URBANITE Data Management Platform

Fritz Meiners
Fraunhofer FOKUS
Digital Public Services
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany
fritz.meiners@
fokus.fraunhofer.de

Sonia Bilbao
TECNALIA, Basque Research and
Technology Alliance (BRTA), P.
Tecnológico Bizkaia, Ed. 700, 48160
Derio, Spain
sonia.bilbao@tecnalia.com

Gonzalo Lazaro
TECNALIA, Basque Research and
Technology Alliance (BRTA), P.
Tecnológico Bizkaia, Ed. 700, 48160
Derio, Spain
gonzalo.lazaro@tecnalia.com

Giuseppe Ciulla
Research & Development Laboratory
Engineering Ingegneria Informatica
Palermo, Italy
giuseppe.ciulla@eng.it

ABSTRACT

This paper describes the Data Management Platform developed in URBANITE H2020 project. This platform provides automatic mechanisms to harvest, curate, fuse and visualize existing open and proprietary data coming from different sources related to urban mobility and transportation (e.g. traffic data from cars, public transport, bikes or ferries; air quality and noise; events, parking, and so on).

KEYWORDS

Data harvesting, data curation, DCAT-AP metadata, data storage

1 INTRODUCTION

One of the main goals of the research carried out in URBANITE H2020 projects, is to provide algorithms, tools and models to support decision-making processes in the field of urban planning and mobility. This support is based on the analysis of the current situation based on harvested and fused data, on data simulations and the prediction of future situations when changing one or more variables. Hence, the availability of good quality data coming from heterogeneous data sources and its interoperability for data aggregation and fusion is highly important.

The Data Management Platform (DMP) provides the components for data acquisition, aggregation and storage. These components are:

- Data Harvesting, Preparation and Transformation covering the entire process of fetching, preparing, transforming, and exporting data for storage
- Data Anonymization to transform datasets in conformity with data protection requirements for further data analysis.
- Data Curation which deals with enrichment and annotation of data.
- Data Fusion and aggregation. Data aggregation is the process of gathering data and presenting it in a summarized format, e.g. to hide personal information or to provide information in a synthetic form. Data fusion is the process of integrating multiple data sources to produce more consistent, accurate, helpful information and sophisticated models than those provided by any individual data source.
- Data Storage & Retrieval providing the means to store and retrieve datasets, DCAT-AP compliant metadata, and related data.
- Data Catalogue offering the functionalities to discover and access the datasets collected and managed by the components of the URBANITE Ecosystem.

2 DMP ARCHITECTURE

Figure 1 represents the component diagram of the Data Management Platform (blue rectangle) and its interaction with the other modules in the URBANITE Ecosystem.

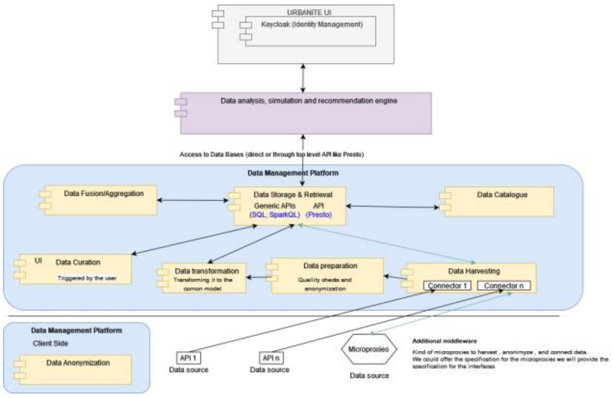

URBANITE UI
Keycloak (Identity Management)
Data analysis, simulation and recommendation engine
Access to Data Bases (direct or through top level API like Presto)
Data Management Platform
Data Fusion/Aggregation
Data Storage & Retrieval
Generic APIs
(SQL, SparkQL)
API
(Presto)
Data Catalogue
UI
Data Curation
Triggered by the user
Data transformation
Transforming it to the comon model
Data preparation
Quality checks and anonymization
Data Harvesting
Connector 1
Connector n
Data Management Platform
Client Side
Data Anonymization
API 1
Data source
API n
Data source
Microproxies
Data source
Additional middleware
Kind of microproxies to harvest , anonimyze , and connect data.
We could offer the specification for the microproxies we will provide the specification for the interfaces

Figure 1. Component diagram of the DMP

3 IMPLEMENTATION

3.1 Data Harvesting, Preparation and Transformation

The process of fetching, preparing, transforming, and exporting data (from now on referred to as *harvesting*), i.e. providing a way

Information Society 2021: 24th international multiconference*, 4–8 October 2021, Ljubljana, Slovenia*

to make heterogeneous data available in defined format and means of access, has been implemented following the form of a pipeline, as shown in Figure 2. This means that data is passed through the pipeline, and each component is agnostic of the other steps. This leads to lose coupling and improves flexibility allowing steps to be omitted if not necessary for a given data source. The pipeline has been implemented using the open source solution named Piveau Pipe Concept [1, 2].

Each of the components in the pipeline is implemented as a service that exposes a common RESTful interface. This way, services can be connected in a generic fashion to implement specific data processing chains. No central instance is responsible for orchestrating the services. A scheduler is in charge of launching the pipes and how services are connected together is specified in a JSON file known as the pipe descriptor.

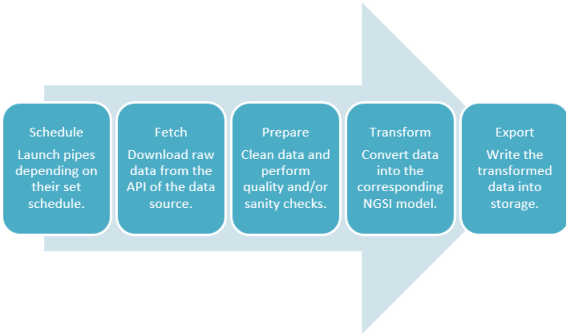

Schedule
Launch pipes depending on their set schedule.
Fetch
Download raw data from the API of the data source.
Prepare
Clean data and perform quality and/or sanity checks.
Transform
Convert data into the corresponding NGSI model.
Export
Write the transformed data into storage.

Figure 2. Harvesting process

In detail, the harvesting process would typically consist of the following steps:

1. The scheduler triggers a pipeline
2. The harvester retrieves the data from the source’s API and forwards it into the preparation component.
3. After cleaning and validating, the preparation component forwards the data to the transformation component.
4. The data is transformed to the applicable NGSI data model and forwarded to the exporter.
5. Finally, the exporter writes the harmonized data into the data storage component.

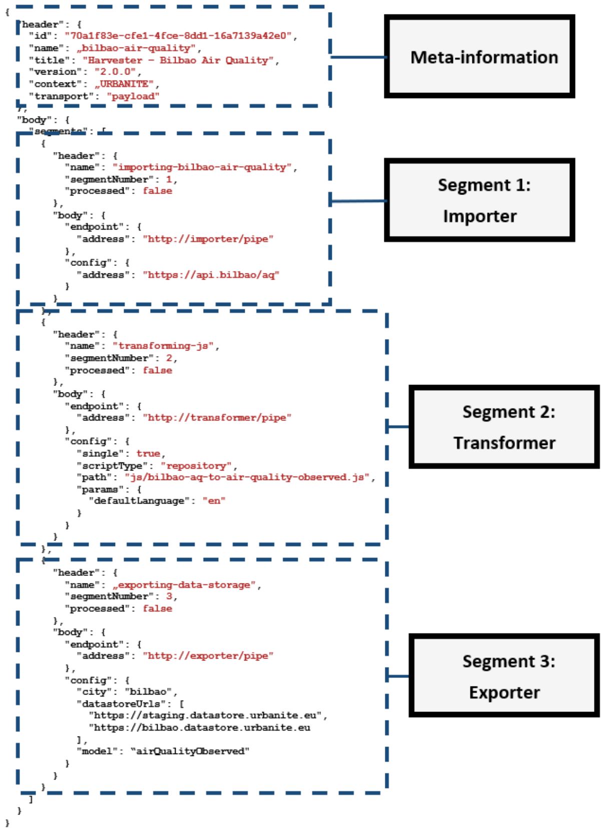
The scheduler serves two main purposes: keeping track of existing pipe descriptors and managing triggers for these pipes. Each pipe descriptor is stored as a JSON file and contains a definition of components (endpoints, chronological order, specific configurations) that make up the processing sequence. Each processing chain is defined in one of these files. The scheduler reads these files to become aware of which pipes are available. These can then be assigned to a periodic trigger for recurring execution.

The data harvesting component is responsible for fetching data from a given API. It does not alter the data. It can be considered the entry point of the data into the pipeline. As such, a dedicated component is required for each type of data source. The harvesting component may implement pagination mechanisms for handling data in chunks. However, this does not impact the pipeline – each chunk is handled individually and does not depend on other chunks.

The data preparation component is responsible for performing initial cleaning and sanitation of the data provided by the harvesting component. This ensures a fixed level of data quality and integrity, which is required by the transformation component to operate flawlessly.

Data transformation is a key step in the harvesting pipeline. It cannot be expected that the municipalities provide their data in one of the common data models developed by FIWARE used in the URBANITE context. As such, the transformation of the heterogeneous data sources into common models is vital for frictionless processing of the data henceforth. For a flexible approach, the actual transformation instructions are loaded via scripts, either JavaScript for JSON based payloads or XSLT for XML based payloads. More engines can be added as pipeline modules at a later point in time.

An example of a pipe descriptor is provided in Figure 3. Each of the segments describes a service in the pipe. In the example there are three services, the first named *importing-bilbao-air-quality* that downloads Bilbao air quality data, the second named *transforming-js* that transforms the data according to a JavaScript file and the third one named *exporting-data-storage* that invokes the storage and retrieval component to store the data and its metadata in two dedicated repositories. The segment number field indicates the order in which the service should be executed in the pipe.



```
{
  "header": {
    "id": "70a1f83e-cfe1-4fce-8dd1-16a7139a42e0",
    "name": „bilbao-air-quality",
    "title": "Harvester – Bilbao Air Quality",
    "version": "2.0.0",
    "context": „URBANITE",
    "transport": "payload"
  },
  "body": {
    "segments": [
      {
        "header": {
          "name": "importing-bilbao-air-quality",
          "segmentNumber": 1,
          "processed": false
        },
        "body": {
          "endpoint": {
            "address": "http://importer/pipe"
          },
          "config": {
            "address": "https://api.bilbao/aq"
          }
        }
      },
      {
        "header": {
          "name": "transforming-js",
          "segmentNumber": 2,
          "processed": false
        },
        "body": {
          "endpoint": {
            "address": "http://transformer/pipe"
          },
          "config": {
            "single": true,
            "scriptType": "repository",
            "path": "js/bilbao-aq-to-air-quality-observed.js",
            "params": {
              "defaultLanguage": "en"
            }
          }
        }
      },
      {
        "header": {
          "name": „exporting-data-storage",
          "segmentNumber": 3,
          "processed": false
        },
        "body": {
          "endpoint": {
            "address": "http://exporter/pipe"
          },
          "config": {
            "city": "bilbao",
            "datastoreUrls": [
              "https://staging.datastore.urbanite.eu",
              "https://bilbao.datastore.urbanite.eu
            ],
            "model": "airQualityObserved"
          }
        }
      }
    ]
  }
}
```

Figure 3. Example of a Piveau Pipe Descriptor

When a pipe is triggered the service in segment number 1 is called. Once finished, data that needs to be passed along the processing chain is written into a payload field of the next component in line. For smaller amounts of data this can happen directly, for larger amounts of data a pointer to an external datastore can be used.

3.2 Data Anonymization

The anonymization component is a RESTful microservice capable of transforming large datasets in conformity with data protection requirements for further data analysis. In order to achieve a certain degree of anonymization the user can mark specific attributes that are likely to reveal information about a person or a smaller group. Those identifiers are then transformed in a way that ensures a sufficient level of anonymization. Currently supported anonymization methods are suppression and generalization, which either delete attribute entries in a row or generalise them according to a fixed hierarchy, such as street -> zip code -> city.

3.3 Data Storage & Retrieval

The Data Storage & Retrieval component provides the means to store and retrieve datasets (transformed to URBANITE common data model compliant with FIWARE) and metadata (DCAT-AP). DCAT-AP [3] is used as the common metadata schema to describe datasets in URBANITE. Two repositories are used, one for the metadata and the other for the transformed data. This is shown in Figure 4.

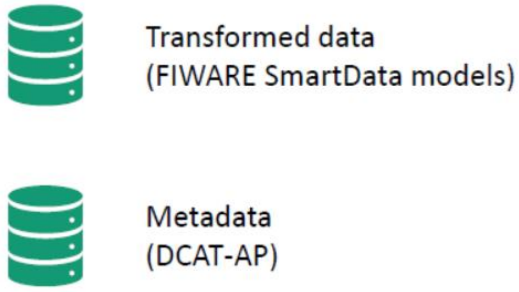


Figure 4. Data Storage & Retrieval repositories

The main concepts of the DCAT-AP model are catalogues, datasets and distributions. A catalogue represents a collection of datasets; a dataset represents a data collection published as part of a catalogue; and a distribution represents a specific way to access to specific data (such as a file to download or an API). This relationship is shown in Figure 5.

The concept dcat:Dataset informs about the title, description, access rights, creator, frequency, spatial/geographic and temporal coverage, spatial and temporal resolution, publisher, etc.

The concept dcat:Distribution provides metadata about the distribution, e.g. the property dcat:accessURL provides the information about how to access to specific data. Other important metadata related to the distribution are, for instance, the license, a description, the format of the data (e.g. CSV, JSON), etc.

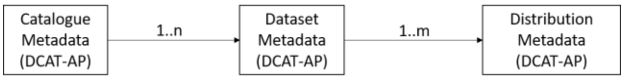

Catalogue Metadata (DCAT-AP)
1..n
Dataset Metadata (DCAT-AP)
1..m
Distribution Metadata (DCAT-AP)

Figure 5. Simplified DCAT-AP Model

The Data Storage & Retrieval component provides REST APIs so that the Exporter of the harvesting process can store the transformed data. Besides, a new DCAT distribution is stored which is associated with the existing metadata of the dataset, with accessUrl equal to the API endpoint to access the transformed data. An example of this is shown in Table 1.

Table 1. Instance of a distribution in JSON-LD format

```
{
    "@id" : "https://urbanite-project.eu/ontology/distribution/009b9f0e-e780-4e9d-8153-520dc8943195",
    "@type" : "dcat:Distribution",
    "description" : "Air Quality information for bilbao day 2021-05-01 in NGSI-LD representation",
    "format" : "http://publications.europa.eu/resource/authority/file-type/JSON_LD",
    "license" : "http://publications.europa.eu/resource/authority/licence/CC_BY",
    "title" : "Air Quality information for bilbao day 2021-05-01",
    "accessURL" : "https://bilbao.urbanite.esilab.org/data/getTDataRange/airQualityObserved/bilbao?startDate=2021-05-01T00%3A00%3A00.000Z&endDate=2021-05-01T23%3A59%3A00.000Z"
}
```

The technology stack used to implement the component consists of three levels, as depicted in Figure 6.

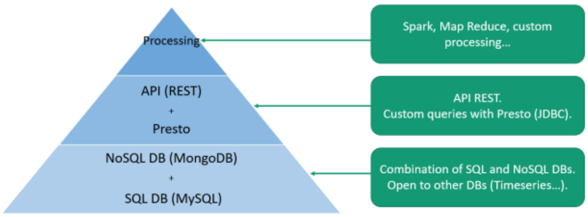

Processing
API (REST)
+
Presto
NoSQL DB (MongoDB)
+
SQL DB (MySQL)
Spark, Map Reduce, custom processing...
API REST.
Custom queries with Presto (JDBC).
Combination of SQL and NoSQL DBs.
Open to other DBs (Timeseries...).

Figure 6. Data aggregation & storage technology stack

At the bottom level we have the storage repositories, being a combination of different types of databases: SQL databases, like MySQL and NoSQL databases, like MongoDB. The design is also open to the usage of other storage mechanisms that may be useful in the future, such as timeseries databases, files, semantic triple stores, etc.

The intermediate level offers the mechanisms for both storing and retrieving data. In turn it consists of two components: a REST API with predefined methods for inserting or accessing data and metadata, and a JDBC connection, through the Presto software, to perform custom queries (SQL statements) different to those offered by the API. All the interaction with the storage system is made through these two mechanisms, not allowing direct access to the data. This makes the choice of the specific database that stores the data transparent to the upper processing layer and can be modified without affecting the processes that make use of the component.

Finally, at the top level we have the processes that can be defined to feed the databases or make use of the data, e.g. data aggregation processes.

3.4 Data Catalogue

The Data Catalogue offers the functionalities to discover and access the datasets collected and managed by the components of the URBANITE ecosystem. Apart from the possibility to search over these datasets, the Data Catalogue also offers the possibility to search useful data across external "federated catalogues" (such as Open Data Portal) to increase the chance to find useful data.

The administrator is in charge of managing the federation of the catalogues, where a catalogue represents a data source. He/she can add new catalogues, delete or edit the existing ones. Moreover, the administrator can manage the platform configurations.

The end-user is then able to perform a federated metadata search among the harmonized DCAT-AP datasets provided by the federated catalogues. Moreover, the end-user can perform SPARQL queries over the federated RDFs provided by the federated catalogues, or he/she can access to statistics about the federated catalogues.

The Data Catalogue exposes APIs to access its functionalities; thus, an external system will be able to interact with the platform using such APIs.

The Data Catalogue is based on Idra [4]. Idra is a web application able to federate existing Open Data Management Systems (ODMS) based on different technologies providing a unique access point to search and discover open datasets coming from heterogeneous sources. Idra uniforms the representation of collected open datasets, thanks to the adoption of international standards (DCAT-AP) and provides a set of RESTful APIs to be used by third-party applications.

Figure 7 depicts the interaction among the Data Catalogue and the other URBANITE's components.

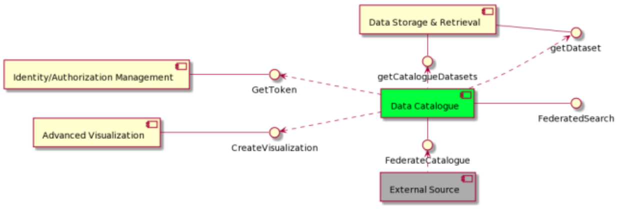
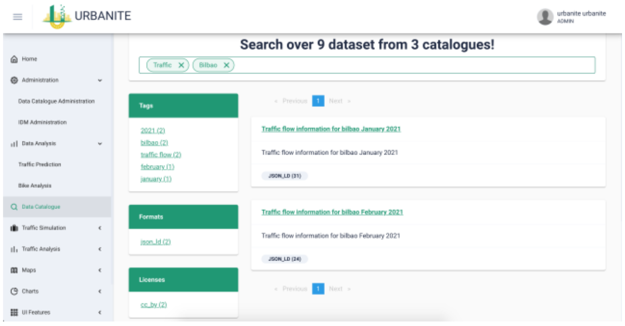


Figure 7. Data Catalogue - Component diagram

The Data Catalogue interacts with 1) Identity/Authorization Management component to allow administrators to access their specific functionalities retrieving the access token that will be further provided to the APIs, 2) Advanced Visualization to build visualization taking advantage of the DCAT-AP distributions it manages and 3) Data Storage & Retrieval to retrieve DCAT-AP datasets and distribution metadata. Finally, the Data Catalogue component is able to federate external sources such as Open Data portals or other sources providing DCAT-AP metadata.



Search over 9 dataset from 3 catalogues!

Traffic × Bilbao ×

2021 (2)
bilbao (2)
traffic flow (2)
february (1)
january (1)

json_ld (2)

cc_by (2)

Traffic flow information for bilbao January 2021

Traffic flow information for bilbao January 2021

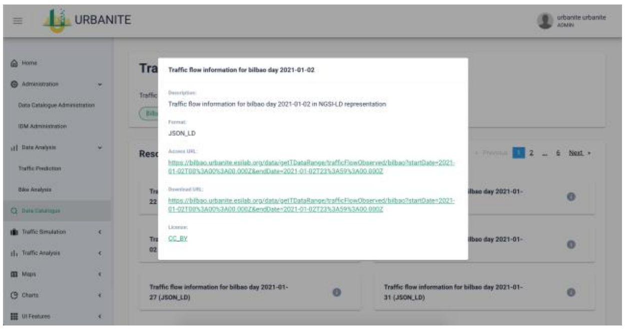
JSON_LD (31)

Traffic flow information for bilbao February 2021

Traffic flow information for bilbao February 2021

JSON_LD (24)

Figure 8. Data Catalogue – Dataset search (example)



Traffic flow information for bilbao day 2021-01-02

Description:
Traffic flow information for bilbao day 2021-01-02 in NGSI-LD representation

Format:
JSON_LD

Access URL:
https://bilbao.urbanite.esilab.org/data/getTDataRange/trafficFlowObserved/bilbao?startDate=2021-01-02T00%3A00%3A00.000Z&endDate=2021-01-02T23%3A59%3A00.000Z

Download URL:
https://bilbao.urbanite.esilab.org/data/getTDataRange/trafficFlowObserved/bilbao?startDate=2021-01-02T00%3A00%3A00.000Z&endDate=2021-01-02T23%3A59%3A00.000Z

License:
CC_BY

Traffic flow information for bilbao day 2021-01-27 (JSON_LD)

Traffic flow information for bilbao day 2021-01-31 (JSON_LD)

Figure 9. Data Catalogue – Details of a dataset (example)

ACKNOWLEDGMENTS / ZAHVALA

This research is funded by the European Union's Horizon 2020 research and innovation program under grant agreement number 870338 (URBANITE: Supporting the decision-making in urban transformation with the use of disruptive technologies).

REFERENCES

[1] Kirstein F., Stefanidis K., Dittwald B., Dutkowski S., Urbanek S., Hauswirth M. (2020) Piveau: A Large-Scale Open Data Management Platform Based on Semantic Web Technologies. In: Harth A. et al. (eds) The Semantic Web. ESWC 2020. Lecture Notes in Computer Science, vol 12123. Springer, Cham. https://doi.org/10.1007/978-3-030-49461-2_38

[2] Piveau solution. https://github.com/piveau-data

[3] DCAT-AP 2.0.1. https://joinup.ec.europa.eu/rdf_entity/http_e_f_fdata_ceuropa_ceu_fw21_f32d70b6e-0d27-40d9-9230-017e4cd00bcc

[4] Idra - Open Data Federation Platform https://idra.readthedocs.io/en/latest/