General Software Architecture

Figure 1 depicts the general software architecture of WP 8.6. It considers a generic “Vertical solution” and another sensor network (abstracted by Object Digital Twins) as Data Generators.

The ODA service, described later, stores the data pushed by the two generators offering the observable data to its clients. We identify three possible clients: Task 8.6.1 Models, and Task 8.6.1. Analytics and the Vertical solution.

A Vertical Solution is a set of hardware/software technologies acting in a specific vertical context (e.g. building, mobility, public lighting, water, etc.) and it is able to collect data from a particular network of data sources (e.g. monitoring sensors). It contains:

- one or more DGs,
- zero or more DCs.

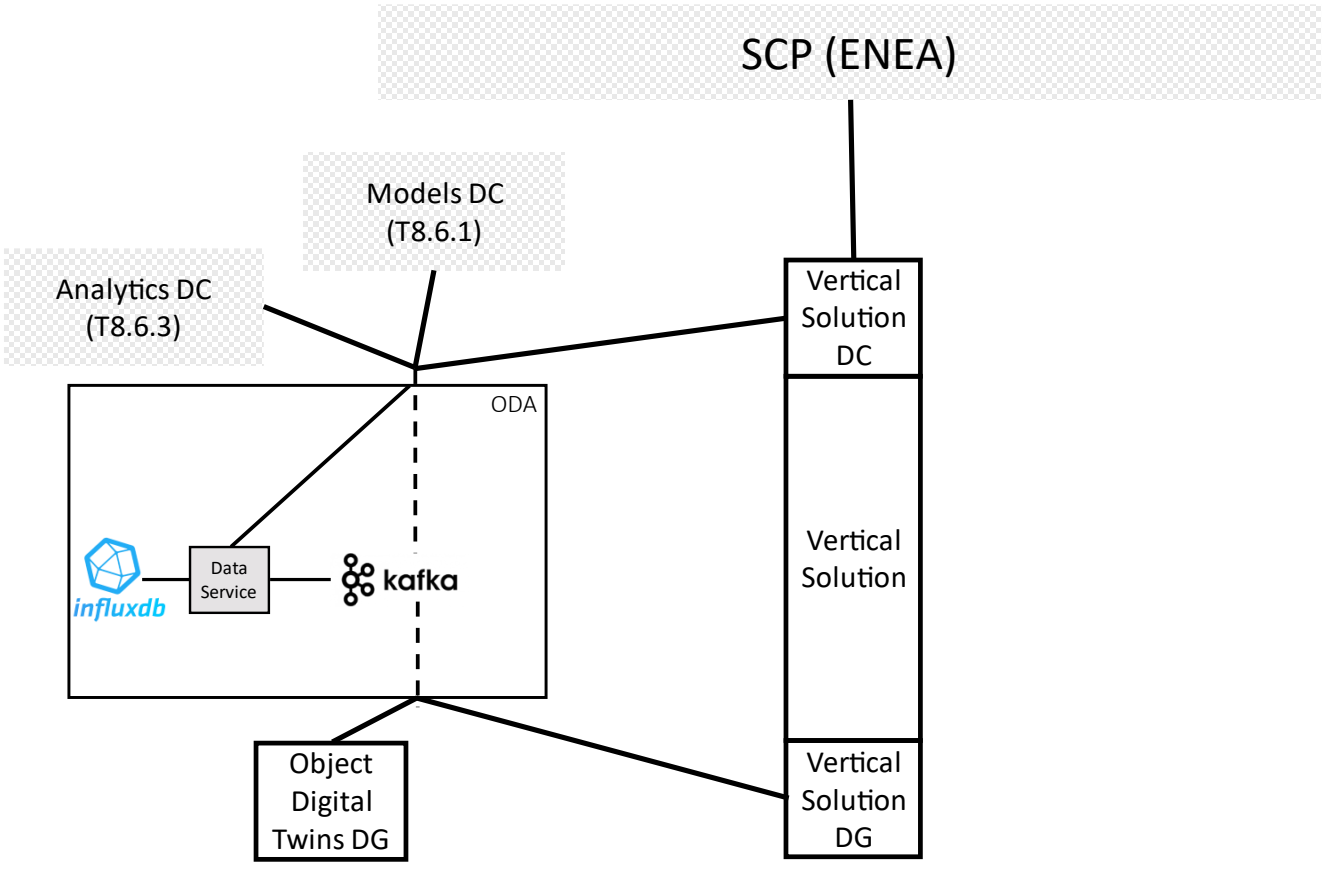

Figure 1: ODA in the WP 8.6

A Vertical Solution could interface with the Smart City Platform (SCP by ENEA) with a specific DC that:

- reads from ODA;
- transforms the data (for example, with a different granularity);
- represents the data with the UrbanDataset format;
- sends the UrbanDataset to the UrbanDatasetGateway web service in the SCP.

The ODA service

The Observable Data Access service (ODA), depicted in Figure 1, is fed by Data Generators (DGs) that push data acquired by sensors, digital twins, or services through a suitable REST API. The ODA stores data in the short term (e.g., a week, a month) and provides data access to Data Consumers (DCs) in two ways:

- A DC can subscribe to a certain topic to get data streamed by generators about such a topic.
- A DC can pull data by invoking a suitable REST API provided by ODA, and

Therefore, the data flow in a system exploiting an ODA service is bottom-up, from the DGs to the DCs.

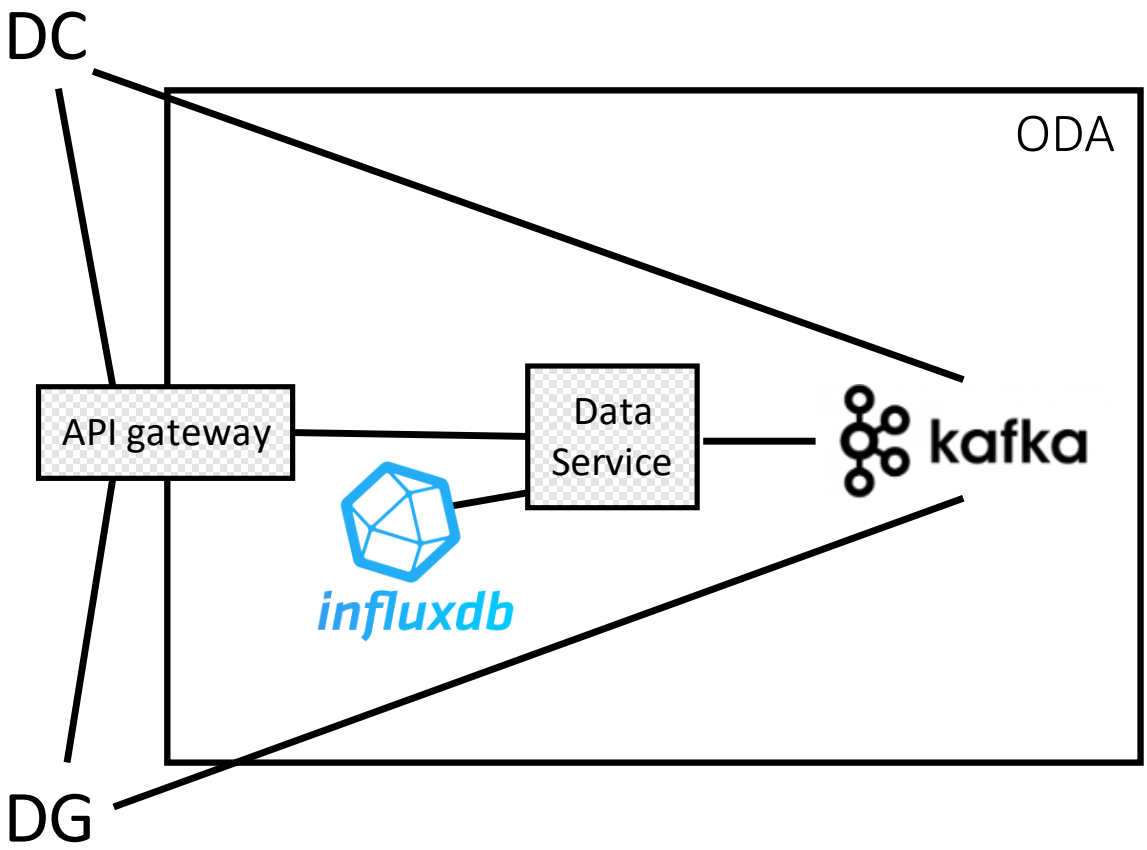

Figure 2: The ODA service

The database of the ODA is a time series database, implemented by exploiting InfluxDB.

Apache Kafka is used as a broker to implement the queue where DGs push data and DCs receive subscribed data.

Both DCs and DGs must register to the ODA service through an API gateway, which delivers the endpoint to reach the Kafka broker. Figure 3 shows how DGs must act to send data to the ODA service.

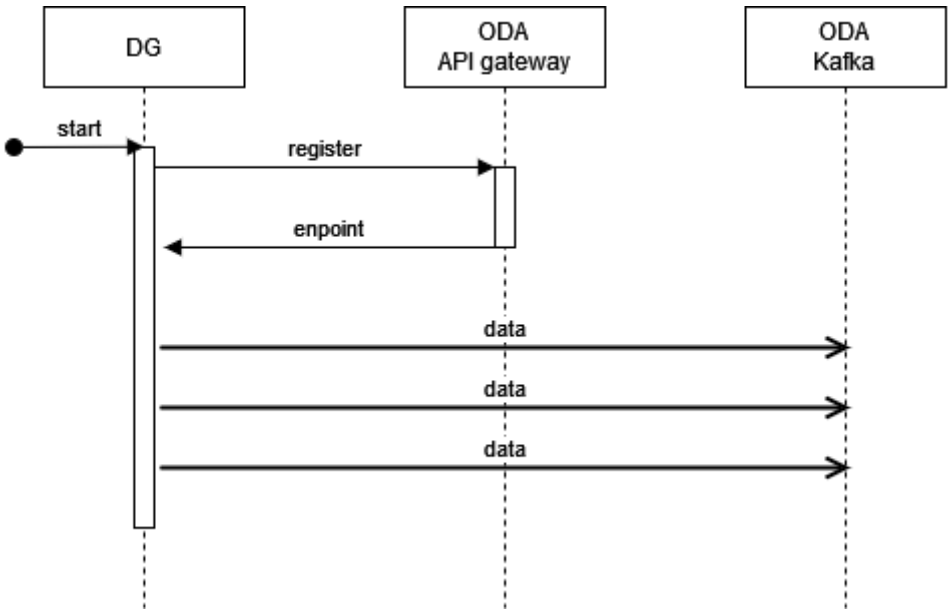

Figure 3: DG sending data

Figure 4 shows how DCs interested in the data streaming must act to received data from the ODA service.

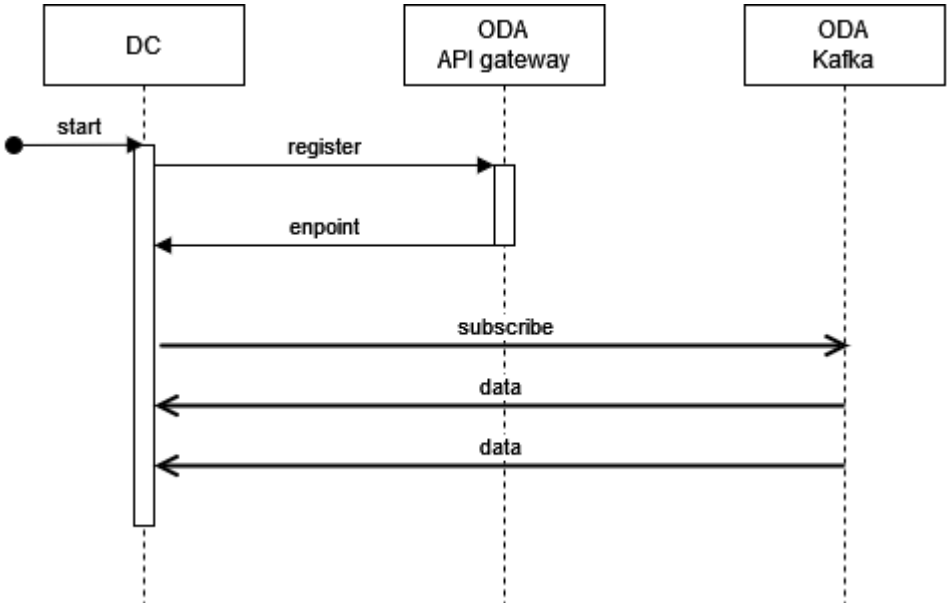

Figure 4: DC receving streamed data

DCs that pull data stored in the DB must send a query to the API gateway. The Data Service component acts as a consumer of the Kafka brokers to store the data in the database.

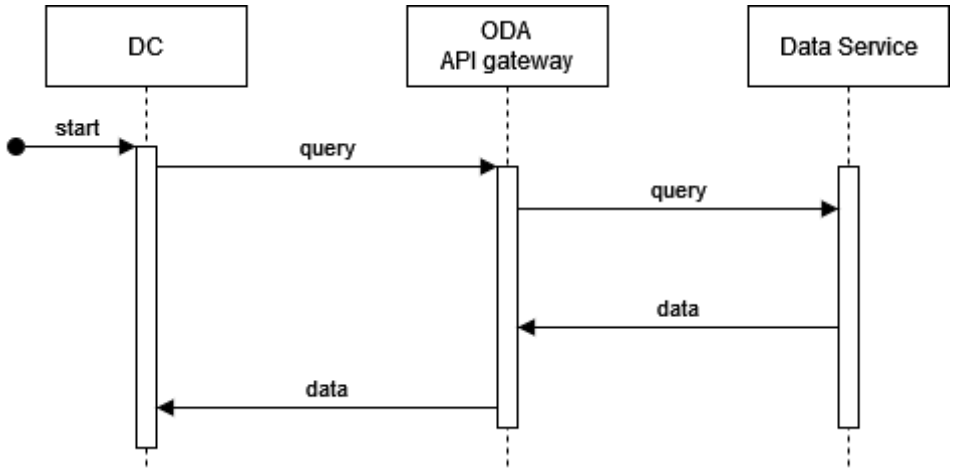

Figure 5: DC queries data stored in the DB

Data format

DGs must send data to the ODA service in the following JSON format:

```
{
  "timestamp":
  "generator_id":
  "topic":
  "data": { whatever data}
}
```

Where is indicated that the timestamp of the data, the generator identifier, and the topic are mandatory fields. The payload can be specified freely in the “data” field.