Cloud services and Data analysis

Topics · https://mobidatalab.eu/knowledge-base/cloud-services-and-data-analysis/



Given the heterogeneity and peculiarities of mobility data, and the various constraints related to their safe and trusted sharing, MobiDataLab envisions the usage of an open federated cloud architecture. In this architecture, complex and often contrasting requirements coming from FAIR (Findability, Accessibility, Interoperability, and Reusability) and privacy principles can be enforced easily and practically.

In principle, a federated cloud can support the sharing of arbitrary resources from arbitrary application domains, with arbitrary consumer groups across multiple administrative domains. A federated architecture is beneficial for MobiDataLab, given its main objective of fostering the sharing of data amongst transport authorities, operators and other mobility stakeholders operating in Europe

which in most of the cases want to maintain the governance of their data.

MobiDataLab targets technology challenges aimed to propose to mobility stakeholders a replicable methodology and sustainable tools that foster the development of a data sharing culture in Europe and beyond. Facing such challenges requires the use of techniques that deal with data fusion, privacy and anonymisation, geographical and semantic enrichment, harmonisation/standardisation, and data processing in general. These will be indeed required to improve the quality, accessibility and usability of mobility data, but also to allow each mobility data provider to share securely and safely their data.

The MobiDataLab Platform

The MobiDataLab platform will demonstrate a cloud-based prototype platform for sharing transport data, accessible to interested mobility actors. It is technically designed according to federated cloud principles. The MobiDataLab platform will showcase how to facilitate access to mobility data in an open, interoperable, and privacy preserving way through the development of open and accessible tools. The platform is primarily designed to demonstrate and offer solutions to reduce and, in some cases, remove current technical limitations identified as barriers to data sharing and reuse. The architecture is depicted in Figure 1. We explain its main key constituent elements next.

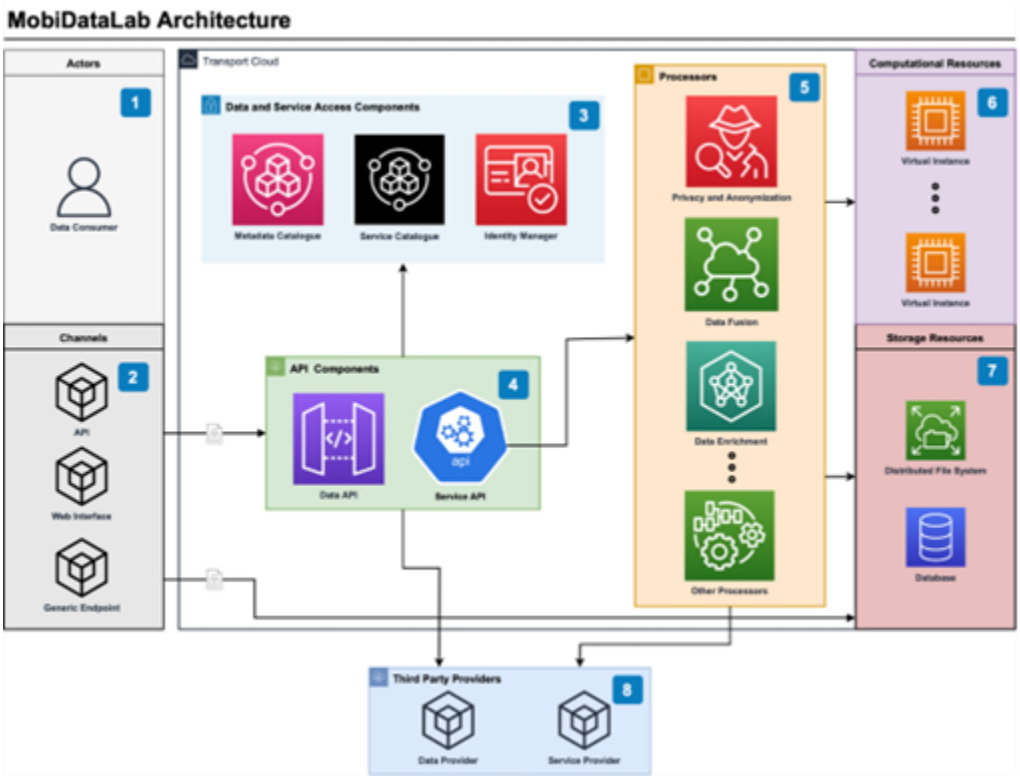

Figure 1: MobiDataLab platform architectural design

The Data Consumer actor is any entity that is interested to use the data and services available within the platform like a data scientist, researcher, domain expert, transport customers and services external to the platform.

The Data consumer interact with the platform through several channels i.e., (1) API endpoints (mainly dedicated to REST API services), (2) web interface endpoints, i.e., dedicated to services which need interaction with the end user (for example scenarios involving data analysis and visualisation tasks), and (3) generic endpoints – for instance, a SPARQL endpoint may enable a Data Consumer to access some knowledge base via RDF queries.

The internal components are associated with Data and Service Access (box 3), APIs (box 4), and Data Processors (box 5). These lean on the usage of Computational (box 6) and Storage (box 7) Resources.

The computational resources provide computation capabilities, i.e., virtual instances provided by the chosen cloud provider and intended to support the execution of the services deployed and running within the Transport Cloud platform. Storage resources enable the storage of information within the [platform such as distributed file system solutions and generic database solutions (e.g., PostgreSQL with PostGIS extension, SPARQL engines, and so on).

Data and Service access

The Data and service component includes a metadata catalogue, a service catalogue and an identity manager. The Metadata component lists the content made available by the MobiDataLab project, such as datasets or metadata and provides access to the mobility datasets via a web portal and API. GeoNetwork and CKAN are employed to support a wide range of data format and fuse the data collection functionalities provided by the two services.

Access to data and services is a critical part of the whole infrastructure. For the data access, many data providers already built their data offers using standard formats and interfaces to make their data available.

One of the services proposed by the MobiDataLab platform is navitia.io. Navitia.io is an **open-source**, **open-service** API suite based on **open-data**, offering advanced features **dedicated to mobility**. Navitia provides the following services:

- Multi-modal journeys computation

- Line schedules
- Next departures
- Exploration of public transport data
- Search & autocomplete on places
- Isochrones

Technically, Navitia is a HATEOAS API that returns JSON formatted result.

Navitia offers **unified APIs *(x4)*** for easy handling:

1. **Passenger Information** API – *Offer the best intermodal routes, taking account of traffic information.*
2. **Search** API – *Help passengers find their way around by geolocating and displaying points of interest around them.*
3. **Autocomplete** API – *Improve passenger experience when entering searches with a powerful, comprehensive autocomplete feature.*
4. **Isochrone** API – *Offer the innovative search by travel time with the isochrone feature.*

Identity and Access Management*

The Identity and Access Management may include identity provisioning, authentication, and authorisation. In the context of mobility data sharing, these mechanisms can be favourably integrated into an API gateway solution.

An **API Gateway** is the traffic manager that interfaces with the actual backend service or data, and applies policies, authentication, and general access control for API calls to protect valuable data. An API gateway is the way to control access to your back-end systems and services, and it was designed to optimize communication between external clients and your backend services, giving you’re the clients a seamless experience. An API gateway ensures scalability and high availability of services. It is responsible for routing the request to the appropriate service and sending back a reply to the requestor. An API gateway maintains a secure connection between the data and APIs and manages API traffic and requests including load balancing. The gateway applies policies, authentication, and general access control for API calls to protect valuable data. An API gateway takes all API calls from clients and routes them to the right microservice using request routing, composition, and protocol

translation.



One of the primary reasons an API gateway is used is that it can invoke multiple back-end services and aggregate the results. Rather than customers having to send a request for each individual service, they can send them to the API gateway, which then passes the request on to the relevant service. In addition, an API gateway provides an alternative to the one-size-fits-all style API. An API gateway can also expose a different API for each client, a necessity in today's ever-evolving environments.

*https://www.tibco.com/reference-center/what-is-an-api-gateway

API component

An API is a programmatic functionality exposed by the functionality provider to the users by the means of internet/HTTP exposure. Software development is homogeneously made with languages tied to the operating system the software runs un and the running scope of the software remains local to the computer the software runs on. APIs will allow local software to call functionalities online regardless the language the software or the API is programmed with, nor the location of the API since the API will be available online.

The API components consists of the Data API and the Service API. The Data API components take the metadata from the catalogues, including the references connecting to the services and accessing the data while the services API will deal with listing and describing the services within the platform.

The first step is to make data and services available by following a standardised and open interfaces approach. Then, the challenges of integrating non-standardised interfaces and data must be analysed to determine the conversion path that will the data to common industry standard format and content.

The different options for integrating proprietary data and service APIs will be explored based on the data and services provided by the project's stakeholders (i.e., the MobiDataLab reference group). Commercial copyright, geographical limitation, GDPR content, data volume agreement may be of the essence before any proprietary content partnership.

We identified metadata catalogue APIs to deliver metadata catalogues and metadata items to the users, a service catalogue API that will depict the different services available within the MobiDataLab platform, basic data access API to supply the API users with usual content we identified as the most common expectation and advanced data and service access API that will allow advanced parameters from the API user to specify filters and types of content to be returned by the advanced APIs.

The Processors

An essential component of the MobiDataLab platform that enable data analysis, are the *processors*. A processor is a component that models some function, that operates on some input data according to some specific logic, in order to produce a final output. Data processors may be needed to perform:semantic enrichment based on common vocabularies, geographical enrichment based on common geometries, data format translation, data fusion, data anonymisation, injection of license specification, and any other data processing tasks that are relevant to the goals of the project.

Three of the most significant processors that enable advanced mobility data analysis in MobiDataLab are the semantic enrichment, geographical enrichment and data anonymization.

Semantic Enrichment

This processor enriches mobility data with semantic dimensions, e.g., points of interest around the end-user, the weather conditions, the transportation mean of a traveller and more. The demonstrator uses the notion of multiple aspect trajectory (MAT) to enrich trajectories or part of them with semantic aspects. For instance, weather conditions might get associated with trajectory segments, whole Points of Interest might get associated with a trajectory stop or transportation means linked to trajectory move segments. Personal data can be associated with the moving object (e.g., gender, social media profile, etc.). The semantic enrichment demonstrator is therefore focused on building such semantically enriched trajectories in the context of the MobiDataLab architecture. The

demonstrator has been derived from the tool called MAT-Builder1. MAT-builder notable points are that it can be implemented purely on top of open-source libraries and tools, and it is effectively able to semantically enrich raw trajectories via the use of a variety of external data sources provided by the platform. Furthermore, it can exploit the semantics provided by the Linked Open Data (LOD) world by generating Knowledge Graphs (KG). These knowledge graphs can be subsequently imported in some Resource Description Framework (RDF) triple store of preference (e.g., GraphDB) and linked with LOD sources for later analysis and querying. The code and documentation concerning the semantic enrichment processor can be accessed at: https://github.com/MobiDataLab/mdl-semantic-enrichment.

Geographical enrichment processor

Among the services developed and provided as a set of APIs by the MobiDataLab project is listed the geographical enrichment processor. The geographical enrichment processor has the objective of enriching mobility data queried by the user application with geographical data content matching the end user application quest regarding the place of interest.

Supported by its own ecosystem, and responding to public solicitations the geographical enrichment processor is backed by Navitia, a public transport journey planner, OpenStreetMap which is an open collaborative cartography service online to supply maps support for applications and OverPass plus HERE technologies for road and ground level functionalities.

The geographical enrichment processor was created using Spring Boot framework along with some other Java and JavaScript libraries in Rest API orientation mindset to comply with an inter-operability vision.

Data anonymization processor

Mobility data collected by some organizations (municipalities, companies, etc.) might be of interest to researchers, in which case these datasets can be released to research organizations or even made public. it is important that when such data contain information about individuals some measures are taken to prevent the release of sensitive information about such individuals. These measures are collectively known as anonymization measures. Anonymous data is not personal data and hence falls outside the scope of the GDPR. However, one of the main barriers to mobility data sharing is assessing the privacy risks of the owned data and being able to anonymize them.

The privacy and anonymization processor allows the users to anonymize trajectory micro data sets using one of the several included methods, to analyse a mobility dataset in a privacy-preserving way and to compute privacy and utility metrics. A trajectory microdata set is a microdata set that contains trajectory data. This kind of datasets are special because the location information included in them can be considered both as quasi-identifiers and sensitive information. Trajectory microdata is prone to privacy attacks on individual users because of two defining characteristics: Trajectory data are highly unique and hard to anonymize.

The anonymization module includes 6 anonymization methods, 1 privacy-preserving analysis and aggregate method and 5 methods to compute different utility and privacy metrics. It also provides a command line interface (CLI) that lets users to use all the module functionalities in a straightforward way. The module is also ready to be deployed in a server and to process requests through an API. It has been designed with a focus on modularity, where pseudonymization or anonymization methods can be built using different components dedicated to preprocessing, clustering, distance computation, aggregation, etc. We have focused on making it easy to add new methods and components, in order to encourage contributions from other researchers.

Resources

Download the Datathon Tools pocket guide