, enabling a brand independent data exchange between the Data Providers and Data Consumers for data coming from multiple industrial sources, applicable for various industrial sectors.
- To be an open and highly scalable big data format flexible enough to harmonize proprietary data harvested by the CPP Manufacturer into datasets tailored to the needs of the data consumers coming from different industrial sectors.
- To satisfy the large range of data and data representations needed by the data customer, therefore the model has to support of different signal and measurement types
- To provide Data Ownership Information to enable the data owner to have full control about his data.

Not to invent the wheel twice, for the CIDM approach various exiting data models were taken as reference. There are several existing data formats fulfilling some of the mentioned requirements, among others the W3C – “Vehicle Data” format, which is only meeting some of the requirements, the ERTICO – “SensorIS” data format, which meets most of the requirements but only supports as data types time series signals. Proprietary and closed source models exist but are out of scope as openness is mandatory. None of the approaches fulfil the key CDIM requirements. Any extension of these standards would be very complex and therefore not recommended.

As most suitable starting point for a data model for standardizing data formats from different industrial sectors the Common Vehicle Information Model (CVIM)[1] was identified. This Model fulfils all stated basic requirements providing a brand-independent and transparent data model. However, the CVIM was originally developed for the brand-independent access of vehicle data by Service Providers.

Therefore, the key extension for the transition from CVIM to CIDM data model is related to the required extensions to support besides vehicle data also CPP data sources from other cyber physical products of various industrial sectors (e.g. CPP type ‘Smart Building’). The extended CIDM Model pushes the Win-Win value chain for all ecosystem partners, due to the fact that the costs for data exchange can be shared by a large amount of data customers, which will make a single service much more economical.

[1] CVIM was developed in the scope of the AutoMat project “Automotive Big Data Marketplace for Innovative Cross-sectorial Vehicle Data Services”, EU funded AutoMat – 644657

2 The CIDM Standard Structure to Meet the Challenges

A data model architecture was selected convenient to satisfy the stated challenges and which fulfil the stated features for the provision of the data. Therebye, the CIDM architecture consists of 3 layers as presented in Figure 2. The CIDM format is unbiased and treats all data in the same way. It defines Signals as information providers and Measurement Channels as the recording of the measurement configuration of those signals. Hereby, CIDM does not differentiate between the data origin within the CPP, the CPP type or its manufacturer.

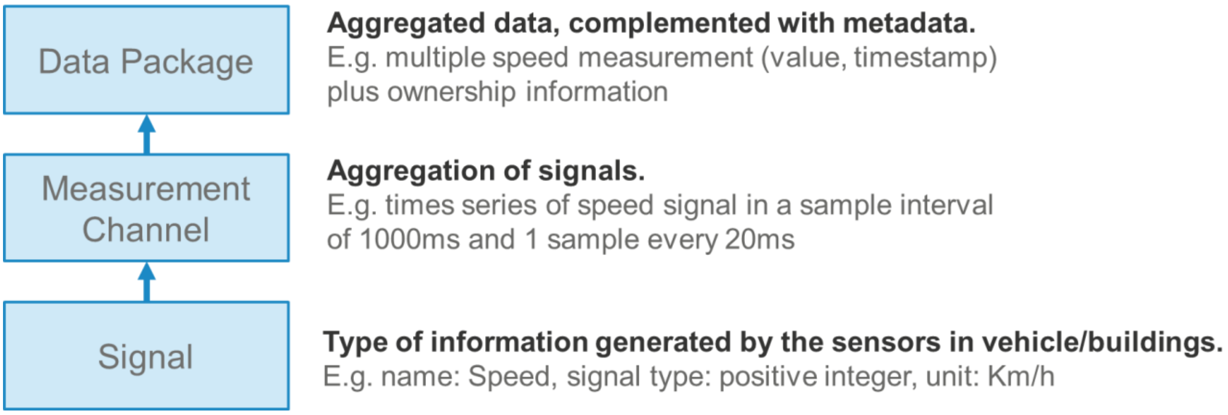


Figure 2: CPP Data Model Main Structure

Signal Layer Specification

Sensors are the perception organs of CPP devices like vehicles and buildings. It is their main duty to detect physical phenomenon and chemical quantities by transferring them into electrical signals. The signal layers describe different types of signals and formats represented in the system. A new property is needed to group signals regarding the signal source type, CPP-type etc.

Measurement Layer Specification

The measurement layer defines how sensor signals are captured and processed. As data contained in signals may far exceed the available transmission bandwidth and as the full resolution may not be required in most applications, signals are pre-processed and aggregated into measurement channels. One Measurement Channel describes how samples from one or more sensor signal are aggregated and measured. Various types of data channels are supported (Time Series, Histogram etc.).

Data Package Layer Specification

Data Packages provide a structure for storing and retrieving data. Data Packages contain the actual data of Signal measurements. They cover the Signals as information providers and the Measurement Channels defining the process of the data acquisition of these Signals. In addition, Data Package provide meta/header information containing time of recording, data ownership information, etc. Data Packages contain data from exactly one Measurement Channel.

3 CIDM Data Provision Tailored to the Service Needs

CPPs can provide a huge amount of data to the outside world. E.g. inside today's vehicles ~4000 bus-signals at a sampling interval of about 10ms are theoretical accessible, corresponds to 2.68 GB/hour. Forwarding that amount of data for thousands of vehicles from various brands would overstress the communication links for the data exchange as well as requiring huge amount of storage areas. Even when technical feasible, such an approach is not reasonable from an economic point of view. Therefore, there is the essential need to reduce amount of data to be transferred and stored.

On the other hand data model has to enable the Service Providers to access CPP data optimal tailored to the needs of their services, in respect to the required signals, sampling rates, type of representations etc. The CIDM offers a wide spectrum of data representations on the various model layers, tailored to the needs of the Service Providers and on the other hand enabling to constrain data volumes to be communicated and stored.

Accessible Signals

Signals represent the in-CPP provider of observations, raw sensors measurements and CPP status information. Signals can vary between different type of CPPs, different brands, models or even CPPs of the same model may differ by value range, sampling rate or resolution etc. Different types of signals both continuous and discrete data (like wipers on/off, outside temperature, room temperature, sun intensity etc.) as well as textual (e.g. Vehicle model) and enumeration (e.g. fuel type) are supported. The CIDM Model enables that the list of signals handled by the CIDM can be modified/extended.

Confirmed by the CPP Manufacturer, each product provides a specific set of signal accessible in the CIDM format. For each type of CPP the accessible signals are specified. However, some manufacturer might not provide all specified data due to missing sensors in the product model or company confidentiality etc.

Types of Data Representation (Measurement Channels)

To minimise the amount of data exchanged and stored the data contained in signals are aggregated and pre-processed, represented by so called Measurement Channels. By several Measurement Channels the CIDM supports various data representations tailored to the needs of the Service Provider community. Examples of CIDM Measurement Channels are presented in Figure x.

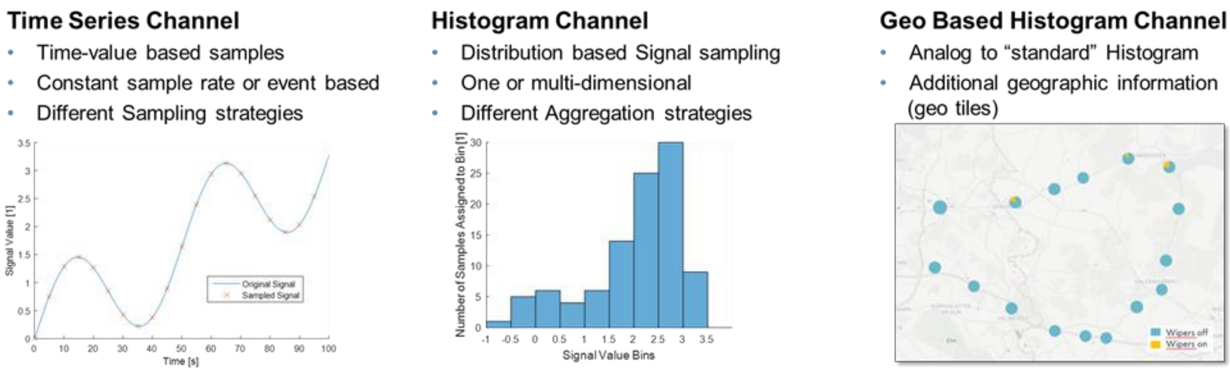


Figure 3: Examples of measurement channels

Typical measurement channels of the CIDM address:

- **Time Series:** Sequences of data recordings (samples) from a signal in a time order with a constant sampling rate or event based. Each data sample is assigned to a specific point in time, respective geo location may be recorded, too. Furthermore, min/max and averaging processing features are supported. Events might represent switching status or parameter thresholds etc. By a service tailored sampling rate, as well as the min/max and event based features the data volumes can be reduced.

- **Histogram:** Distribution of sampled sensor data, providing a rough sense of the density of the underlying distribution of the sampled sensor data. Thereby, the entire value range is divided into a series of intervals and it is counted how many values fall into each interval. CIDM also support multidimensional histograms. Histograms may be created over different time ranges so called "Capture Interval". They may have dimension of days, months etc. Histograms enable a very high sampling rate but an enormous reduction of the data to be stored and retrieved. However, there is a loss of reference in respect to time and location of the samples.

- **Geo-based Histogram:** The concept of geo-histograms enriches the histogram approach by a geospatial dimension. To the classic histogram information (how often a certain value occur in a certain time period) additionally the information of where a certain histogram was observed during the observation period is recorded. The additional geographic information is represented by geo-tiles, embodying a certain map tile based on the Mercator projection.

The CIDM architecture is open to add additional Measurement Channels if required for the analysis of available signals by the Service Providers for their services.

Data Package the Transport Format

On the Data Package layer, several measurement channels are aggregated in Data Packages to be stored and retrieved. In addition to the data themselves, Data Packages also contain metadata, with support information like ownership and quality assessment etc. Therefore, the CVIM Data Package is divided in two parts: (i) descriptive metadata and (ii) the aggregated data. (see Figure xx). Metadata support indexing and sorting the data by leveraging the statistic properties, timestamps and geographic estimate in form of a bounding box. In addition, it provides ownership information, privacy levels and data stakeholders. For the sake of completeness and validity, CPP manufactures can sign data packages using sequence numbers, checksums and signature. The second part of Data Packages stores aggregated data of Measurement Channels. This may be either a time series recording or pre-processed value distribution in form of a histogram etc.

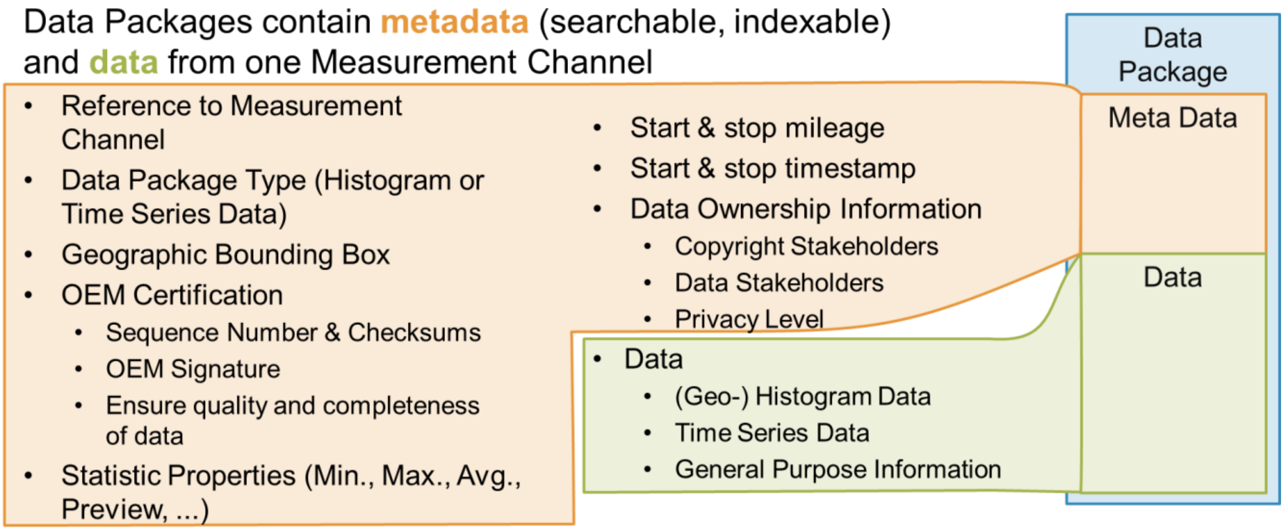


Figure 4: Data Package structure

Finally, it is important to mention that each CPP needs to transmit not just sensor information, but also information about the CPP itself – i.e. basic CPP information. To illustrate, it may be relevant to know the colour of a vehicle, the materials or number of windows in a room as this may condition the readings of temperature sensors: e.g. a black vehicle under the sun will probably yield much hotter estimations. The same happens for a room in which too many windows are located. Note that such basic CPP information could be attached as additional parameters to the existing measurement channels; yet, basic CPP information is probably not going to change frequently, and therefore they do not need to be sent in every reading. Therefore, the Data Package passes through the creation of a specific basic CPP information channel, in which one basic CPP information measurement is transmitted at the beginning– or whenever an update is made necessary.

In fact, the CPP model explained was conceived for the Cross-CPP project in which the main entities exchanging information where vehicles and buildings. However, the widely use of this model as the data model in other domains such as aeronautics, smart manufacturing, oceanographic studies is almost straight forward as only the new channels will have to be defined. The model could even be shared among competitors.

4 CIDM Represents a Living Standard

The brand-independent, open and transparent CIDM standard is not rigid, but rather representing a living standard. Whenever, it is required in reference to the market needs of the service provider community to extent the amount of signals to be recorded or to modify/extent the type of measurement channels the CIDM standard can be adapted to match these requirements. This feature of the CIDM standard to permit respective extension/modifications represents a key measure to enlarge significantly the number of data customers.

A standardisation board composed by the key shareholders of Cross CPP Ecosystem will decide about the CPP data model reconfiguration requests of the Service Providers and an updated version of the CIDM will be agreed.

CROSS-CPP

About Cross-CPP

The objective was to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research was to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project developed: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to provide enhanced local weather forecast and predict and optimise energy consumptions in households. The project has built upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, were used as a basis for Cross-CPP development extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

Funded by the Horizon 2020 Framework Programme of the European Union

9 Annex 8: CIDM Open Specification

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

CIDM

Common Industrial Data Model

CIDM Common Industrial Data Model

Open Specification

November 2020

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780167

Foreword

The key motivation of the Cross-CPP project is to develop a unified set of services with which to man-age cross-sectorial data streams. This would enable its exploitation through an IT environment in which the data is collected and stored, serving as both a provider of features (e.g. detection of anomalies in on-board sensors) and an enhancer of those which already exist (e.g. weather forecasting, energy optimisation…).

Thus, this platform bridges the gap between industries (from a wide range of ambits, such as automotive or IoT) and the access to valuable, high volume data that is being continuously generated by products of their interest (vehicles, smart home devices, etc.). This uninterrupted stream of data from massively produced items will enable both the combination of data streams from various sources and an interdisciplinary Big Data-driven business potential for both manufacturers and IT organisations.

However, the viability of these business opportunities is currently hindered by how fragmented the industrial environment is with respect to proprietary software approach that prevails. This prevents from developing clear models and tools for the mentioned cross-sectorial collaboration, as it limits access to data from devices of interest and impedes the development of agreed terms of use, privacy, etc.

In contrast, the Cross-CPP project focuses on developing a readily available source of data and associated services to the outside world. Therefore, the establishment of an operative cross-sectorial Big Data Ecosystem is founded upon three fundamental pillars. Namely, its openness for the integration of diverse data providers (including the provision of a standardized cross-sectorial data model), its operation via a Big Data marketplace and a controlled access to the available data streams (including data ownership concerns).

Cross-CPP consortium partners

Executive Summary

In accordance with the purpose of collecting data from vehicles and buildings (property of various manufacturers), Cross CPP requires a common understanding of data and information. Within the scope of the project, the Common Industrial Data Model (CIDM) was developed as open and highly scalable automotive big data format. The CIDM that is the extension of the CVIM of the Automat project. The CIDM provides a brand-independent and transparent data model, which harmonizes proprietary data into generic datasets. However, the CIDM is not rigid, rather representing a living data structure, where in reference to the needs of the service provider community the amount of signals to be recorded as well as the type of measurement channels can be modified or extended.

This document comprises the specification of the CIDM and describes a key result of the Cross CPP project. A machine-readable CIDM definition is publicly available as part of the SDK.

This specification document is structured as follows:

1 Terminology and Format of the CIDM Specification

1.1 Terminology of Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

1.2 Specification Format

The specification is provided in text form throughout the following sections and in a machine-readable format as part of the SDK, which is publicly available at github. This enables automated testing and validation of CIDM data and Cloud Storage Provider Interfaces. The CIDM specification and definition exploits the following formats and standards:

- CIDM Relationship Modelling: Unified Modeling LanguageTM (UML), Version 2.5
- CIDM Model Definition: JSON Schema, Version 4
- CIDM Reference and Prototype Implementation: JavaScript Object Notation (JSON)
- Cloud Storage Provider Interface Specification: OpenAPI, Version 3.0
- Cloud Storage Provider Prototype Implementation: Python, Version 3.6

1.3 JSON Schema Datatype Extension

JSON Schema allows detailed and rich data model descriptions. Nevertheless, it supports the following seven primitive data types for JSON values:

- Array – a JSON array
- Boolean – Value may be true or false
- Integer – Number without fraction or exponent part
- Number – Any number, Number includes integer
- Null – Null (empty) value
- Object – JSON object
- String – Text value

To improve and validate contents of fields the JSON Schema Validation defines semantic validation using the format attribute. Table 1 further extends this format attribute to provide a more detailed description of data types within the CIDM definition and Cloud Storage Provider specification

Table 1. Extension of JSON Schema Format Attributes.

Value Type (JSON Schema) “type” attribute	Format “format” attribute	Description	Example
integer	int8	signed 8-bit integer	- 128
integer	int16	signed 16-bit integer	32767
integer	int32	signed 32-bit integer	- 2147483648

integer	int64	signed 64-bit integer	1
integer	uint8	unsigned 8-bit integer	255
integer	uint16	unsigned 16-bit integer	65535
integer	uint32	unsigned 32-bit integer	4294967295
integer	uint64	unsigned 64-bit integer	0
number	double	single precision floating point value representing a 32-bit binary format IEEE 754 value	-3.1415926, +Infinity
number	double	double precision floating point value representing a 64-bit binary format IEEE 754 value	3.1415926, - Infinity
string	date	As defined by full-date in RFC3339	2016-04-07
string	date-time	As defined by date-time in RFC3339	2016-04- 07T11:45:13.01Z
string	uuid	Universally unique identifier (UUID representing a 128-bit unique value in a hexadecimal representation	664f4826-9563-4a56- 9d73-c8cf55572fd5
string	email	E-Mail address representation	mail@cross-cpp.eu
string	version	CIDM Version Format with Schema a.b.c: - a: Major Version - b: Minor Release - c: Revision	1.2.1

2 The Cross-CPP Cross-Industrial Data Model

This section defines the Cross CPP data model, that is the extension of the CVIM of the Automat project. The main changes of the data model involve:

- Extension of CVIM towards CIDM, to support various CPP data sources
- Inclusion of new signal list related to CPP type ‘Smart Building’
- Extension by new types of measurement channels (event based & Basic CPP Information)
- Creation of CPP specific measurement channel lists (extended vehicle lists & new smart building lists)
- Modification of the timestamp requirements to support single value data packages

The following sections 3.2, 3.3 and 3.4 describe the Signal layer, the Measurement layer and the Data layer respectively.

2.1 Model Architecture

The FP CIDM architecture has the same three layers architecture as the CVIM. The Signal layer contain Signals that are the information providers within the CPP devices like vehicles or smart buildings. Signals observe the environment and generate data, they detect physical and chemical phenomenon, for example, speed, temperature, charge state level, etc.

The Measurement layer provides signals’ data aggregation. The data needs to be pre-processed since raw sensor data exceeds the available storage and transferring capacity, to reduce the size of data down-sampling and histograms methods are provided. The CVIM specification provide detailed information about the applied methods.

The Data Layer aggregates data inside data packages to store and transfer. “*One data package contains data from exactly one signal measured with one Measurement Channel. In addition to the actual data, Data Packages contain header information (“meta data”). This header information provides ownership of the data and gives quality of signal indications by OEM signatures or describes parameters of the measurement (e.g. time, rough position estimate, etc.)*”.

Figure 1 shows the high-level architecture of the CIDM. The bottom layer contains Signals. The next layer is the Measurement Layer. The top layer is the data layer.

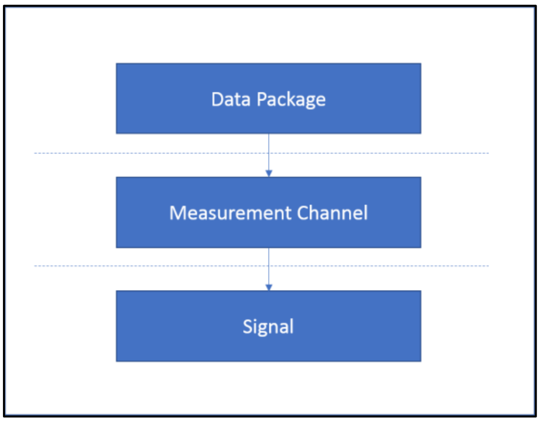


Figure 1: Layered High-level View of the Common Industrial Data Model (CIDM)

2.2 Signal Layer Specification

Sensors are the perception organs of CPP devices like vehicles and buildings. It is their main duty to detect physical phenomenon and chemical quantities by transferring them into electrical signals. The signal layers describe different types of signals and formats represented in the system. A new property is needed to group signals regarding the signal source type, cpp-type. Figure 2 shows the UML modelling of the signals for CIDM.

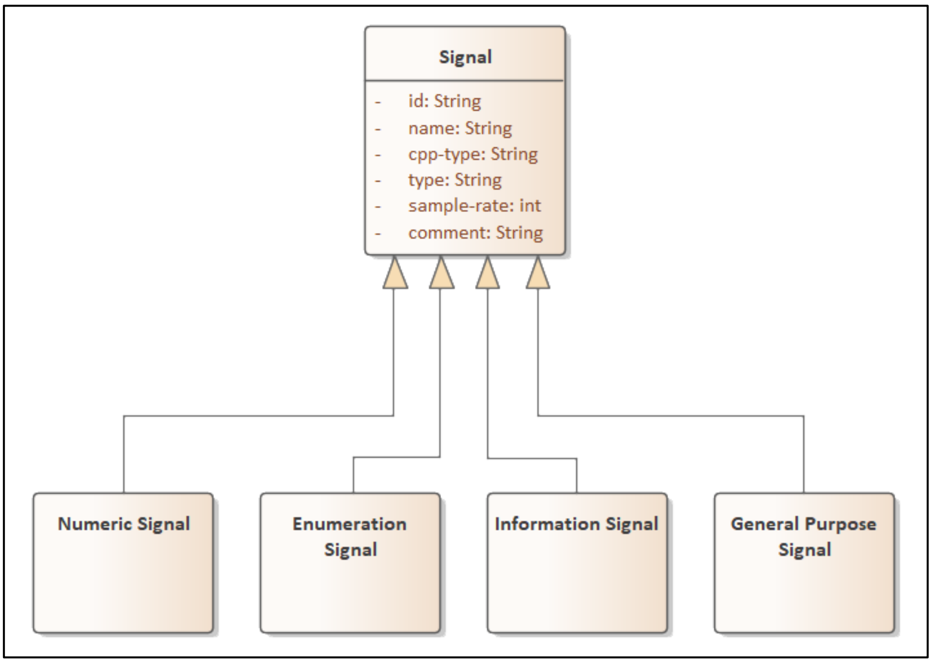


Figure 2. Signal UML Model.

The cpp-type is a required property that must be one of the two values, “vehicle” or “building”. Table 2 shows the complete definition of the Signal.

Table 2. Signal property definition.

Property	Occur-rence	Type	Format	Description
Common Properties				
`id`	Required	String		Unique Identifier of the Signal
`name`	Required	String		Name of the Signal
`cpp-type`	Required	String	one of: - vehicle - building	Type of the CPP
`type`	Required	String	one of: - numeric - enumeration - information - general-purpose	Type of the Signal

`format`	Optional	String		Signals representation format
`sample-rate`	Required	Numeric	double	Sample rate in Hz (Samples per Second). Must be larger than or equal to zero.
`comment`	Optional	String		Description of the signal
Numeric Signal				
`type`	Required	String	"numeric"	Type of the Signal needs to be numeric
`format`	Required	String	*<numeric formats>*	Signal's numeric representation (e.g. uint8, double, etc.)
`min`	Required	Number	<according to format>	Minimum Signal value
`max`	Required	Number	<according to format>	Maximal Signal value
`resolution`	Required	Number	<according to format>	Signals resolution
`unit`	Required	String		Unit of the Signal (e.g. ºC)
Enumeration Signal				
`type`	Required	String	"enumeration"	Signal's type attribute needs to be "enumeration"
`items`	Required	Array	String	String array with possible Signal values
Information Signal				
`type`	Required	String	"information"	Signal's type attribute needs to be "information"
`format`	Required	String		Signals representation format (e.g. VIN, etc.)
General Purpose Signal				
`type`	Required	String	"general-purpose"	Signal's type attribute needs to be "general-purpose"
`*`	Optional	Any	No	May be extended with further attributes

2.3 Measurement Layer Specification

"*The measurement layer defines how sensor signals are captured and processed. One Measurement Channel describes how samples from one (or more - in the case of multidimensional histograms) sensor signal are aggregated and measured*" (TUDO, Consortium Partners, 2017). Figure 3 shows the Measurement Channel UML model.

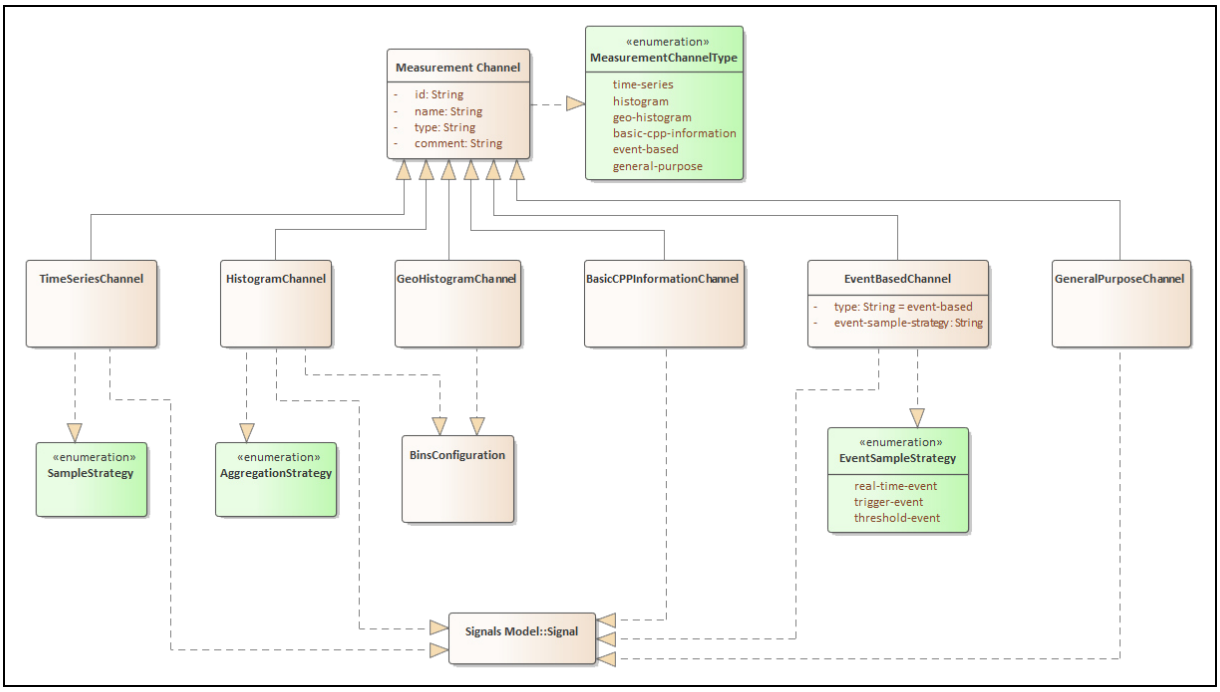


Figure 3. Measurement Channel UML Model.

The basic CPP information channel provides static information that is not measured by sensors but provides information about the CPP device, like the colour of a vehicle, the number of floors of a building, the identification number of a car, etc.

The event-based measurement channel provides information of events that occurs when the value of a measurement gets to a specific value (real-time-event) or when the value passes a specific threshold (threshold) and the last one that provides information about the event and the data-packages that have triggered the event. Table 3 details the measurement channel specification.

Table 3: Measurement channel definition

Property	Occur-rence	Type	Format	Description
Common Properties				
`id`	Required	String	No	Unique Identifier of the Measurement Channel
`name`	Required	String	No	Name of the Measurement Channel
`type`	Required	String	one of: - time-series - histogram	Type of Measurement Channel

			- geo-histogram - general-purpose - event-based - basic-cpp-information	
`comment`	Optional	String	No	Description of the signal
Time Series Measurement Channel				
`type`	Required	String	“time-series”	Type of the Measurement Channel needs to be time-series
`format`	Required	String		Data type format of the samples
`dimension`	Optional	Number	uint32	Dimension of the Time-series. If dimension is not given, one-dimensional is assumed
`capture-interval`	Required when on-change is false	Number	double	Capture interval between two samples in seconds. Only required, when on-change is false.
`on-change`	Required	Boolean		Does Measurement-Channel only record changes in signal
`sample-strategy`	Required	String	one of: - min - max - average - last-known-value	Signal sampling strategy
`signal`	Required	Object	Array of Signal Object	See section 4.3 for Signal object definition
Histogram AND geo-Histogram Measurement Channel				
`type`	Required	String	one of: - histogram - geo-histogram	Type of the Measurement Channel needs to be histogram or geo-histogram
`aggregation-strategy`	Required	String	one of: - time - count - min - max	Histogram values aggregation strategy
`capture-interval`	Required	Number	double	Capture Interval of one Histogram. Needs to be

				larger than zero. +Infinity is valid (see IEEE 754).
`dimensions`	Required	Number	uint32	Dimensions of the Histogram
`bins`	Required	Array	Bin-Configuration Object	Array of bin configurations. Array needs to contain exactly one configuration for every dimension/axes of the histogram. See also section 4.4.1.1
Geo-Histogram Measurement Channel				
`type`	Required	String	"geo-histogram"	Type of the Measurement Channel needs to be geo-histogram
`geo-resolution`	Required	Numeric	double	Zoom level of the geo-histogram according to section 4.2.1.3
Basic CPP Information Measurement Channel				
`type`	Required	String	"basic-cpp-information"	Type of the Measurement Channel needs to be basic-cpp-information
`signal`	Required	Object	Signal Object	See section for Signal object definition
Event Based Measurement Channel				
`type`	Required	String	"event-based"	Type of the Measurement Channel needs to be event-based
`format`	Required	String		Data type format of the samples
`event-sample-strategy`	Required	Event Sample	one of: - real-time-event - trigger-event - threshold-event	Event sampling strategy
`comment`	Optional	String	No	Description of Event Strategy
General Purpose Measurement Channel				
`type`	Required	String	"general-purpose"	Type of the Measurement Channel needs to be general-purpose

`signal`	Required	Any		See section 0 for Signal definition

2.4 Data Layer Specification

"*Data Packages contain the actual data of Signal measurements. As Signals are the information providers and Measurement Channels define the process of data acquisition from those Signals, Data Packages provide a structure for storing the data. In addition, they provide meta / header information containing time of recording, data ownership information, etc. Data Packages contain data from exactly one Measurement Channel*". This leads to six different types of Data Packages that are defined similar as the Measurement Channels:

- Time Series Data Package
- Histogram Data Package
- Geo-Histogram Data Package
- Event based data Package
- Basics CPP information Data Package
- General Purpose Data Package

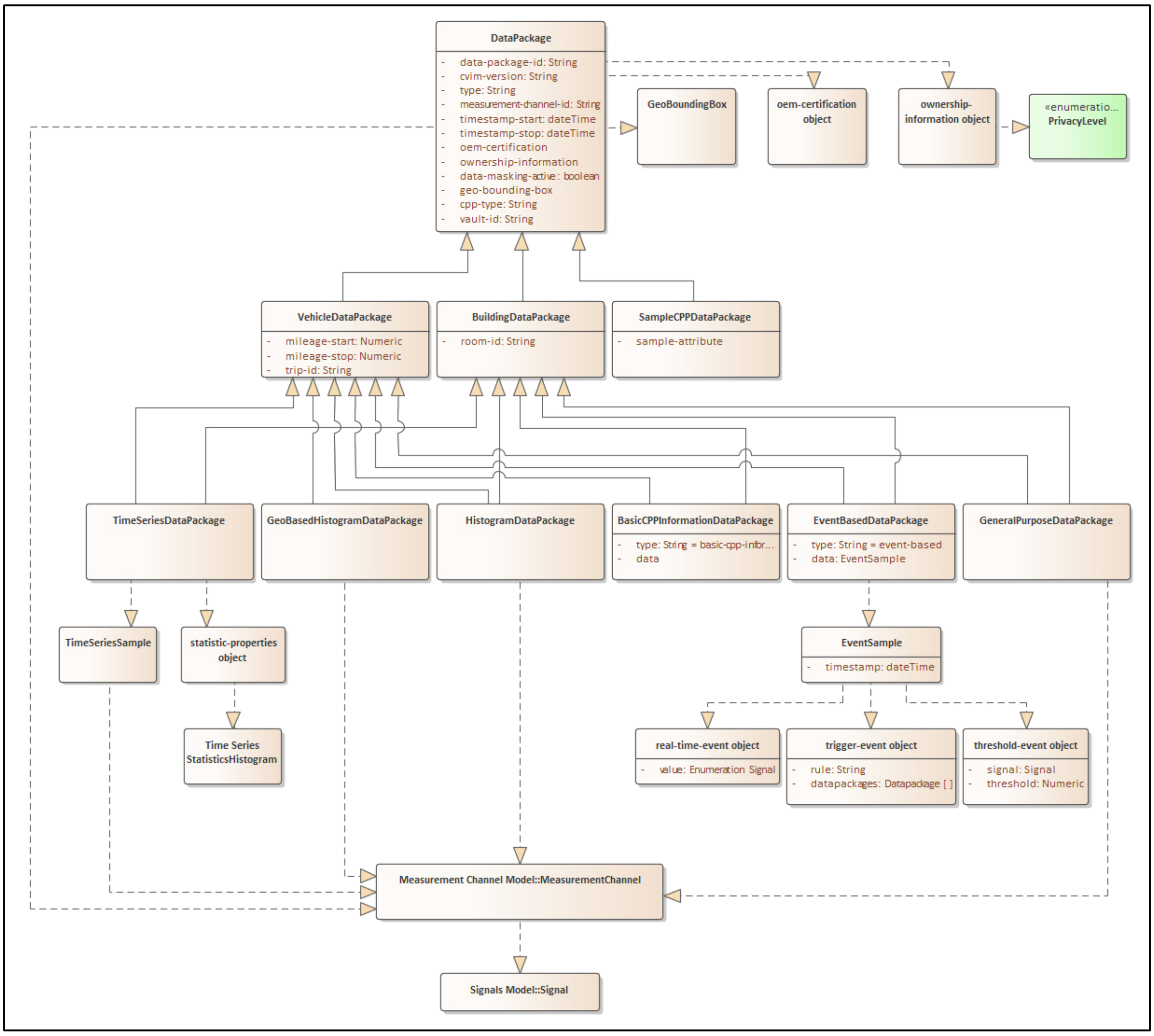


Figure 4. Data Package UML Model.

The new data-package definition includes cpp-type to indicate is the data-package belongs to a “building” or a “vehicle” cpp and two new definition of data-package type, basics-cpp-information data-package and the event-based data-package.

The data of basics-cpp-information data-package depends of the signal type definition of the measurement channel - numeric, enumeration, information or general-purpose.
Event-based data-package has an additional property, “event-sample-strategy”, to indicate three different type of event, real-time, trigger and threshold. According to the type of event, the “data” object has two mandatory properties, “timestamp” and “value”, and an optional property named “datapackages” that is an array of the data-packages that triggers the “trigger-event”. See Table 4 and Table 5 for the entire specification.

The “building” CPP devices provides a set of sensors distributed along the rooms of the building, in order to identify the devices of the same room a new property has been included, “room-id”.

Table 4. Data Package definition.

Property	Occurrence	Type	Format	Description
Common Properties				
data-package-id	Required	String	UUID	Identifier of the Data Package. Unique per Cloud Storage Vault, Set by Cloud Storage Provider
cvim-version	Required	String	version	The name of the property is for backward compatibility with CVIM. Version 0.0.1 to 1.2.0 are CVIM and CIDM from 1.2.1
type	Required	String	one of: - time-series - histogram - geo-histogram - general-purpose - event-based - basic-cpp-information	Type of the Data Package
vault-id	Required	String	UUID	ID of the Cloud Storage Vault, where the data is stored in.
cpp-id	Optional	String	any	ID of the CPP
cpp-type	Required	String	one of: - vehicle - building	Type of the CPP
trip-id	Optional	String	any	Trip-ID of the User
room-id	Optional	String	any	ID of the room in a building where the measurement data was collected
measurement-channel-id	Required	String		Identifier of the Measurement Channel whose data is inside this data package
mileage-start	Optional	Number	double	Mileage at the start of the measurement in kilometres (km)
mileage-stop	Optional	Number	double	Mileage at end of measurement (km)
geo-bounding-box	Optional	Object	Geo-Bounding-Object	Geographic bounding box, see section 4.5.1.1

Property	Occurrence	Type	Format	Description
Common Properties				
location	Optional	Object	Location-Object	Single location including latitude and longitude
oem-certification	Optional	Object	OEM-Certification-Object	OEM Certification, see section 4.5.1.2
data-ownership-information	Optional	Object	Ownership-Information - Object	Data Ownership Information, see section 5.4.1.3
expiration-date	Optional	String	date-time	Data expiration date
data-masking-active	Optional	Boolean		Indicates status of data-masking (true = active)
Time Series Data Package				
type	Required	String	"time-series"	Type of the Measurement Channel needs to be time-series
timestamp-start	Required	String	date-time	Measurement start time
timestamp-stop	Required	String	date-time	Measurement stop time
number-of-samples	Required	Number	uint32	Number of samples that are stored in data
statistic-properties	Optional	Object	statistic-properties-object	Provides statistic properties about the data, see section 4.5.1.4
data	Required	Array	time-series key-value-pair - object	Array of time-series-data Objects. The size of the array mist equal number of samples, see section 0
Histogram Data Package				
type	Required	String	histogram	Type of the Measurement Channel needs to be histogram
timestamp-start	Required	String	date-time	Measurement start time
timestamp-stop	Required	String	date-time	Measurement stop time

Property	Occurrence	Type	Format	Description
Common Properties				
`data`	Required	(Multi-dimensional) Array	Number	Array containing the bin counts. Size of array must match the dimension and bin configuration of the related Measurement Channel. Number format depends on Histogram aggregation-strategy
Geo-Histogram Data Package				
`type`	Required	String	geo-histogram	Type of the Measurement Channel needs to be geo-histogram
`timestamp-start`	Required	String	date-time	Measurement start time
`timestamp-stop`	Required	String	date-time	Measurement stop time
`data`	Required	(Multi-dimensional) Array	Number	Array containing the bin counts. Size of array must match the dimension and bin configuration of the related Measurement Channel. Number format depends on Histogram aggregation-strategy. The outer most dimension is the geo-dimension. It must match in its size the size of the geo-tiles array.
`geo-tiles`	Required	Array	Geo-Tile Object	Array of geo-tile objects. Only visited tiles are included.
Basic CPP Information Data Package				
`type`	Required	String	"basic-cpp-information"	Type of the Measurement Channel needs to be basic-cpp-information
`timestamp`	Required	String	date-time	Measurement date time
`data`	Required	Any		Data depends on the type of signal of the measurement channel

Property	Occurrence	Type	Format	Description
Common Properties				
Event Based Data Package				
`type`	Required	String	"event-based"	Type of the Measurement Channel needs to be event-based
`timestamp`	Required	String	date-time	Measurement date time
`event-sam-ple-strategy`	Required		one of: - real-time-event - trigger-event - threshold-event	Event sampling strategy
`data`	Required	Object	Event Sample Object	event-based data Object indicating an event
General Purpose Data Package				
`type`	Required	String	"general-purpose"	Type of the Measurement Channel needs to be general-purpose
`timestamp`	Required	String	date-time	Measurement date time
`data`	Required	Any	time	Datatype depends on Measurement Channel

Table 5. Event Sample Object

Property	Occurrence	Type	Format	Description
`timestamp`	Required	timestamp	date-time	Timestamp of the event
`value`	Required	string		e.g. *"Ignition On"*, *"Wipers Off"*
`datapackages`	Optional	Array of datapackage		

3 Annex

3.1 OpenAPI Specification of the Company Backend REST API (yaml)


```yaml
---
swagger: "2.0"
info:
  version: 3.0.0
  title: Cloud Storage Provider API Definition
  contact:
    name: Elisa Herrmann
    url: http://www.cross-cpp.eu
    email: elisa.herrmann@atos.net
host: cloudstorage-api.datagora.eu
basePath: /
schemes:
- https
consumes:
- application/json
produces:
- application/json
security:
- api_key: []
paths:
  /users:
    get:
      tags:
      - User Management
      summary: Retrieve all users
      description: This functionalities provides a list of all users within this Cloud Storage Pro-
vider. This API call is not specified and may only be used by the Cloud Storage Provider for
internal user management. Returns an array of `User` objects.
      operationId: usersGET
      parameters: []
      responses:
        "200":
          description: OK
          schema:
            type: array
            items:
              $ref: '#/definitions/FullUser'
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
    post:
```

```
      tags:
      - User Management
      summary: Create new user
      description: API call registers a new user and creates new Cloud Storage Vault at the Provider. This API call is not specified and is intended to be used for internal user management.
      operationId: usersPOST
      parameters:
      - in: body
        name: User
        description: Defines full name, login and password of the user.
        required: true
        schema:
          $ref: '#/definitions/User'
      responses:
        "200":
          description: OK
          schema:
            $ref: '#/definitions/FullUser'
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "409":
          description: Conflict - User already exists
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /users/access:
    post:
      tags:
      - Contract Management
      summary: Grant access permission
      description: Acquires access permissions to the `api_key` that is used in this request. If `key`is a `vault-write-key` `write permission`is granted, if `key`is `vault-read-key` `read permission`is granted.
      operationId: usersAccessPOST
      parameters:
      - in: body
        name: key
        description: (Read or write) Access key to one Cloud Storage Vault.
        required: true
        schema:
          $ref: '#/definitions/key'
      responses:
        "200":
          description: OK
          schema:
            $ref: '#/definitions/inline_response_200'
        "400":
          description: Bad request
        "401":
```

```
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /users/access/{key}:
    get:
      tags:
      - Contract Management
      summary: Validate access token
      description: tbd
      operationId: usersAccessGET
      parameters:
      - name: key
        in: path
        description: Access key for Cloud Storage Vault
        required: true
        type: string
        format: uuid
      responses:
        "200":
          description: OK
          schema:
            $ref: '#/definitions/inline_response_200_1'
      x-swagger-router-controller: Default
    delete:
      tags:
      - Contract Management
      summary: Release access permissions
      description: Releases access permissions of the `api_key` that is used in this request. If `key`is a `vault-write-key` `write permission`is released, if `key`is `vault-read-key` `read permission`is released.
      operationId: usersAccessKeyDELETE
      parameters:
      - name: key
        in: path
        description: Access key for Cloud Storage Vault
        required: true
        type: string
        format: uuid
      responses:
        "200":
          description: OK
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
```

```yaml
        description: Not found
      default:
        description: Unexpected error
    x-swagger-router-controller: Default
  /datapackages:
    post:
      tags:
      - Data Storage Interface
      summary: Push data packages into Cloud Storage
      description: This API call enables OEMs to write data packages into the cloud. Data packages are stored within a CIDM container structure. Every data package needs to contain the correct vault-id. The OEM needs write permission to the user's Cloud Storage Vault. The Cloud Storage Provider assignes an unique datapackage-id to every delivered data package.
      operationId: datapackagesPOST
      parameters:
      - in: body
        name: Data Package Container
        description: CIDM container structure containing valid and json encoded CIDM data packages
        required: true
        schema:
          type: array
          items:
            $ref: '#/definitions/DataPackage'
      responses:
        "200":
          description: OK
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /datapackages/{datapackageid}:
    get:
      tags:
      - Data Provisioning Interface
      summary: Get data package
      description: Retrieve one data package with `datapackage-id`.
      operationId: datapackagesIdGET
      parameters:
      - name: datapackageid
        in: path
        description: Unique data package identifier
        required: true
        type: string
      - name: metadata
        in: query
        description: Retrieve *only* metadata, default=false
        required: false
```

```
          type: boolean
          default: false
        responses:
          "200":
            description: OK
            schema:
              $ref: '#/definitions/DataPackage'
          "400":
            description: Bad request
          "401":
            description: Unauthorized
          "403":
            description: Forbidden
          "404":
            description: Not found
          default:
            description: Unexpected error
        x-swagger-router-controller: Default
    /datapackages/query:
      post:
        tags:
        - Data Provisioning Interface
        summary: Query data packages
        description: Allows searching for data packages. Cloud Storage Provider will only include
Cloud Storage Vaults into search where `api_key` has read access.
        operationId: datapackagesQueryPOST
        parameters:
        - in: body
          name: query
          description: Query for data packages
          required: true
          schema:
            $ref: '#/definitions/Query'
        responses:
          "200":
            description: OK
            schema:
              type: array
              items:
                $ref: '#/definitions/DataPackage'
          "400":
            description: Bad request
          "401":
            description: Unauthorized
          "403":
            description: Forbidden
          "404":
            description: Not found
          default:
            description: Unexpected error
        x-swagger-router-controller: Default
    /datapackages/query_stream:
      post:
```

```yaml
      tags:
      - Data Provisioning Interface
      summary: Query data packages
      description: Allows searching for data packages. Cloud Storage Provider will only include Cloud Storage Vaults into search where `api_key` has read access.
      operationId: datapackagesQueryStreamPOST
      parameters:
      - in: body
        name: query
        description: Query for data packages
        required: true
        schema:
          $ref: '#/definitions/Query'
      responses:
        "200":
          description: OK
          schema:
            type: array
            items:
              $ref: '#/definitions/DataPackage'
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /datapackages/basicCppQuery:
    post:
      tags:
      - Data Provisioning Interface
      summary: Query basic CPP information data packages
      description: Allows searching for data packages. Cloud Storage Provider will only include Cloud Storage Vaults into search where `api_key` has read access.
      operationId: basicCppQueryPOST
      parameters:
      - in: body
        name: query
        description: Query for data packages
        required: true
        schema:
          $ref: '#/definitions/Query'
      responses:
        "200":
          description: OK
          schema:
            type: array
            items:
              $ref: '#/definitions/DataPackage'
```

```
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /datapackages/basicCppQuery_stream:
    post:
      tags:
      - Data Provisioning Interface
      summary: Query basic CPP information data packages
      description: Allows searching for data packages. Cloud Storage Provider will only include
Cloud Storage Vaults into search where `api_key` has read access.
      operationId: basicCppQueryStreamPOST
      parameters:
      - in: body
        name: query
        description: Query for data packages
        required: true
        schema:
          $ref: '#/definitions/Query'
      responses:
        "200":
          description: OK
          schema:
            type: array
            items:
              $ref: '#/definitions/DataPackage'
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /notifications:
    post:
      tags:
      - Notifications
      summary: Subscribe Push Notification
      description: Functionality of the marketplace to subscribe to push notification events. Push
notifications are sent, whenever users put data into their Cloud Storage Vaults.
      operationId: notificationsPOST
      parameters:
```

```
      - in: body
        name: config
        description: Push Notification URL
        required: true
        schema:
          $ref: '#/definitions/config'
      responses:
        "200":
          description: OK
        "400":
          description: Bad request
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
    delete:
      tags:
      - Notifications
      summary: Unsubscribe Push Notification
      description: No more push notifications will be sent from the Cloud Storage Provider to the
marketplace.
      operationId: notificationsDELETE
      parameters: []
      responses:
        "200":
          description: OK
        "401":
          description: Unauthorized
        "403":
          description: Forbidden
        "404":
          description: Not found
        default:
          description: Unexpected error
      x-swagger-router-controller: Default
  /access:
    get:
      tags:
      - Contract Management
      summary: Validate authentication token
      description: t.b.d.
      operationId: accessGET
      parameters: []
      responses:
        "200":
          description: OK
          schema:
            $ref: '#/definitions/inline_response_200_2'
        "403":
          description: Forbidden / No authorization token in header
```

```
        "404":
          description: Authorization token not found!
        "500":
          description: Internal Error
      x-swagger-router-controller: Default
securityDefinitions:
  api_key:
    description: Provides authentification for OEM and Marketplace. Must be sent in HTTP
header.
    type: apiKey
    name: Authentication
    in: header
definitions:
  FullUser:
    allOf:
    - $ref: '#/definitions/User'
    - {}
  User:
    type: object
    required:
    - full-name
    - login-name
    - password
    properties:
      full-name:
        type: string
        description: Full name of the user
      login-name:
        type: string
        format: email
        description: Login name of the user (e-mail)
      password:
        type: string
        description: Password of the user
    example:
      password: password
      full-name: full-name
      login-name: login-name
  DataPackage:
    type: object
    required:
    - cpp-type
    - cvim-version
    - data
    - measurement-channel-id
    - type
    - vault-id
    properties:
      vault-id:
        type: string
        description: Cloud Storage Vault ID, where the data packages are pushed into.
      datapackage-id:
        type: string
```

```
          description: Unique identifier of the data package. Property is set by Cloud Storage Pro-
vider.
          readOnly: true
        data:
          type: object
          description: CIDM data
          properties: {}
        cvim-version:
          type: string
          description: Version of the CIDM protocol >=1.1.2
        type:
          type: string
          description: Type of the Data Package
        cpp-type:
          type: string
          description: Type of the CPP device, "vehicle" or "builging"
        cpp-id:
          type: string
          description: CPP ID for identifying vehicles or bulding of the same owner and vault-id
        measurement-channel-id:
          type: string
          description: Identifier of the Measurement Channel whose data is inside this data pack-
age
        timestamp:
          type: string
          description: Measurement timestamp (basic-cpp-information and event-based)
        timestamp-start:
          type: string
          description: Measurement start time
        timestamp-stop:
          type: string
          description: Measurement stop time
        mileage-start:
          type: number
          description: Mileage at the start of measurement in kilometres (km)
        mileage-stop:
          type: number
          description: Mileage at the end of measurement in kilometres (km)
        geo-bounding-box:
          type: object
          description: geographic bounding box (see reference manual section 6.5.1.1)
          properties: {}
        room-id:
          type: string
          description: Identifier of the room in a building cpp-type
        oem-certification:
          type: object
          description: OEM certification (see reference manual section 6.5.1.2)
          properties: {}
        ownership-information:
          type: object
          description: Data Ownership Information (see reference manual section 6.5.1.3)
          properties: {}
```

```yaml
      expiration-date:
        type: string
        description: Data expiration date
      data-masking-active:
        type: boolean
        description: Indicates status of data-masking (true = active)
    example:
      datapackage-id: datapackage-id
      vault-id: vault-id
      data: '{}'
      data-masking-active: true
      cvim-version: cvim-version
      type: type
      mileage-stop: 6.027456183070403
      measurement-channel-id: measurement-channel-id
      timestamp-start: timestamp-start
      geo-bounding-box: '{}'
      mileage-start: 0.8008281904610115
      oem-certification: '{}'
      ownership-information: '{}'
      timestamp-stop: timestamp-stop
      expiration-date: expiration-date
  Query:
    type: object
    properties:
      datapackage-id:
        type: array
        description: Array of Data Package IDs
        items:
          type: string
          description: Data Package ID
      measurement-channel-id:
        type: array
        description: Array of Measurement Channel IDs
        items:
          type: string
          description: Measurement Channel ID
      vault-id:
        type: array
        description: Array of Cloud Storage Vault IDs
        items:
          type: string
      submit-time:
        $ref: '#/definitions/Query_submittime'
      metadata:
        type: boolean
        description: Request only metadata, default=off
        default: false
    example:
      datapackage-id:
      - datapackage-id
      - datapackage-id
      submit-time:
```

```yaml
      min: 2000-01-23T04:56:07.000+00:00
      max: 2000-01-23T04:56:07.000+00:00
    metadata: false
    vault-id:
    - vault-id
    - vault-id
    measurement-channel-id:
    - measurement-channel-id
    - measurement-channel-id
key:
  type: object
  properties:
    vault-access-key:
      type: string
      format: uuid
      description: Access key for Cloud Storage Vault
config:
  type: object
  properties:
    handler-url:
      type: string
      format: uuid
      description: URL for push notifications
    level:
      type: string
      description: Defines the level of the notification ('id-only', 'metadata' or 'full')
Query_submittime:
  properties:
    min:
      type: string
      format: date-time
      description: Earliest Data Package submission time
    max:
      type: string
      format: date-time
      description: Latest Data Package submission time
  description: Data Package submission time
  example:
    min: 2000-01-23T04:56:07.000+00:00
    max: 2000-01-23T04:56:07.000+00:00
inline_response_200:
  type: object
  properties:
    full-name:
      type: string
    vault-id:
      type: string
      description: ID of the user's Cloud Storage Vault.
inline_response_200_1:
  type: object
  properties:
    full-name:
      type: string
```

```
    vault-id:
      type: string
      format: uuid
      description: ID of the user's Cloud Storage Vault.
    type:
      type: string
      description: '`read` or `write` access key'
    in-use:
      type: boolean
      description: '`true` when key is already in use, otherwise `false`'
inline_response_200_2:
  type: object
  properties:
    name:
      type: string
    type:
      type: string
```

3.2 CIDM v1.2.1 jsonSchema

```
{
    "$schema": "http://json-schema.org/draft-04/schema#",
    "description": "Common Industrial Data Model",
    "type": "object",
    "properties": {
        "Signal": {
            "$ref": "#/definitions/Signal"
        },
        "MeasurementChannel": {
            "$ref": "#/definitions/MeasurementChannel"
        },
        "DataPackage": {
            "$ref": "#/definitions/DataPackage"
        }
    },
    "additionalProperties": false,
    "definitions": {
        "TimeSeriesMeasurementChannel": {
            "title": "TimeSeriesChannel",
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "time-series"
                    ]
                },
                "capture-interval": {
                    "type": "number"
                },
                "on-change": {
                    "type": "boolean"
                },
                "sample-strategy": {
                    "type": "string",
                    "enum": [
                        "min",
                        "max",
                        "average",
                        "last-known-value"
                    ]
                },
                "signal": {
                    "$ref": "#/definitions/Signal"
                },
                "format": {
                    "type": "string"
                },
                "dimension": {
                    "type": "number"
                }
```

```
            },
            "required": [
                "type",
                "capture-interval",
                "on-change",
                "sample-strategy",
                "signal"
            ]
        },
        "MeasurementChannel": {
            "type": "object",
            "properties": {
                "id": {
                    "type": "string"
                },
                "name": {
                    "type": "string"
                },
                "type": {
                    "type": "string",
                    "enum": [
                        "time-series",
                        "histogram",
                        "geo-histogram",
                        "general-purpose",
                        "event-based",
                        "basic-cpp-information"
                    ]
                },
                "comment": {
                    "type": "string"
                }
            },
            "required": [
                "id",
                "name",
                "type"
            ],
            "oneOf": [
                {
                    "$ref": "#/definitions/TimeSeriesMeasurementChannel"
                },
                {
                    "$ref": "#/definitions/HistogramMeasurementChannel"
                },
                {
                    "$ref": "#/definitions/GeoBasedHistogramMeasurementChan-
nel"
                },
                {
                    "$ref": "#/definitions/GeneralPurposeMeasurementChannel"
                },
```

```
                {
                    "$ref": "#/definitions/BasicCppInformationMeasurement-
Channel"
                },
                {
                    "$ref": "#/definitions/EventBasedMeasurementChannel"
                }
            ]
        },
        "HistogramMeasurementChannel": {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "histogram",
                        "geo-histogram"
                    ]
                },
                "aggregation-strategy": {
                    "type": "string",
                    "enum": [
                        "time",
                        "count",
                        "min",
                        "max"
                    ]
                },
                "capture-interval": {
                    "type": "number"
                },
                "dimensions": {
                    "type": "integer",
                    "minimum": 1
                },
                "bins": {
                    "type": "array",
                    "minItems": 1,
                    "items": {
                        "type": "object",
                        "properties": {
                            "type": {
                                "type": "string",
                                "enum": [
                                    "linear",
                                    "logarithmic",
                                    "custom"
                                ]
                            },
                            "lower-bound": {
                                "type": "number"
                            },
```

```
    "upper-bound": {
        "type": "number"
    },
    "signal": {
        "$ref": "#/definitions/Signal"
    },
    "number-of-bins": {
        "type": "integer",
        "minimum": 0
    },
    "alternative-bin-labels": {
        "type": "array",
        "items": {
            "type": "string"
        }
    }
},
"required": [
    "type",
    "lower-bound",
    "upper-bound",
    "signal",
    "number-of-bins"
],
"oneOf": [
    {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "linear",
                    "logarithmic"
                ]
            }
        },
        "required": [
            "type"
        ]
    },
    {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "custom"
                ]
            },
            "custom-bounds": {
                "type": "array",
                "items": {
```

```
                                    "type": "number"
                                }
                            }
                        },
                        "required": [
                            "type",
                            "custom-bounds"
                        ]
                    }
                ]
            }
        }
    },
    "required": [
        "type",
        "aggregation-strategy",
        "capture-interval",
        "dimensions",
        "bins"
    ],
    "additionalProperties": false
},
"GeoBasedHistogramMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
                "geo-histogram"
            ]
        },
        "geo-resolution": {
            "type": "number"
        }
    },
    "required": [
        "type",
        "geo-resolution"
    ],
    "additionalProperties": false,
    "allOf": [
        {
            "$ref": "#/definitions/HistogramMeasurementChannel"
        }
    ]
},
"GeneralPurposeMeasurementChannel": {
    "type": "object",
    "properties": {
        "type": {
            "type": "string",
            "enum": [
```

```
                    "general-purpose"
                ]
            },
            "signal": {
                "$ref": "#/definitions/Signal"
            }
        },
        "required": [
            "type",
            "signal"
        ],
        "additionalProperties": false
    },
    "BasicCppInformationMeasurementChannel": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "basic-cpp-information"
                ]
            },
            "signal": {
                "$ref": "#/definitions/Signal"
            }
        },
        "required": [
            "type",
            "signal"
        ],
        "additionalProperties": false
    },
    "EventBasedMeasurementChannel": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "event-based"
                ]
            },
            "format": {
                "type": "string"
            },
            "event-sample-strategy": {
                "type": "string",
                "enum": [
                    "real-time-event",
                    "trigger-event",
                    "threshold-event"
                ]
            },
```

```json
                "comment": {
                    "type": "string"
                }
            },
            "required": [
                "type",
                "signal",
                "event-sample-strategy"
            ],
            "additionalProperties": false
        },
        "DataPackage": {
            "type": "object",
            "properties": {
                "datapackage-id": {
                    "type": "string"
                },
                "vault-id": {
                    "type": "string"
                },
                "trip-id": {
                    "type": "string"
                },
                "cpp-id": {
                    "type": "string"
                },
                "cvim-version": {
                    "type": "string",
                    "enum": [
                        "1.0.0",
                        "1.0.1",
                        "1.2.0",
                        "1.2.1"
                    ]
                },
                "cpp-type": {
                    "type": "string",
                    "enum": [
                        "vehicle",
                        "building"
                    ]
                },
                "type": {
                    "type": "string",
                    "enum": [
                        "time-series",
                        "histogram",
                        "geo-histogram",
                        "general-purpose",
                        "event-based",
                        "basic-cpp-information"
                    ]
```

```
},
"measurement-channel-id": {
    "type": "string"
},
"submit-time": {
    "type": "string",
    "format": "date-time"
},
"mileage-start": {
    "type": "number"
},
"room-id": {
    "type": "string"
},
"mileage-stop": {
    "type": "number"
},
"geo-bounding-box": {
    "type": "object",
    "properties": {
        "latitude-min": {
            "type": "number"
        },
        "latitude-max": {
            "type": "number"
        },
        "longitude-min": {
            "type": "number"
        },
        "longitude-max": {
            "type": "number"
        },
        "altitude-min": {
            "type": "number"
        },
        "altitude-max": {
            "type": "number"
        }
    },
    "additionalProperties": false
},
"location": {
    "type": "object",
    "properties": {
        "latitude": {
            "type": "number"
        },
        "longitude": {
            "type": "number"
        }
    },
    "additionalProperties": false
```

```
},
"oem-certification": {
    "type": "object",
    "properties": {
        "signature": {},
        "checksum": {},
        "sequence-number": {}
    },
    "additionalProperties": false
},
"expiration-date": {
    "type": "string",
    "format": "date-time"
},
"data-ownership-information": {
    "type": "object",
    "properties": {
        "privacy-veto-rights": {
            "type": "object",
            "properties": {
                "consent-level": {
                    "type": "string",
                    "enum": [
                        "public",
                        "shared",
                        "private"
                    ]
                },
                "data-format": {
                    "type": "string",
                    "enum": [
                        "time-series",
                        "histogram"
                    ]
                },
                "jurisdiction": {
                    "type": "string",
                    "enum": [
                        "Europe",
                        "any"
                    ]
                },
                "storage-constraint": {
                    "type": "string",
                    "enum": [
                        "OEM storage",
                        "Personal storage"
                    ]
                }
            },
            "required": [
                "consent-level"
```

```
        ]
    },
    "copyright-stakeholders": {
        "type": "array",
        "items": [
            {
                "type": "object",
                "properties": {
                    "name": {
                        "type": "string"
                    },
                    "status": {
                        "type": "string"
                    }
                },
                "required": [
                    "name",
                    "status"
                ],
                "additionalProperties": false
            }
        ]
    },
    "data-stakeholders": {
        "type": "array",
        "items": [
            {
                "type": "object",
                "properties": {
                    "name": {
                        "type": "string"
                    },
                    "status": {
                        "type": "string"
                    }
                },
                "required": [
                    "name",
                    "status"
                ],
                "additionalProperties": false
            }
        ]
    },
    "data-privacy-level": {
        "type": "string",
        "enum": [
            "public",
            "shared",
            "private"
        ]
    }
```

```
            },
            "additionalProperties": false
        },
        "data-masking-active": {
            "type": "boolean"
        },
        "signatures": {
            "type": "array",
            "items": [
                {
                    "type": "object",
                    "properties": {
                        "signatory": {
                            "type": "string"
                        },
                        "checksum": {
                            "type": "string"
                        },
                        "signature": {
                            "type": "string"
                        }
                    },
                    "required": [
                        "signatory",
                        "checksum",
                        "signature"
                    ],
                    "additionalProperties": false
                }
            ]
        }
    },
    "required": [
        "cvim-version",
        "type",
        "measurement-channel-id",
        "vault-id",
        "cpp-type"
    ],
    "oneOf": [
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "time-series"
                    ]
                },
                "number-of-samples": {
                    "type": "integer",
                    "minimum": 1
```

```json
},
"statistic-properties": {
    "type": "object",
    "properties": {
        "min": {
            "type": "number"
        },
        "max": {
            "type": "number"
        },
        "average": {
            "type": "number"
        },
        "histogram": {
            "type": "object",
            "properties": {
                "measurement-channel-id": {
                    "type": "string"
                },
                "data": {
                    "type": "array",
                    "minItems": 0,
                    "maxItems": 10,
                    "items": {
                        "type": "number"
                    }
                }
            },
            "required": [
                "measurement-channel-id",
                "data"
            ]
        }
    }
},
"data": {
    "type": "array",
    "minItems": 1,
    "items": {
        "type": "object",
        "properties": {
            "timestamp": {
                "type": "string",
                "format": "date-time"
            },
            "value": {
                "type": [
                    "array",
                    "string",
                    "number",
                    "boolean"
                ],
```

```
                    "items": {
                        "type": [
                            "string",
                            "number",
                            "boolean"
                        ],
                        "minLength": 1
                    }
                }
            }
        }
    },
    "timestamp-start": {
        "type": "string",
        "format": "date-time"
    },
    "timestamp-stop": {
        "type": "string",
        "format": "date-time"
    }
},
"required": [
    "type",
    "number-of-samples",
    "data",
    "timestamp-start",
    "timestamp-stop"
]
},
{
"type": "object",
"properties": {
    "type": {
        "type": "string",
        "enum": [
            "histogram",
            "geo-histogram"
        ]
    },
    "data": {
        "type": "array",
        "items": {
            "type": [
                "number",
                "array"
            ],
            "minItems": 1,
            "items": {
                "type": "number"
            }
        }
    },
```

```
                "timestamp-start": {
                    "type": "string",
                    "format": "date-time"
                },
                "timestamp-stop": {
                    "type": "string",
                    "format": "date-time"
                }
            },
            "required": [
                "type",
                "data",
                "timestamp-start",
                "timestamp-stop"
            ]
        },
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "general-purpose",
                        "basic-cpp-information"
                    ]
                },
                "data": {},
                "timestamp": {
                    "type": "string",
                    "format": "date-time"
                }
            },
            "required": [
                "type",
                "data",
                "timestamp"
            ]
        },
        {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "event-based"
                    ]
                },
                "data": {
                    "type": "object",
                    "properties": {
                        "event-type": {
                            "type": "string",
```

```
                                                "enum": [
                                                    "real-time-event",
                                                    "trigger-event",
                                                    "threshold-event"
                                                ]
                                            },
                                            "event-data": {
                                                "type": "object",
                                                "properties": {
                                                    "value": {
                                                        "type": "string"
                                                    },
                                                    "event-datapackages": {
                                                        "type": "array",
                                                        "minItems": 1,
                                                        "items": {
                                                            "$ref": "#/defini-
tions/DataPackage"
                                                        }
                                                    }
                                                },
                                                "required": [
                                                    "value"
                                                ]
                                            }
                                        },
                                        "required": [
                                            "event-type",
                                            "event-data"
                                        ]
                                    },
                                    "timestamp": {
                                        "type": "string",
                                        "format": "date-time"
                                    }
                                },
                                "required": [
                                    "type",
                                    "data",
                                    "timestamp"
                                ]
                            }
                        ]
                    },
                    "Signal": {
                        "title": "CIDM Signal",
                        "description": "Signals are the perception organs of vehi-
cles. It is their main duty to detect physical phenomenons and chemical quan-
tities by transferring them into electrical signals. They observe the envi-
ronment and gener-ate the data that is exchangeable at AutoMat’s market-
place. They are one of the core components of the AutoMat project. Fig-
```

```
ure 11 shows the UML modelling of the signals. Within AutoMat all infor-
mation pro-viders are modelled as Signal. They can be classi-
fied as static signals or changing/non-static signals, having a sam-
ple rate larger than zero. ",
            "type": "object",
            "properties": {
                "id": {
                    "type": "string"
                },
                "name": {
                    "type": "string"
                },
                "cpp-type": {
                    "type": "string",
                    "enum": [
                        "vehicle",
                        "building"
                    ]
                },
                "type": {
                    "type": "string",
                    "enum": [
                        "numeric",
                        "information",
                        "enumeration",
                        "general-purpose"
                    ]
                },
                "format": {
                    "type": "string"
                },
                "sample-rate": {
                    "type": "number",
                    "minimum": 0
                },
                "comment": {
                    "type": "string"
                }
            },
            "required": [
                "id",
                "name",
                "type",
                "sample-rate",
                "cpp-type"
            ],
            "oneOf": [
                {
                    "$ref": "#/definitions/NumericSignal"
                },
                {
                    "$ref": "#/definitions/EnumerationSignal"
```

```
            },
            {
                "$ref": "#/definitions/InformationSignal"
            },
            {
                "$ref": "#/definitions/GeneralPurposeSignal"
            }
        ]
    },
    "GeneralPurposeSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "general-purpose"
                ]
            }
        },
        "required": [
            "type"
        ]
    },
    "InformationSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "information"
                ]
            },
            "format": {
                "type": "string"
            }
        },
        "required": [
            "type",
            "format"
        ]
    },
    "EnumerationSignal": {
        "type": "object",
        "properties": {
            "type": {
                "type": "string",
                "enum": [
                    "enumeration"
                ]
            },
            "items": {
                "type": "array",
```

```
                    "items": {
                        "type": "string"
                    }
                }
            },
            "required": [
                "type",
                "items"
            ]
        },
        "NumericSignal": {
            "type": "object",
            "properties": {
                "type": {
                    "type": "string",
                    "enum": [
                        "numeric"
                    ]
                },
                "format": {
                    "type": "string",
                    "enum": [
                        "int",
                        "uint",
                        "float",
                        "double"
                    ]
                },
                "min": {
                    "type": "number"
                },
                "max": {
                    "type": "number"
                },
                "resolution": {
                    "type": "number"
                },
                "unit": {
                    "type": "string"
                }
            },
            "required": [
                "type",
                "format",
                "min",
                "max",
                "resolution",
                "unit"
            ]
        }
    }
}
```

About Cross-CPP

The objective was to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research was to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project developed: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to provide enhanced local weather forecast and predict and optimise energy consumptions in households. The project has built upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, were used as a basis for Cross-CPP development extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

Funded by the Horizon 2020 Framework Programme of the European Union

10 Annex 9: Ecosystem Policy Guideline

Ecosystem for Services based on integrated Cross-sectorial Data Streams from multiple Cyber Physical Products and Open Data Sources

CROSS-CPP

CROSS-CPP Ecosystem Policy Guidelines

Cross-CPP ecosystem Internal (Legal, Privacy, Consent) Policies Guides

This project has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 780167

Table of Contents

1. Introduction

The process/activity related Methodology Guidelines describes regulations in respect to the interaction between the key Cross-CPP Ecosystem Stakeholders with the Ecosystem, such as contractual and privacy/security issues, as well as update of reference data model regulations.

1.1. Purpose

The presented methodology guidelines cover legal, privacy and consent regulations for key processes/activities in respect to the actions/roles of the various stakeholders and their interaction required for the operation of the Cross-GPP Ecosystem.

1.2. Audience

The methodology guidelines are meant for all stakeholders of the Cross-CPP Ecosystem, as well as candidates planning to be a stakeholder.

1.3. Scope

The contents of the methodology guidelines will cover regulations in respect to the interaction between the various stakeholders.

Cross-CPP team does not take responsibility, when not following the instructions given in this guide.

If you find a regulation for which there is no content in this guide you can request it through: general-support@cross-cpp.eu.

1.4. Contact

Cross-CPP Project website: https://cross-cpp.eu

Cross-CPP Marketplace: https://datagora.eu

Marketplace support: marketplace-support@cross-cpp.eu

Context Monitoring and Extraction Module (CME): context-support@cross-cpp.eu.

Policy guidelines and in general: general-support@cross-cpp.eu

2. Structure of the Guide

In reference to methodology concept outlined in D1.3[1], in the following process/activity oriented regulations are presented, comprising the actions/roles and interactions of the various stakeholders participating in the Cross-CPP Ecosystem. The key aspect driving the required regulation is the GDPR (General Data Protection Regulation)[2] approved by the European Parliament, the Council of the European Union and the European Commission. It protects people's personal data throughout the European Union (EU). The decree also affects data exports from the EU.

The General Data Protection Regulation enforces rules that protect people against a wide variety of privacy issues. It enforces the right for people to lawfully agree with companies to use their private information. It also enforces the right for people to have their private information no longer accessible by a company. It enforces that users have the right to allow their private information to become public or not. The regulation also makes sure that no personal data is processed unless the user has allowed the processor of personal data to do so.

The GDPR is aimed at giving citizens control over their personal data. Hence, as outlined in D1.3[3] in the scope of Cross-CPP the following regulations have to be declared, addressing the interaction between the various stakeholders of the Cross-CPP Ecosystem:

[1] D1.3 Public Innovation Concept

[2] European Parliament and Council of the European Union, „Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation)", Official Journal L 119 of 4.5.2016, 1, 2016

[3] There are other directives which also covers personal data transfer/protection, in particular:

Proposal for a REGULATION OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL concerning the respect for private life and the protection of personal data in electronic communications and repealing Directive 2002/58/EC (Regulation on Privacy and Electronic Communications) - COM/2017/010 final available at: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52017PC0010

- Contractual Regulations between the key stakeholders
- Privacy /Security Issues

Furthermore, the two main data models of the Cross-CPP Ecosystem the

- CIDM Standard as the brand independent format for the data exchange in the Cross-CPP Ecosystem and
- Context Model

have to be updated, because these data models have living formats which might require adaptations in case the data consumer are asking for additional signals/channels or extended context information needed for their services. Therefore, a respective regulation update is required.

The key stakeholders of the Cross-CPP Ecosystem value chain and their relationship are presented in Figure 1. In the order of the data flow these are:

- The CPP Owner represents the owner of the collected data during the operation of the CPP.
- The CPP Manufacturer is harvesting the CPP signals and after some proprietary to a non-proprietary data transformation and preprocessing activities he forwards the data to the CPP cloud storage vault in the CIDM standard format.
- The Cloud Storage Provider stores the received data in a self-contained storage vault for each CPP owner the data belong to.
- The Marketplace Operator, as the one-stop-shop access point, represents the mediator between the data of the various CPP Owners and the Service Providers (data consumers). The Marketplace will also offer a Data Analytics Toolbox and a Context Sensitivity Service providing easy to use big data analytic functionalities and context supported functionality to Service Providers.
- The Service Provider is the data consumer who, based on the data from the various CPP Owners, creates B2B and B2C services and offers these to their customers.
- Furthermore, there is a Standardization Board required to update the CIDM standard format in respect to changing market needs of data consumers. This Board consists of representatives of the CPP manufacturer and the Marketplace.

COMMISSION STAFF WORKING DOCUMENT on the free flow of data and emerging issues of the European data economy. {COM(2017) 9 final}, 10.01.2017, available at: https://ec.europa.eu/digital-single-market/en/news/staff-working-document-free-flow-data-and-emerging-issues-european-data-economy

COMMISSION STAFF WORKING DOCUMENT Guidance on sharing private sector data in the European data economy {COM(2018) 232 final} available at: https://ec.europa.eu/digital-single-market/en/news/staff-working-document-guidance-sharing-private-sector-data-european-data-economy

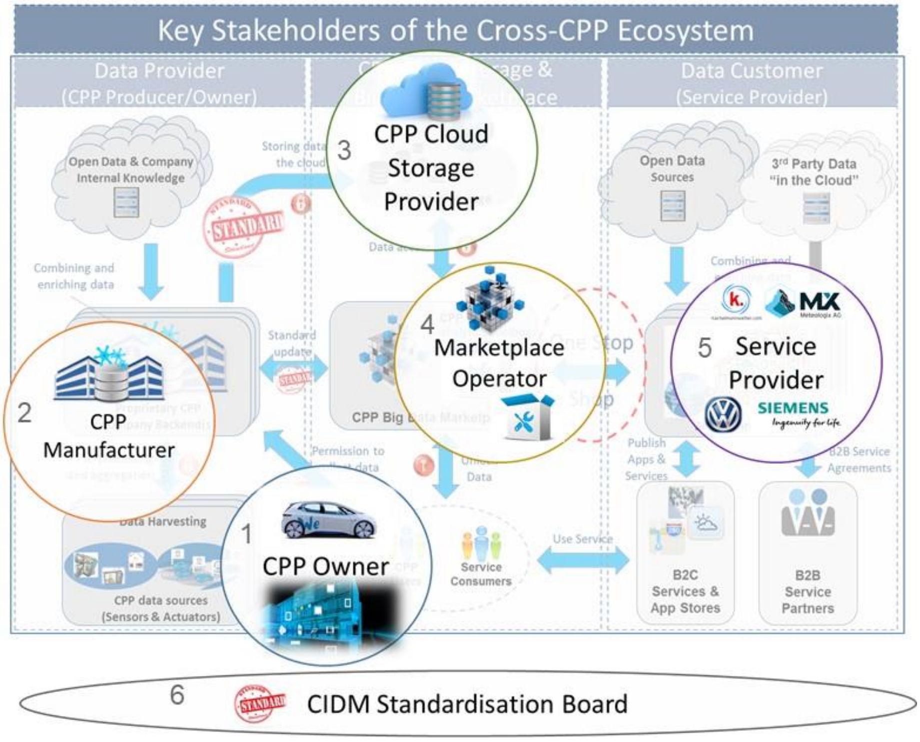


Figure 1: Key Cross-CPP Ecosystem Stakeholder

3. Contractual Regulations

As already outlined above, the GDPR is aimed at giving citizens control over their personal data. This causes the necessity to formulate contractual regulations for the interaction between the various stakeholders of the Cross-CPP Ecosystem guaranteeing the citizens control over their personal data. This requires the following contractual regulations between data owner and respective stakeholders:

- CPP data owner and CPP Manufacturer
- CPP data owner and Cloud Storage Provider
- CPP data owner and Marketplace Provider
- CPP data owner and Service Provider (data customer)
- CPP data owner and Provider of the Marketplace Services (Analytics Toolbox & Context)

The Marketplace represents the "One-Stop-Shop", the single point of entry for a brand-independent access for Service Providers to retrieve data streams from multiple cyber-physical mass products. The Marketplace is the meditator for all the actors involved in the flow of the data in the Cross-CPP Ecosystem. Thereby, the Marketplace knows which stakeholders are participating in the Cross-CPP Ecosystem, which data from which data owner are harvested by which CPP Manufacturer and used by which Data Consumer. In this respect, the Marketplace, in turn, will maintain the mapping between CPP Owners (e.g. CPP_ID, which signals/channels), CPP Manufacturer (e.g. ID, data forwarded storage vaults), Storage Providers (e.g. ID, CPP Owner ID, data available) and Data Consumer: (e.g. registration, data request, data Owner access right).

This central stakeholder interaction management by the Marketplace represents the basic concept of the Cross-CPP Ecosystem in order to carry out the access control of the CPP Owner's data by the various data consumers at run-time, as well as being able to answer accurately future requests by any Service Provider.

3.1. Data Harvesting Agreement

In the following section the regulations are presented to guarantee that the CPP Owner has full control of the data harvested from his CPP. For this, the CPP owner has to give the permission to the CPP Manufacturer to collect the available CPP data and to pre-process them. The arrangement of this contractual agreement must be driven by the CPP Manufacturer, because only he knows which signals of the individual CPP are available to be forwarded to the outside world. Furthermore, the CPP Owner has to agree that the CPP Manufacturer forwards his data to a cloud storage vault (location to be decided at agreement point).

Action/Topic	**Regulation**
Regulation topic	This regulation applies to the process that the CPP Manufacturer gets the CPP owner's permission to gather, pre-process and forward data to his storage vault in the cloud.
Stakeholders Affected	The stakeholders addressed by the data harvesting related regulations are: • CPP Owner • CPP Manufacturer • Big Data Marketplace Provider • Storage Provider
Registration	A CPP owner has to register as a data provider at the Marketplace and the CPP Manufacturer has to be informed. A CPP Manufacturer who intends to participate in the Cross-CPP Ecosystem to offer data of his CPP Owners to the outside world has to register as a partner at the Marketplace, too.
Access Rights Initialization	A registered CPP Owner needs to give the permission to the CPP Manufacturer to collect which signals/channels under which constraints and to forward these data to be stored in the CPP Owner's Cloud Storage vault. The CPP Manufacturer has to configure the data logger of the respective CPP to harvest the agreed signals/data, adapt the respective preprocessing and to be prepared to forward the CPP data in the CPP-Owner's storage vault in the CDIM format. The contract information has to be forwarded to the Marketplace for the CPP Owner contract registration and management, covering information about CPP Owner, CPP device and harvested CPP data.

Action/Topic	Regulation
Access Rights update	If the CPP owner decides to modify the harvested signals or constraints the access right contract with the CPP Manufacturer must be updated. A respective logger adaptation required has to be initiated by the CPP Manufacturer and the Marketplace informed about the modifications.
Access Rights Termination	In the case that a CPP Owner does not want to provide data any more (for instance, because the vehicle was sold), he can revoke the CPP Manufacturer's data gathering authorization. In order to stop data gathering, based on the CPP Owner's request, the CPP Manufacturer has to reconfigure the involved CPP data harvesting configuration so that no data are gathered any more. This will trigger in the Marketplace the process of removing the Data Owner from any active data delivery contracts to Service Providers in which he is involved. The Storage Provider has to be informed about this termination, as precondition to delete the data Owner's historical data in his storage area.
Implementation examples/alternatives	The implementations for the data harvesting and the communication with the CPP Manufacturer Backend, as well as any pre-processing activities are entirely proprietary solutions of the CPP Manufacturer. To set up the permission for the CPP Manufacturer to collect which signals/channels under which constraints a wide spectrum of solution alternatives might be possible such as: • Out of a provided list of accredited signals and constraints the CPP Owner directly selects the data allowed to be harvested. • Marketplace supports the selection process • The CPP Manufacturer supports the selection process (e.g. for vehicles the car dealer might supports the CPP owner to set up the harvesting permission during the sales arrangement). There is a huge demand for innovative solutions for future extension of the solution.

3.2. Booking of CPP Owner's Storage Vault

In the following the regulations are presented to guarantee that the CPP Owner has full control of the data stored from his CPP. For this, the CPP Owner has to agree with an accredited Cloud Storage Provider to store his CPP data forwarded by the CPP Manufacturer in a CPP Owner's storage vault. Furthermore, the Marketplace has to be informed about this agreement.

Action/topic	**Regulation**
Regulation topic	This regulation applies to the process that the CPP Owner rents a cloud storage vault for his CPP data forwarded by the CPP Manufacturer.
Stakeholders affected	The stakeholders addressed by the storage related regulations are: • CPP Owner • Storage Provider • CPP Manufacturer • Marketplace
Storage Accreditation	A Cloud Storage Provider targeting to offer data storage vaults in the scope of the CPP Ecosystem has to fulfill a set of mandatory requirements (e.g. interface standard, data transmission security, data access rules etc.). The compliance with the Cross-CPP accreditation criteria has to be controlled by the Marketplace. If the Storage Provider owns such an accreditation he will become part of the Marketplace catalogue and can offer storage areas to any CPP Owner.
Storage Rights Initialization	The CPP Owner chooses a cloud storage vault for his CPP specific data. For this purpose the CPP Owner enters a contract with a Cloud Storage Provider for creating his own cloud storage area. This cloud storage vault represents the private data storage for the CPP Owner, where his data forwarded by the CPP Manufacturer will be stored. The CPP Owner's cloud storage vault has to be registered in the Marketplace. In reference to this contract, the CPP Manufacturer has to be informed to which storage location the data of the CPP owner have to be forwarded.
Storage Rights Termination	The CPP Owner can at any time resign to store his data in his cloud storage vault. The termination process is also triggered when the owner intends to revoke his data gathering agreement with the CPP Manufacturer (e.g. car is sold). In this case the Storage Provider has to delete the data Owner's historical data in his storage area. The Storage Provider has to inform the Marketplace that the storage vault is not accessible anymore. This triggers the update of the Marketplace's list of active data vaults and CPP Owners. Furthermore, the CPP Manufacturer has to be informed to stop the data transfer activity.

Action/topic	**Regulation**
Implementation examples/ alternatives	Any institution with a respective storage infrastructure may apply for an accredited Storage Provider such as the CPP Manufacturers, Banks etc. There is also a wide spectrum of alternatives to support CPP Owner in selecting a storage provider, such as: • Out of a list of accredited storage provider the CPP Owner directly contacts a storage provider • Marketplace supports the storage provider selection process • For vehicles e.g. the dealer of the CPP Manufacturer might supports the CPP owner storage selection together with the harvesting permission agreement in the scope of the conclusion of the sale arrangement. • etc. There is a huge demand for innovative solutions for future extension of the solution

3.3. Data Access Agreement Service Provider

In reference to interaction between CPP Owner and Service Provider, regulations are in place to guarantee that the CPP Owner has full control of the usage of the data from his CPP by any Service Provider. Therefore, for the data the Service Provider would like to have access to, the CPP owner has to give the permission to the Service Provider to use his data for a specific service offered to customers of the Service Provider. Thereby, the Marketplace supports the Service Providers in the form of an easy access and detection of needed data (refer to the Service Provider user guideline for more information). The Marketplace is also involved in the enforcement of the access permission between the CPP Owner and the various Service Providers, an essential pre-requisite for the data access control of the Marketplace at run-time.

Action/Topic	**Regulation**
Regulation topic	This regulation applies to the process of using CPP Owner data by Service Providers to generate their services.
Stakeholders affected	The stakeholders addressed by the data access related regulations are: • CPP owner • Service Provider • Big Data Marketplace
Registration	Any Service Provider who intends to have access to the data of the CPP Ecosystem has to register as a Data Consumer at the Marketplace. To overcome the certification process some requirements are mandatory such as compliance with data protection regulations etc.

Action/Topic	**Regulation**
Order Release	Based on the information provided by the Marketplace (data catalogue, test data, statistics) the Service Provider has to specify a request (an offer) covering the data he wants to have access to for his services. In reference to the data request, the Marketplace has to check the availability of relevant data sources. The Service Provider has to be informed with regard to the available data sources. In case the Service Provider is satisfied with the number of data sources and their characteristics, he will proceed to place an offer for the selected data.
Access Right Initialization	The Marketplace represents the central point to manage the CPP Owner permissions for the data access by the Service Providers. Generally, the Marketplace forwards the Service Provider offer to the CPP Owners for which data was selected. This approach has to be followed for B2B type services (e.g. local weather information for farmers, generation of route roughness information for civil services etc.) where the Service Providers have no direct contact with the CPP Owner. This B2B type services mostly use anonymized data from a large amount of CPP Owners. As already described above the Marketplace has to identify the respective data owner and forward the data request. If the CPP Owner gives the access permission (accepting the offer), this contract represents the reference for the access control at run-time by the Marketplace. In the B2C type business, where the Service Provider may have direct contact with the CPP Owner an alternative approach feasible. In this case the Service Provider can directly contact the CPP Owner to arrange an access agreement. For instance, when booking the service the CPP Owner directly agrees that the Service Provider has access to his data. In this case, the Marketplace has to be informed about this agreement.
Access Right Update	In case of required updates of data requests by Service Providers the Marketplace has to manage the adaption of the modified data access permissions of the Data Owner. Thus, also the access control at run-time has to be modified.
Access Rights Termination	In order to stop data access, either from CPP Owners side or from Service Provider side the Marketplace has to delete the data access agreement for this CPP Owner/Service Provider relation.
Implementation examples/ alternatives	Even though the access agreement deals with a one-to-one relation between CPP Owner and Service Provider simple time and cost efficient contract arrangement solution are needed (e.g. sending email or newsletters or via webpage or dashboard where CPP Owners can

Action/Topic	**Regulation**
	search for new offers). There is a huge demand for innovative solutions for future extension of the solution.

3.4. Data Access Agreement Marketplace Services

The Marketplace also offers a Data Analytics Toolbox and a Context Sensitive solution. The toolbox services provide easy to use big data analytic functionalities to support Service Providers with low big data expertise and knowledge. The context sensitive solution enables the use the CPP data on the context under which the data is generated to improve user experience and filter CPP data for provision according to the Service Provider need for the services. As integral part of the Marketplace, these Marketplace Services are offered and can be booked via the Marketplace by any member of the Service Provider community of the Cross-CPP Ecosystem.

As any other Service Providers, also the Toolbox and Context Sensitive solution Providers have to ask the CPP Owner for the permission to use his data for these services. Therefore, also the Marketplace Services have to arrange a contractual agreement with the CPP Owner to guarantee that he has full control over the use of his data.

Action/Topic	**Regulation**
Regulation topic	This regulation applies to the process of using CPP Owner data by the Marketplace Services (Data Analytic Toolbox and Context Sensitive solution).
Stakeholders affected	The stakeholders addressed by the data access related regulations are: • CPP owner • Provider of the Marketplace Services • Marketplace Provider
Registration of Marketplace Services	Any Toolbox or Context Sensitive solution Provider who wants to offer and operate his service via the Marketplace has to make an agreement with the Marketplace Provider to install the required data access for the respective Marketplace Service, as well as to integrate the service into the user interface and the user guidelines as required for the usage by any external Service Provider.

Action/Topic	**Regulation**
Access Rights Initialization	For each of the data requests (offers) that want to use the Toolbox and Context Sensitive solution, the Marketplace identifies the corresponding CPP Owners. These owners, will accept (or not) the offer in question (creating the data access agreement), not only for the use of the CPP data but also for the for the access to the CPP data by the Marketplace services at run-time.
Access Rights Update	In case the data requirement of the Marketplace Services change the Marketplace has to manage the adaption of the modified data access permissions of the CPP Owner and also to modify the access control at run-time.
Access Rights Termination	In order to stop data access, either from CPP Owners side or from Marketplace Service Provider side the Marketplace has to delete the data access agreement for this CPP Owner/Marketplace Service Provider relation.
Implementation examples/ alternatives	Just as the arrangement of the access agreement between CPP Owners and Service Providers, also for the agreement between CPP Owners and Marketplace Service Providers simple time and cost efficient contract arrangement solution are needed. There is a huge demand for innovative solutions for future extension of the solution.

3.5. Marketplace Authorization to Broker Owner's Storage Data.

The following regulations guarantee that the Data Owner has full control that a Marketplace Provider is authorized to broker his data. For this, the Data Owner will authorize a Marketplace Provider to access the data in the Data Owner's storage area and to forward them to specific authorized service providers, having a contractual arrangement with the Data Owner to use his data.

The Storage Provider has to be informed which Marketplace Provider has access to the data of a Data Owner. Furthermore, in the scope of the contractual agreement between Service Provider and Data customer regarding the data access (see 6.1.3), the Service Provider has to be informed via which Marketplace Provider he has access to the desired data.

Action/Role	**Regulation**
Regulation topic	This regulation applies to the process to authorize a Marketplace Provider to access the data in the Data Owner's storage area and to forward them to specific authorized Service Providers

Action/Role	**Regulation**
Stakeholders affected	The stakeholders addressed by the data access related regulations are: • Data Owner • Marketplace Provider • Storage Provider • CPP Service Provider
Marketplace Offer	Any Marketplace Provider who intends to participate in the Cross-CPP Ecosystem has to offer his brokering services to Data Owners. A Marketplace Provider targeting to offer data brokering services in the scope of the CPP Ecosystem has to fulfill a set of mandatory requirements (e.g. interface standard, data transmission security, data access rules etc.).
Cally. *Marketplace Rights Initialization*	The CPP Owner authorizes a Marketplace Provider to broker his data. For this purpose the CPP Owner enters a contract with a Marketplace Provider that allows the respective Marketplace to have access to the data in the Data Owner's specific data storage area at the respective Storage Provider. Furthermore, the Marketplace Provider will be authorized to forward these data to specific, by the Data Owner authorized Service Providers. In reference to this contract, the Storage Provider has to be informed which Marketplace Provider is allowed to have access to the data of a Data Owner. Furthermore, the Service Provider has to be informed which Marketplace Provider he has to address to get access to the desired data of a specific Data Owner.
Marketplace Rights Termination	The CPP Owner can at any time resign the authorization of a Marketplace Provider to broker his data. The termination process is also triggered when the Data Owner resigns his cloud storage vault with the Storage Provider. In this case the Storage Provider has to delete the data access rights of the respective Marketplace. Also the Service Provider has to be informed that the specific Owner's data cannot be accessed via this Marketplace anymore.
Implementation examples/ alternatives	Any institution with a respective Marketplace infrastructure and services fulfilling a set of mandatory requirements compatible with the Cross-CPP Ecosystem may offer his services to any Data Owner.

4. Privacy/Security Issues

The following the Cross-CPP regulations address privacy/security issues with special reference to the GDPR. The Cross-CPP ICT solution shall allow to adapt the privacy, security ethical rules in the cross sectorial services to the current user's context, based on identified conditions (context) under which the CPP data is being generated.

Action/topic	**Regulation**
Regulation topic	This regulation applies to the to privacy/security issues of the Cross-CPP Ecosystem
Stakeholders affected	The stakeholders addressed by privacy/security issues: • Marketplace • CPP owner • CPP Manufacturer • Storage Provider • Service Provider • Provider of the Marketplace Services
Reference	General Data Protection Regulation (GDPR) approved by the European Parliament, the Council of the European Union and the European Commission, covering existing laws and regulations governing personal data protection and privacy in the European Union.
Targets	The stated target aims that the CPP Owner can choose whether to share own data with third parties for their commercial purposes based on a contract. Moreover, the data owner must be able to deactivate specific data accesses etc. All these principles require that the data owner always must be aware about 'who is using which data for what purpose', even for the usage of anonymized data by a Service Provider. Furthermore, a secure data communication between the various stakeholders of the Cross-CPP Ecosystem has to be guaranteed.
Privacy by Design	'Privacy by design' and 'privacy by default' has to become an essential principles in EU data protection rules. This means that data protection safeguards has to be built into products and services from the earliest stage of development and that privacy-friendly default settings should be the norm. The position of Privacy officer should be installed to ensure that regulatory requirements are fulfilled in each one of the components of the Cross-CPP Ecosystem.

Action/topic	**Regulation**
Personalized /Anonymization Data	In general, the Cross-CPP Ecosystem supports the provision of CPP Owner data for services based on personalized or anonymized data. In the case of individual services based on personalized data, the access agreement has to allow the Service Provider to retrieve personal details of the CPP Owner. This applies for a wide range of B2C type services based on personalized data (e.g. insurance case). Another large group of services is based on anonymized CPP Owner data from a large amount of CPPs. This comprises B2B type services (e.g. weather forecast to farmers) as well as B2C type services (e.g. road fog information for the vehicle drivers). Thereby, in reference to the data flow in the Cross-CPP Ecosystem the anonymization should happen as early as possible. However, to guarantee to the CPP Owner full control of the usage of his data in case of the Cross-CPP Ecosystem the anonymization can be first realised at the Marketplace before forwarding the CPP data to Service Providers.
Data Security	For the secure data exchange between the various stakeholders of the Cross-CPP Ecosystem encryption/decryption technologies have to be applied. The data storage has to ensure the trustworthiness of the data source, protect the privacy of the data owners, prevent data from unauthorized manipulation and ensure data quality. The Manufacturer Backends have to mask CPP data following the privacy rules that are defined by each CPP Owner. The Marketplace have to ensure that Service Providers' requests do not break the desire of the CPP Owner. The Marketplace have to check the integrity and completeness of the data to avoid providing damaged or corrupted data to Service Providers. Device data requests may contain a field to denote that personal information is not required. In these cases, the Marketplace only forward anonymous data.
Implementation examples/ alternatives	For the encryption/decryption activities for the data exchange between the various stakeholders of the CPP Ecosystem there are various commercial tools available, e.g Bdrive, tresorit.

5. CIDM Update Regulations

The Common Industrial Data Model (CIDM) Standard represents a living format which might require adaptations in case the data consumers are asking for additional signals/channels needed for their services. In the following the regulations and the involved stakeholders are described for the CIDM update procedure.

Action/Topic	**Regulation**
Regulation topic	This regulation applies to the processes and activities addressing the update of the CIDM Standard
Stakeholders affected	The stakeholders involved in the CIDM update process are representatives of: • Big Data Marketplace • CPP Manufacturer • CPP Standardization Board Thereby, the Marketplace takes the role as central point for the CIDM management.
CPP Standardization Board	The CPP Standardization Board has to be constituted by representatives of the participating CPP Manufacturers, chaired by a representative of the Marketplace.
Collection of Update Requests	The Marketplace has to collect CIDM update requests forwarded by Service Providers asking for data not yet provided by CPPs. This requires an extension of the actual version of the CIDM by additional signals/channels. The collected update requests have to be forwarded to the Standardization Board. Tailored to suit the market needs a date for a Standardization Board meeting has to be agreed to decide on potential CIDM updates.
CIDM Update Decision	In respect to market needs and business aspects the Standardization Board has to decide which of the Service Provider requests will be accepted and an updated version of the CIDM will be agreed. The Board has to define regulations for the permission of the CPP Owners in respect to the extended CDIM data harvesting and usage. A CIDM update is accepted if 2/3 of the CPP members vote for. However, none of the CPP Manufacturer is explicitly forced to provide the additional signals/channels.
CIDM Update Stakeholder Actions	The marketplace has to forward the updated CIDM to all affected stakeholders of the Cross-CPP Ecosystem, such as: • CPP Manufacturers

Action/Topic	**Regulation**
	• Storage Providers • Service Providers • Provider of the Marketplace Services The Marketplace has to update the catalogue presented to the data consumer. If the CPP Owner agrees, the CPP-Manufacturer Backend has to update their data logging, enabling the harvesting of the new signal/channels to be forwarded to the storage area. The Storage Provider has to update the internal CIDM data representation of the data owner's storage area in respect to the agreed access rights between CPP Owner and CPP Manufacturer to guarantee that the new CIDM data structure is stored properly.

6. Context Model Update Regulations

The Context Model represents a living format which might require adaptations in case the data consumer are asking for additional filters for the CPP data. In the following the regulations and the involved stakeholders are described for the Context Model update process.

The definition of the context (context modelling) allows for time and cost effective extensions with additional data sources, rules and CPP from various sectors

The context Models have to be specific for various CPP and sectors. However, in order to gain efficiency in model definition, generic context models are defined which that can be adapted to the specific CPP/sector. Therefore, the context model includes generic, sector and CPP specific concepts so that it will be extensible for different sectors and industrial areas.

The context Models have to be specific for various CPP and sectors.

Action/Role	**Regulation**
Regulation topic	This regulation applies to the processes and activities addressing the update of the Context Model.
Stakeholders affected	The stakeholders involved in the Context Model update process are representatives of: • Marketplace Provider • Context Model Provider

Action/Role	**Regulation**
Collection of Update Requests	The Marketplace collects context related filters update requests forwarded by the Service Provider asking for filters not yet provided by the Context Sensitive solution. This requires an extension of the current version of the Context Model and/or associated reasoning rules (refer to the Marketplace developer guide for more information). The collected update requests are forwarded to the Context Sensitive solution provider.
Context model Update Decision	In respect to market needs and business aspects the Context Sensitive solution provider will decide if and when the requests will be implemented. However, the Context Sensitive solution provider is not explicitly forced to provide the additional CPP data filters.
Context model Update in the Stakeholder Action	Once the Context model and/or new rules for CPP Data filtering, the Marketplace presents the updated CPP Data filtering options to the Service Providers and future data filter is presented to the Service Providers.

7. FAQ

For any questions or inquiries about the use of the Cross-CPP Ecosystem or the contents of it or this guidelines, please forward it to: general-support@cross-cpp.eu.

Q: What methodologies are covered by this guide?

A: The presented methodology guidelines cover legal, privacy and consent regulations for key processes/activities in respect to the actions/roles of the various stakeholders and their interaction required for the operation of the Cross-CPP Ecosystem.

Q: As an interested stakeholder, which contractual regulations do I have to take into account?

A: There are methodologies for different kind of contractual regulations addressing different stakeholders of the ecosystem. More information about which regulations are affecting each stakeholder can be found in chapter 3.

Q: Do you also consider privacy and/or security issues regulations?

A: The Cross-CPP ICT solution shall allow to adapt the privacy, security ethical rules in the cross sectorial services to the current user's context, based on identified conditions (context) under which the CPP data is being generated. More details about this are in chapter 4.

8. Glossary

Administrator: Cross-CPP marketplace system administrator

AEON: AEON application

AEON application: publication/subscription based communication application

AEON channel: set configuration for communication between two actors through AEON application

Cross-CPP: System

Analytics Toolbox: set of available analytics functions to be requested by the Data Consumer

Channel: sampler of the data the signals process

CIDM: Common Industrial Data Model

CIDM model: standardized data model for industrial data-driven services

Contract: entity that resumes the acceptance of a data request from a data owner

CME: Context Monitoring and Extraction module

CPP: cyber-physical product

CPP Data: data created by a CCP and sent to the system by the Data Provider

CPP Owner: Data Owner which CPP is registered in the Cross-CPP data-marketplace

Data Consumer: actor who receives the data created by owners to use it on the creation or improvement of services

Data Consumer Wallet: group of MP functionalities for Data Consumers

Data Owner: owner of the CPP that sends data to the system

Data Provider: OEM that provides its users data to the Cross-CPP marketplace

Data Request: set of configurations that define a scope for CPP Data to be received by a Data Consumer

Marketplace: Marketplace Web Application

My Data Wallet: group of MP functionalities for Data Owners

MP: Marketplace

OEM: Original Equipment Manufacturer

Offer: published and available Data Request

Provider: Data Provider

Service Provider: Data Consumer

Signal: information provider of the data the CPP sensors generate

System: the whole lot of applications that conforms Cross-CPP, including Marketplace Web Application and Marketplace Server.

Figures

About Cross-CPP

The objective is to establish an IT environment for the integration and analytics of data streams coming from high volume (mass) products with cyber physical features, as well from Open Data Sources, aiming to offer new cross sectorial services and focusing on the commercial confidentiality, privacy and IPR and ethical issues using a context sensitive approach. The project addresses cross-stream analysis of large data volumes from mass cyber physical products (CPP) from various industrial sectors such as automotive, and home automation. The business objective of the research is to allow for analyses of such data streams in combination to other (non-industrial, open) data streams and for the establishment of diverse enhanced sectorial and cross-sectorial services. The project will develop: (i) New models for integration and analytics of data streams coming from multi-sectorial CPP, including shared systems of entity identifiers applicable to multi-sectorial CPP (as well as the definition of agreed data models for data streams from multiple CPP aiming at defacto standard; (ii) Ecosystem, including a common Marketplace, and methodology to use such models to build multi-sectorial cloud based services, (iii) Toolbox for real-time and predictive cross-stream analytics, context modelling and extraction, and dynamically changing security policy, privacy and IPR conditions/rules and (iv) set of services such as services based on a combination of data streams from home automation and (electrical) vehicles to pro-vide enhanced local weather forecast and predict and optimise energy consumptions in households. The project will build upon the results from past and current projects, where results from the project AutoMat, addressing services developed based on data streams from vehicles, will be used as a basis for further development aiming to extend it to integrated, cross-sectorial data streams analytics. More information is available at https://cross-cpp.eu

Funded by the Horizon 2020 Framework Programme of the European Union

CROSS-CPP