Deliverable

D3.1 IoT stack and API specifications v1

Onboarding IoT data in the DUET platform

Project Acronym:	DUET	
Project title:	Digital Urban European Twins	
Grant Agreement No.	870697	
Website:	www.digitalurbantwins.eu	
Version:	1.0	
Date:	21.9.2020	
Responsible Partner:	imec	
Contributing Partners:	-	
Reviewers:	T. Dalianis (ATC) Thomas Adolphi (VCS) Walter Lohman (TNO) Yannis Charalabidis (external expert) Pieter Morlion (external expert)	
Dissemination Level:	Public	X
	Confidential – only consortium members and European Commission	

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

Revision History

Revision	Date	Author	Organization	Description
0.1	12.06.2020	Philippe Michiels/Koen Triangle	imec	Initial structure
0.2	08.07.2020	Philippe Michiels	imec	editing & processing feedback
0.3	13.07.2020	Philippe Michiels	imec	finishing touches
0.4	31.08.2020	Philippe Michiels	imec	review
0.5	11.09.2020	Philippe Michiels	imec	post-review changes
1.0	18.09.2020	Koen Triangle	imec	Final version

Editorial note

At the time of writing the proposal there was little understanding on the design and scope of the digital twin project. The boundaries of the DUET system were not entirely clear. In the early design phase it became clear that IoT stacks would not be part of the core system. Since IoT stacks and their roles are quite well understood by now, the focus of this deliverable has shifted to the mechanisms of connecting the IoT stacks and other data sources with the platform and onboarding their data. As such the title is somewhat misleading to the reader.

Table of Contents

Executive Summary

A lot of the value of urban digital twins follows from being able to interconnect data sources and models (such as Cityflows - one view on multimodal flows in the city) to gather new insights into the dynamics of urban areas, maximally tapping into the principle of serendipity. However, combining various data sources still is far from trivial. Combining data sets with different formats, using different units and without common identifiers in itself is already difficult to say the least. Add in different levels of quality and availability and the hopes of combining data sets quickly disappears.

For this reason, DUET is designed to normalize incoming data onto a common data model. This data model will be aligned with common standards and can be extended by the DUET administrator to support more data formats. When mapped to a unified data model, it becomes much easier to assess the compatibility of datasets and to integrate them into a solution.

This document gives a technical description on how the on-boarding of IoT data sources and by extension geographical data sources is designed. We consider three types of data sources:

1. IoT event data: live event streaks of sensor data coming in at a steady rate. Two types of interaction mechanisms apply:
    a. Webhook ingest: the data provider actively pushes data to DUET,
    b. Polling ingest: DUET polls the data source for new data regularly.
2. IoT time series data: these contain historical sensor data. Queries from users are relayed to the providing system which executes the query and returns the result. For this to be possible, the client system needs to implement a basic interface to allow such querying.
3. Geographical data: these sources contain geographical data, typically this data comes from a web feature service or another geo data source.

We also discuss how data is registered in a central data catalog system and how a knowledge graph mapping is used to uniformize the data in the Digital Twin.

Note 1:
Some components in the digital twin may have a dual role. Typically, simulation models not only consume data but also publish data. Other examples may be data processors that prepare and/or enrich data for further use in the digital twin.

Note 2:
It is unclear if context brokers that provide context about sensor data should be considered a fourth possible data source type. This will depend on further insights from developing the digital twin architecture.

1. Introduction

Digital twins promise better monitoring, new insights and the possibility of executing what-if-analysis. This capability however hinges on the availability of data and the possibility to combine several data sources. This in turn implies that the data is interoperable (EIF - The New European Interoperability Framework). Unfortunately, full interoperability in data is still the exception, not the rule.

In order for DUET to offer data in a meaningful way to city planners and other users, it has to facilitate non-interoperable data to be onboarded and then mapped to a uniform format or ontology. By default the most common interoperable standards such as NGSI-V2, NGSI-LD, some OGC standards, etc. will be supported by DUET out of the box. Other formats, including custom and non-standard formats provided by external parties, should be easily onboardable.

See also:

- NGSI standards (v2 | ld)
- OGC - Open Geospatial Consortium

1.1 Objectives

The DUET connector architecture is designed to deal with versatile data sources and can handle the mapping of large quantities of data in a scalable way. This deliverable has the following objectives:

1. Onboarding IoT Event data & context data
    a. supporting for NGSI-V2 sources
    b. support for OGC sensor things
    c. support for custom data sources
2. Onboarding IoT time series data
    a. supporting the NGSI-V2 interface for the alpha version
    b. support for other (custom) sources
3. Onboarding Geographical data
    a. OSM as a road network
    b. Flemish road register for the alpha version
    c. OGC standards City GML and GeoJSON(-LD)
    d. Other data sources as needed

1.1.1 Smart City Domains

DUET will define a default set of smart city domains that can be governed by its administrators. These serve as chapters for the platform and can be used to navigate through available data sources. Additionally, these can be used to refine searches in the knowledge graph.

The pilots will focus mostly on the mobility and environment domains and on their interactions more specifically. However, it may be possible that other use cases are on-boarded as well. Below we list the different smart city domains and typical data sources.

Note:
Technical details about the sensors will be added to subsequent releases of this document as the pilots progress.

Note:
For more details about the pilots, we refer to D3.3 Smart City domains, models and interaction frameworks v1.

Default smart city domains:

- **Mobility**: the mobility domain focuses on everything related to traffic, traffic optimization, congestion avoidance, traffic circulation planning etc. All pilots as well as the alpha version will have a mobility component. At this point in time we envision to use the following sensors:
    - WiFi scanning data for assessing how busy an area in the city is (e.g., https://citymesh.com/)
    - Mobile cell data for similar estimations (e.g., proximus Real Time Crowd Management data https://proximusapi.enco.io/asset/rtcm/documentation)
    - Counting loops in roads operated by authorities
    - Cameras for multi-modal traffic counting (e.g., www.telraam.net)
    - Other cameras and motion sensors to detect road users

    In some cases, the accuracy of automated traffic counts will be validated with manual counting.

    Using these counts and several traffic models the pilots will each focus on one or more of the following use cases:
    - Mapping crowd and traffic movement in the city
    - Simulating the implementation of a traffic circulation plan in the city and its effects on air quality (see also environment)
    - Short term prediction of air quality based on meteorological data and expected traffic intensity on a wider scale (see also environment)
    - Mapping traffic intensity and associating it with other data such as number of accidents, security, demography etc.

- **Environment**: this domain will also be part of the different pilots. The domain focuses on anything that is environment related including air quality, water quality, (de)forrestation, noise, climate & weather, etc. The pilots will work to associate air quality with traffic and mobility data, allowing local authorities to simulate the effects of implementing mobility plans.

    There are many sensors available for monitoring air quality, ranging from very accurate but expensive sensors typically used by governmental organizations or institutions to very cheap but not very accurate sensors that can be deployed on a larger scale. The reliability of the less expensive sensors can be increased by calibrating their measurements using the results of the more expensive (reference) sensors.

    By mapping traffic intensity and weather conditions to measured air quality attributes, it becomes possible to create mathematical models that can predict the impact of traffic changes on air quality.

- **Economy**: The economy domain is less driven by sensor data and more by statistical data such as the location and nature of businesses, demographic data. Sensor data may nevertheless be relevant. Examples are mobility metrics and data on road capacity. All these metrics can be relevant for city planners to decide where to develop what activities in the smart city.

- **Safety & security**: This domain focuses on getting a better understanding of safety and security in the city and their driving factors. Typical use cases rely on statistical data of accidents and incidents in cities along with other data such as demography, aspects of the neighbourhood and even weather data to predict the risk of incidents at any given time. This allows preventive measures such as increased police presence and pre-emptive traffic control.

- **Health & living**: The health & living domain focuses on quality of life in the smart city. It is connected to the other domains in several ways.
    - Health has obvious connections with the environmental domain given that environmental factors are linked to well being. Other interesting links with health are education and demographics.
    - Another driver for the quality of life in the city is the presence or absence of facilities such as schools, shops, hospitals,etc.
    - Health and quality of life are also linked to security in that a feeling of insecurity can be a major stress factor and unsecure neighbourhoods are typically less healthy to live in.

- **Energy**: Energy production and consumption are closely tied to the environment domain, but also to health & living. A few example use cases are:
    - Monitoring energy consumption to identify areas with poorly isolated housing
    - Matching local energy production (wind & solar) with storage capacity (batteries in houses, factories and cars) and with demand can result in savings by consuming the energy closer to the source

The following smart city domains are less sensor driven. Duet will not focus on them but might on-board use cases if the occasion presents itself.

- **Public space**
    - City planning of public facilities
    - Determining the impact of events and infrastructure works
    - Inventorizing & managing the public domain in terms of security, access control, occupation, etc.

- **Government & regulation**
    - Publishing regulatory data, announcements, activities, etc.
    - Automation of administrative processes for civilian matters, business matters & education matters
    - (Semi-)automatic validation of permit requests

- **Culture**
    - Publishing of cultural data & the cultural calendar
    - Personalized cultural offerings
    - Preservation of heritage

- **Education**
    - Personalized training trajectory
    - Optimize educational offering based on demography and other parameters
    - Make training materials public

Digital twins offer the opportunity to combine data from different domains to figure out the relationship between different aspects of the city and better understand the dynamics that drive the way in which cities work. Understanding these dynamics enables us to experiment with the city in order to improve the city across the different domains.

1.2 Roadmap

In subsequent work, the following questions have to be answered

1. What is a good query language or API for querying the different data sources? How do we implement the interaction patterns? More specifically:
    a. What approaches exist to connect live data streams from sensors. How can we publish them and how do we determine buffering behavior?
    b. How do we connect context data and use that data to provision out digital twin context graphs?
    c. How do we implement querying data across the DUET platform?
    d. How do we connect geo data and make it sufficiently queryable?
2. Do we need to wire-up context data sources and if so, how? Context data is important and having access to it will drive several use cases such as clicking on a sensor on a map and displaying relevant context information. Also, when creating simulations, interactions with the virtual city twin will drive manipulations of that context. How do we wire this up?
3. Is there a good uniform way of dealing with geographical data, how to connect geo data sources? We have to keep in mind that many of the existing Geo standards are fit-for-purpose and mapping them to a universal format may not make a lot of sense.
4. How do we accommodate for interactions of the users. Can we look at ETSI SAREF for defining an interaction model using tasks and commands for instance?

2. IoT Stacks & data sources

2.1 Smart City Domains, sensors and IoT data

2.1 Introduction

Regardless of the standards they implement, scalable IoT stacks will come featuring typical components.
See also https://www.fiware.org/ and Synchronicity

The starting point is where sensor data enters the stack via the southbound. This usually happens via an agent that takes in the data and maps it in a format suitable to store in the context broker.

Often, a buffer exists between the context broker database and the agent to allow for a scalable processing pipeline. When sensor data comes in, the agent updates a few entities at once to reflect what has happened:

1. If the device is unknown but can be auto provisioned, the agent will create one or more device entities containing all that is known about the device and installed sensors and one or more observed entities to keep track of what is actually being measured.
2. When the devices and observed entities are ready, the agent stores the measurement values. Depending on the situation it may do much more than that:
    a. Update device attributes related to telemetry,
    b. Fetch calibration parameters and apply them to the measurement,
    c. Mark sensors as faulty when out-of-bound measurements are detected,
    d. etc.

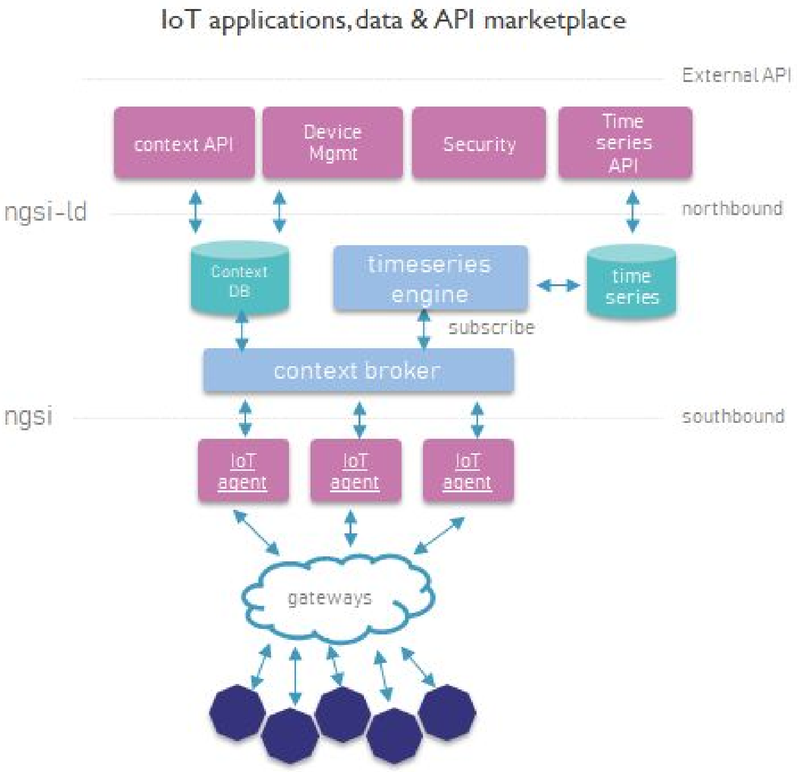


Figure: an NGSI-based IoT stack exposing a context API (REST json-LD), a time series API and featuring the possibility to call webhooks with measurements.

Upon receiving the data, the context broker - depending on the configuration - may call another API for keeping a historical track record of the measurements. This can happen in various ways:

- publishing on a queue
- calling a webhook
- keeping a buffer that can be polled by an external component

The time series database will then offer the possibility to query data based on geographical location, a time window, the sensor producing the data, the type of measurement and possibly the measurements made.

Finally in the northbound, both the context data and the historical data are exposed through REST APIs that allow users to interact with the IoT stack.

2.1.1 Fiware

The different use cases and the alpha version will rely in large part on fiware compliant data providers for IoT data. The fiware project specifies data standards and standard APIs that are designed to

1. Break vertical silos and decompose IoT solutions into independent interoperable components
2. Provide a context information management system that can be used to
    a. Enrich measurement data and help understand what is truly going on
    b. Provision, manage, control and monitor city IoT infrastructure
    c. Expose the actual state of the city via APIs
3. Define a common information model for IoT data
4. Allow the development of interoperable city apps that can be deployed around the world and create an ecosystem of smart city application providers and users

The fiware project identifies the following independent components in IoT stacks:

- Devices: sensors, actuators and combinations thereof as well as gateway devices
- Agents: Software components that can communicate with the devices to collect measurements from sensors and to send commands to actuators. Agents transform the data into an NGSI-compliant format and use the context API to send the data to the context information management system (aka. the context broker)
- Context broker: A database that is aware of devices and what they can measure that maintains an actual state. Typically devices and their metadata are provisioned here in order to keep track of them through their state. The state can be manipulated through the API (e.g., in order to send a command to a device actuator) or by the latest measurement that came in via an agent.
- Time series database (TSDB): A time series database typically records state changes of devices and their measurements for future analysis. In NGSI, the context broke API allows TSDBs to subscribe to state changes for this purpose. The TSDB API provides several ways of accessing the historical data.

All these components communicate across standardized APIs and a common information model (NGSIv-2 - see http://fiware.github.io/specifications/ngsiv2/latest/). The IoT stack in the figure above complies with this architecture.

Aside from the standard APIs and the common information model, fiware also specifies several IoT-domain-specific standards on top of the information model.
See https://www.fiware.org/developers/smart-data-models/ for an overview.

NGSI-LD

The drive towards more interoperability and the use of open linked data in Europe has resulted in a new variant of NGSI. NGSI-LD (LD standing for Linked Data) is being specified by the ETSI Industry Specification Group (ISG) for Cross Cutting Context Information Management (CIM).
The NGSI-LD standard combines the ngsi-v2 context and tsdb specifications into one specification where the common information model is grounded in RDF. This aims to increase interoperability of the resulting data sets.

Advantages of NGSI

- Separating IoT stacks into loosely coupled interoperable components is a good idea. Even if some components are not compliant to the NGSI standards it is typically sufficient to write an adapter to make it fit. The basic principle of separation of concerns is already being applied in the market with success.
- NGSI presents a common information model to exchange IoT data in three forms: event data to be transmitted across APIs, context data available upon request, time series data for storing and retrieving batches of historical IoT data. Even when system components do not comply with the interface definitions, it is quite easy to provide adaptation layers to existing systems that conform to the common information model. This again allows providers to transition to a compliant architecture daily easily.

Disadvantages of NGSI

- At this point in time no real mature implementations exist for the NGSI standards, which makes it hard to integrate NGSI at the core of a robust IoT architecture.
- The information model, although extensible, seems to lack some necessary constructs. The idea of device composition, the notion of calibration and management feathers for provisioning and monitoring are missing.
- The specification lacks a security model which makes it hard to deploy such solutions in a multi-party context.
- There are no structures in the specification to define derived entity types. It is unclear if there are best practices for dealing with derived data streams such as raw data and its calibrated counterpart.
- The standard is work in progress and is expected to evolve significantly.

Since some of the data sources will come from NGSI-compliant systems DUET will benefit from having NGSI ingest receptors.

2.1.2 SynchroniCity

The goal of the OASC SynchroniCityproject is to establish and validate a framework for IoT and AI enabled services. It relies on MIMs (minimal interoperability mechanisms) to facilitate solutions from data collection to data usage through standardized APIs and common information models. SynchroniCity wants to open up a global market where cities and businesses develop IoT- and AI-enabled services through pilots to improve the lives of citizens and grow local economies.

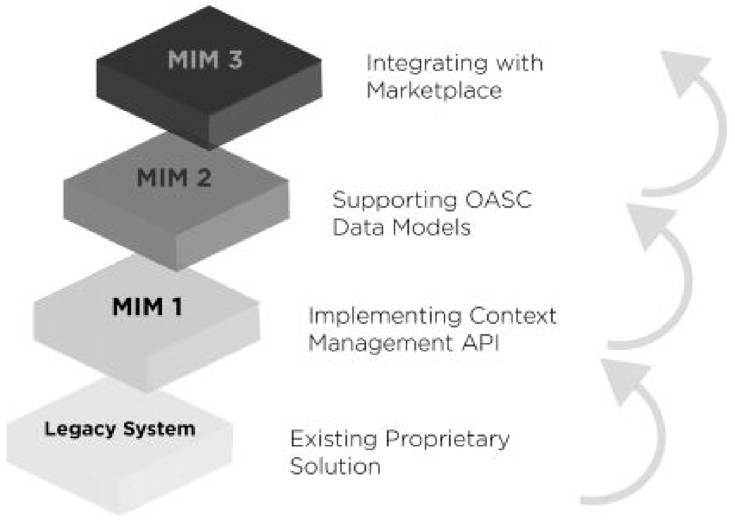

MIM 3
Integrating with Marketplace
MIM 2
Supporting OASC Data Models
MIM 1
Implementing Context Management API
Legacy System
Existing Proprietary Solution

Figure: Minimal Interoperability Mechanisms (MIMs) are universal tools for achieving interoperability of data, systems, and services between cities and suppliers around the world.

As the table below illustrates, the common data models and standard APIs defined by NGSI fir into the SYnchroniCity framework.

MIM	Name	Standards / Baselines	Reference
1	OASC Context Information Management MIM	ETSI NGSI-LD API , OMA 1 NGSI, ITU- T SG20/FG-DPM FIWARE NGSI	Reference Architecture for IoT-Enabled Smart Cities https://synchronicity-iot.eu/wp-content/uploads/2018/09/SynchroniCity_D2.10.pdf
2	OASC Data Models MIM	SAREF, FIWARE, GSMA, schema.org, SynchroniCity RZ + partner data models	Guidelines for the definition of OASC Shared Data Models (SC-D2.2) Catalogue of OASC Shared Data Models for Smart City domains (SC-D2.3; to be released) https://synchronicity-iot.eu/wp-content/uploads/2018/05/synchronicity_d2_2_guidelines_for_the_definition_of_oasc_shared_data_models.pdf
3	OASC Ecosystem Transaction Management MIM	TM Forum Business Ecosystem API, FIWARE Business Ecosystem and Marketplace Enabler API, SynchroniCity API	Basic Data Marketplace Enablers (SC-D2.4) https://synchronicity-iot.eu/wp-content/uploads/2018/09/SynchroniCity_D2.4.pdf Guidelines for the integration of IoT devices in OASC compliant platforms (SC-D2.6) https://synchronicity-iot.eu/wp-content/uploads/2018/09/SynchroniCity_D2.6.pdf

The primary goal of digital twin platforms such as DUET is to provide central access points to interoperable data sources to feed models and visualization services. This aligns to a great extent with the problem domain that SynchroniCity is addressing. It is as such important to take into account all the guidelines, recommendations and good practices proposed by the project.

2.1.3 DUET and Open Data

As discussed in the European Data Portal's Analytical Report 6: Open Data in Cities 2 [14], more and more data sets are being published as Open Data. Most of these data sets are coming from cities and regional or national authorities. The growing amount of open data sets is encouraging but many more needs to be done to tap into the full potential of data and digital twins.

Digital twins are data hungry systems that require high volumes of high quality data in order to gain new insights from predictions and simulations. The availability of such data is far from straightforward. When looking for data, scientists and other data users typically face the following challenges:

- **Discoverability**: Data sources may exist but finding them may be an issue. DUET must provide a (federated) data catalog that makes datasources discoverable by indexing all of their properties.
- **Availability**: Data sources may not always be available at the time they are needed. DUET can be configured to buffer certain data (sets) to ensure availability if needed.
- **Usability/quality**: The necessary meta-data for assessing if the data is usable for the task at hand is not always available. Data properties such as the tools used to collect the data, data resolution and quality are not always clearly stated. Also licensing and fair use of the data may not be defined. DUET will provide a standards based (DCAT) data catalog that allows us to track all kinds of data set attributes.
- **Interoperability**:
    - technical: Technical interoperability as proposed by SynchroniCity is a great asset to digital twins. However, many data providers will not have the capability to quickly deliver standardized APIs. The DUET receptor architecture will provide generic communication patterns (pub/sub, request/response) and common protocols HTTP REST, webhooks, MQTT, sFTP, ··· to onboard data.
    - semantic: Despite the increased popularity of common information models and open linked data DUET will have to be able to onboard other data sources as well. Below we discuss how DUET addresses this issue by means of an internal unified ontology and mappings.
- **Stable identifiers**: A key purpose of digital twins is to be able to combine data sets. This requires IoT data or context data to refer to other resources e.g., a certain Point of Interest defined in a geo service. If the identifiers that other datasets are pointing to can change, then this poses serious integration challenges. If identifiers can be different or change over time, a reconciliation service can be used. See https://reconciliation-api.github.io/specs/latest/
- **Interlinking**: It is equally beneficial that IoT data sources also refer to other resources whenever relevant. This allows for a smoother integration of data in the models. Many models often match datasets based on geography, but it can be a challenge to do this efficiently and reliably.

Open Linked Data is more than just 'open data'. It holds value by making sure that the data can be integrated in a durable and systematic way. Data sources that are not discoverable, do not expose stable identifiers, are not semantically defined and/or do not point to other resources are of limited value for digital twins.

2.2 IoT Live Events

Some digital twin scenarios require the influx of live data. This means that data coming from actual sensors deployed in the field needs to enter the twin with a reasonable latency to reflect the actual state of the urban area covered.

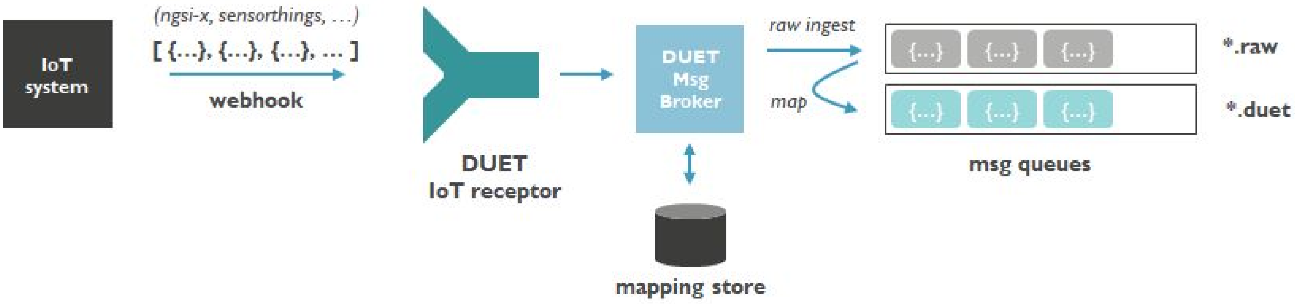
There are two strategies than can achieve this:

1. Subscribing: DUET acts as a subscriber and data is pushed in by calling a DUET webhook using proper parameters
2. Polling: DUET actively polls a preconfigured external endpoint to fetch measurements at regular time intervals

DUET will support both scenarios.

Below a basic architecture is shown for the subscription scenario. The polling scenario is very similar. The DUET IoT event receptor exposes a webhook that can be called by the client to submit IoT events.

2.2.1 Processing pipeline



The events are processed as follows:

1. The receptor webhook is called with the following parameters/data
   a. credentials
   b. the data source ID - created when the IoT event source is registered (see below)
   c. the measurement payload (one or more measurements)
2. The receptor passes the data on to the Duet Message Broker (via the Message Gateway) that applies basic validation and verifies proper access. When all checks are passed, the data is published on a queue (kafka topic) for further processing. This queue contains the raw data - as submitted and not necessarily compliant with the internal ontology.
3. From the raw data topic, messages get picked up again and mapped according to a predefined mapping to the internal ontology. The mapped data is re-published on a queue that is conforming to the ontology.
4. From there, the data consumers can use the data at will as long as they have the proper access rights.

2.2.2 Registering & Configuring the endpoint

Like all other data sources, the IoT live event data source needs to be registered with the DUET Data Catalog. Source registration causes a unique ID to be assigned to the source that can be used to send in data. Registration is done by providing the following information:

- [dc:title] Data source name: a logical name that makes sense for users
- [dct:type] Data source type: a data type selected by the ontology matching the type of data, for instance AirQualityObserved or TrafficFlowObserved

- Data source mapping: A mapping selected from the registered mappings that maps the data source's native format to an internal ontology
- [DCAT properties?] Other parameters: rate limitation, timeout, request size limitation, etc.
- Metadata: DCAT metadata properties for registration in the data catalog, cfr data catalog section

For Polling support:

- Polling URL for the IoT event provider
- Other parameters: polling interval, timeout, response size limitations, etc.

Note:
We will build upon the DCAT vocabulary, extending it with properties as needed.

2.3 Time series data

The biggest difference between historical IoT data sources and live IoT data sources is the flow of the data itself. Time series data sources return data upon request. In order to support this in a generic way, the target data source must understand how the DUET time series connector requests information. This requires implementing a generic interface.

Such generic interfaces already exist. A good example are the NGSI-V2 and NGSI-LD time series APIs. DUET will support at least one of these, but preferably both out of the box. Making a time series database DUET compatible then boils down to implementing a few time series API endpoints.

Other connectors may also be implemented, for supporting SQL databases, CSV data dumps and linked data fragments.

2.3.1 DUET Query language

When a DUET data consumer requests data from a historical IoT data source, this is done using a simplified query interface. This query language or API permits to query data based on geographical location, a time window, the sensor producing the data, the type of measurement and possibly the measurements made.

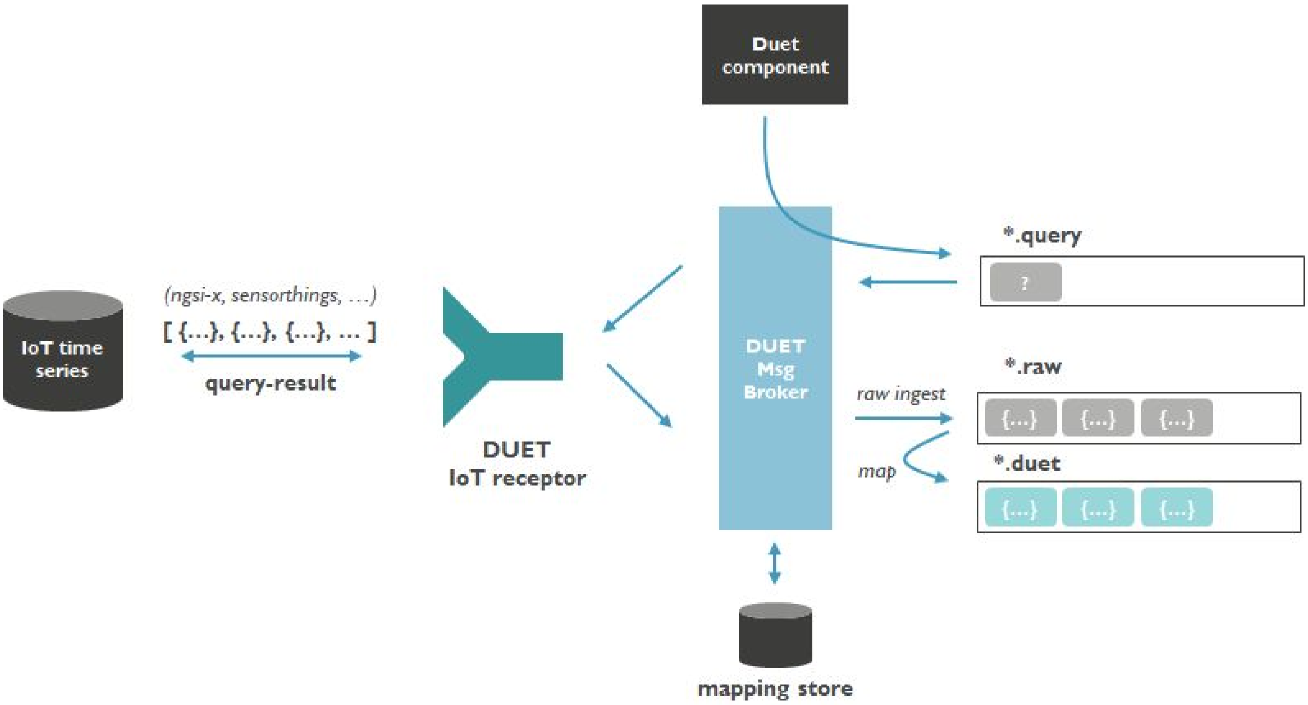
The specifics of this language remain to be specified in a later version of this document. Such queries will contain:

- the data source ID
- the query itself
- some query parameters such as paging and result size limitations
- a timeout for getting the result

2.3.2 Time Series Query Processing

- The DUET data catalog will receive a query and publish it on a query queue associated with the matching data source ID.
- The time series connector picks up the query and sends it to the target system.
- The result is returned and published on the raw data queue of the broker via the DUET message Gateway

- Subsequently, the data elements are mapped onto the internal DUET ontology using a predefined mapping.
- The resulting data is re-published on another queue



2.4 Context data

As previously established, many IoT stacks will keep track of the IoT context in a context broker. These brokers typically expose their data via an API that permits to do some querying on them.

At the point of writing, it remains unclear if being able to connect such APIs to DUET adds sufficient value to be considered and if so, how this can be done in a generic way. It may for instance be sufficient to allow the fetching of individual context entities using a single ID.

2.5 Geographical data

The strategy for wiring up geographical data will be very similar to that of time series data: connectors supporting specific standard services may be provided out of the box. Examples of such standards are: CityGML, GeoSPARQL, GeoAPI/GeoJSON, WFS, ...

The query processing and data flows are similar, including publishing the raw results and mapping them onto the internal ontology.

2.6 Data catalog

All data sources pushing one of the above categories of data will be registered in a central component called the DUET Data Catalog. The DUET data catalog uses the DCAT data model and vocabulary as a starting point for the alpha version to index all registered data sources.

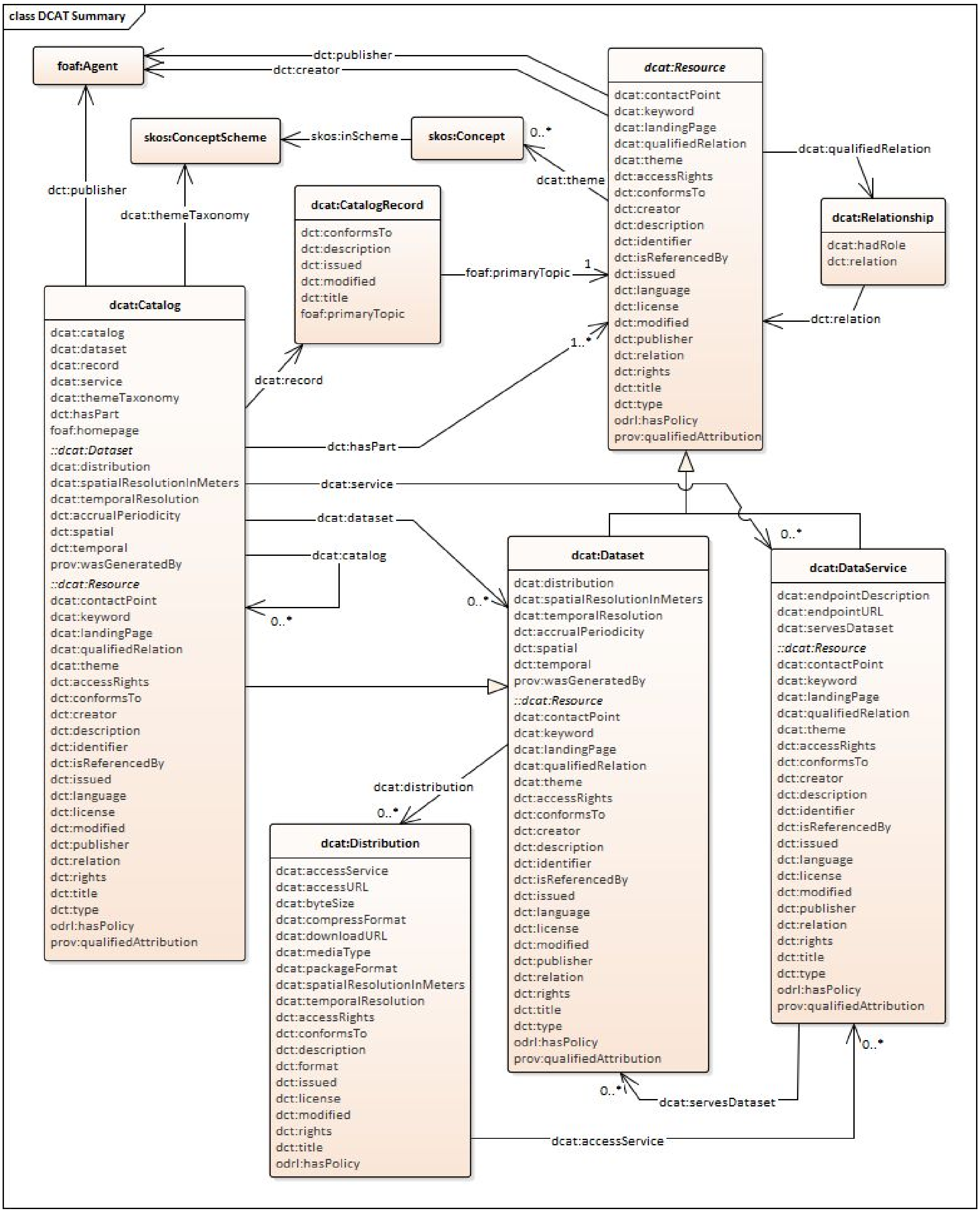


Figure: The data model for [DCAT - Data Catalog Vocabulary (DCAT) - Version 2]

The following DCAT attributes are deemed particularly relevant for the implementation in DUET:

- Homepage
- Themes: referring to smart city domains
- Accessrights
- Conforms to: reference to a concept in the knowledge graph
- Contact point
- Description

- Title
- Spatial & temporal:
    - frequency
    - spatial resolution
    - frequency
    - temporal coverage
    - temporal resolution
    - generated by
- Quality
    - Reliability of the data according to some scoring
    - Pointer to calibration process (if any)

2.6.1 Data Registration API

Register Data Source

POST

/[version]/catalog/data

parameters (none)

payload:

- Receptor type: the correct receptor to deal with the provided service
- Receptor parameters (see above)
- DCAT based description of the data source

returns

- Resource including ID

List Data Sources

GET

/[version]/catalog/data

parameters:

- limit & offset (pagination)
- search (textual search)
- any (dcat) property for filtering

returns

- List of resources

Get Data Source

GET

/[version]/catalog/data/[data-source-id]

parameters (none)

returns

- Resource with the ID

Update Data Source

PUT

/[version]/catalog/data/[data-source-id]

parameters (none)

payload:

- resource with all properties

returns

- Updated Resource

update Data Source

PATCH

/[version]/catalog/data/[data-source-id]

parameters (none)

payload:

- resource with all properties you want to update

returns

- Updated Resource

Unregister Data Source

DELETE

/[version]/catalog/datasource/[data-source-id]

parameters (none)

returns (none)

2.6.2 Data Query API

Time series & Geo - Time series & geo queries work by sending a query to the data source. Upon submitting the query, the requesting component receives a correlation ID that allows it to retrieve the query results from the broker. The query is processed asynchronously and the results are published on the internal broker.

Send a query a time series or geo data source

POST

/[version]/catalog/data/[data-source-id]/query

parameters:

- limit & offset (pagination)

payload:

- A query object (see below)

returns

- Correlation ID: An ID that allows the client to get the results

Query object

The query object includes (among other things)

- entityId: the id of the entity or list of entity ids. There will be a mapping from the internal ids to the federated query parameter ids
- type: comma separated list of entity types whose data are to be included in the response
- from, to: start and end of the query interval, specified in ISO-8601 format.
- bbox or polygon: string, geospatial index from which data should originate

Fetch query results from a time series / geo data source

GET

/[version]/catalog/data/query-result/[correlation-id]

To be specified further. The idea is that a client can 'consume' the result as a stream.

Subscribe to an IoT event data source

POST

/[version]/catalog/data/subscribe/[data-source-id]

parameters:

payload: a subscription object including:

- entityId: the id of the entity. There will be a mapping from the internal ids to the federated query parameter ids
- type: comma separated list of entity types whose data are to be included in the response
- Any 'property'='value' query parameter, e.g., device=<id> to filter observations made by a certain device.

returns

- A list of entities of a certain type

Get an entity from an IoT context data source

We consider context data sources to provide context information in the form of a few first-class citizens, listed below. A list of semantically well-defined properties can be associated to each of these entities.

1. Things: Real-world entities to which measurements can be associated. They can have any number of properties and are identified by stable IDs.
2. Devices: A Device is a technical component used for measuring or actuation. **Sensors** are Devices used for sensing and/or measuring. Whatever is being measured directly is a property of that device (actual state). **Actuators** are devices used for controlling things. Whatever is being controlled is a property of the device.
3. Device boxes: A device box contains one or more devices. They have an optional location, since some DeviceBoxes can be attached to moving objects. In that case the Observations output by the Devices shall have a location.
4. Observations: Denotes observations such as water quality and air quality.
5. Locations: by using the schema.org type for GeoCoordinates, it isn't possible to use a reference coordinate system, since that is defined as using WGS-84. This type can however be extended to, for example, have an originalLocation property that does have a reference coordinate system, and that is defined in our own ontology.

This permits us to define the following generic API where entity-type is one of the above first-class citizens.

GET

/[version]/catalog/data/[data-source-id]/{device, devicebox, observation, thing, location}

parameters:

- limit & offset (pagination)
- bbox or polygon: string, geospatial index from which data should originate
- Any 'property'='value' query parameter, e.g., device=<id> to filter observations made by a certain device. Using the <field>_from and <field>_to attribute to filter on datetime fields

payload: (none)

returns: A list of entities of a certain type

GET

/[version]/catalog/data/[data-source-id]/{device, devicebox, observation, thing, location}/[entity-id]

parameters: (none)
payload: (none)
returns: The matching entity if it exists

2.7 Knowledge Graph

The knowledge graph is an essential component of the DUET system. It makes the information offered through the data APIs searchable for both systems and humans. It also allows to create a navigable structure on top of registered data sources.

The knowledge graph will contain a formal semantic definition of data offered through the registered data sources and as such may assist in providing validation components to ensure that published data conforms to these definitions.

Note:
As opposed to some interpretations of the concept of a knowledge graph, the data itself will not be stored internally in the DUET knowledge graph. Abstraction layers such as GraphQL may be provided to transparently query the knowledge graph and underlying data sources. Cfr Linked Data Fragments & communica concepts.

2.7.1 High-level functional roles of the knowledge graph

- A searchable & navigatable data catalog
    a. Definition of smart city domains (see below)
    b. Description of semantic concepts within domains
        i. Vocabulary
        ii. Implementation model
    c. Data source registry, providing for each data source
        i. A data source type (IoT events, IoT time series, IoT context, Geo data, ⋯)
        ii. Association of the data source with a smart city domain and concept
        iii. Support for data catalog standard DCAT and addition of metadata
- Data validation component ensuring structural and semantic correctness of incoming data
    a. Allow the different APIs to validate data using the knowledge graph
    b. Semantic validation

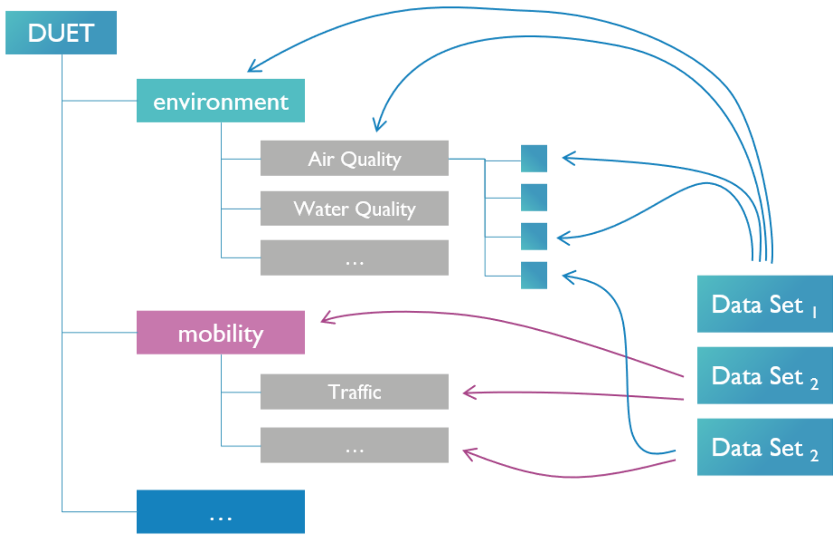


Figure: Data sets in the data catalog belong to one or more smart city domains and map onto a semantic definition in the knowledge graph.

2.7.2 Definition of smart city domains

The knowledge graph contains an ontology of all concepts known to the digital twin. For the purpose of overview, these concepts will be grouped into the smart city domains.

DUET will define a default set of smart city domains that can be governed by its administrators. These serve as chapters for the platform and can be used to navigate through available data sources. Additionally, these can be used to refine searches in the knowledge graph.

Potential smart city domains:

- Economy
- Government & regulation
- Environment
- Energy
- Education
- Public space
- Health & living
- Mobility
- Safety & security

2.7.3 Km4City

The Km4City research project started in the 2013 with a generic name, smart city ontology, it was named Km4City later. The information model at that time was focussed on the Open Data of Florence Municipality including POIs and street maps from Tuscany. The Km4City (https://www.km4city.org/) project aims to build a central platform that groups smart city data sets, services and applications under a unified API and a common information model with a heavy focus on a unified ontology. This data-first approach aligns with DUET which also proposes a semantic standardization for smart city data through a unified ontology.

The Km4City ontology (https://www.km4city.org/img/Km4City-v1-6-4.svg) can be inspirational to the DUET project.

2.8 Mapping & validating data

2.8.1 Defining mappings

Mappings are registered in the DUET catalog. What kind of mappings DUET will support out-of-the-box will depend on the mappings needed for the pilots.

Note:
The internal data-representation in DUET is json/json-ld, so any mapping should produce json(-ld). Some mappings may preserve semantics but transform only the structure, e.g., RDF-to-json, XML-to-json, ...

As far as we can determine at the point of this writing, the following mappings will be supported:

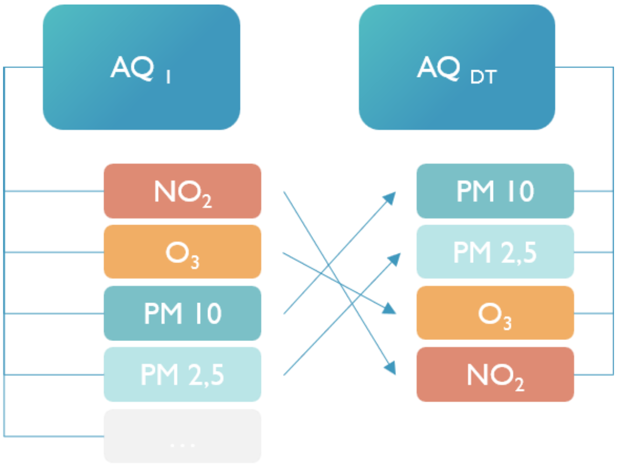
- Ontology based mapping where semantic properties are mapped onto each other. This is a simple form of flat data mapping assuming that the data contains no complex structures;
- Code-based mapping: a mapping based on running custom code (e.g., nodejs) in a sandbox to map data elements;
- CSV mapping where table-formatted data is mapped according to a mapping either based on column index or based on column names if the first row contains column headers.

2.8.2 Ontology based mappings

The concept of ontology-based mappings starts from the observation that often overlapping standards are used for the same purpose. In the context of DUET where combining data sources is key, the ability to map data across these definitions is key.

See also RML - RDF Mapping Language

Figure Structure-preserving mappings that allow to map one semantic definition to another can be simple and effective.



2.8.3 Other mappings

Other more complicated mappings can be necessary. Plenty of mapping languages and standards exist to help there. In the end, when needing transformations that require complex logic, the question is if this is still the responsibility of DUET to handle that.

Technologies that can be considered are:

- XSLT
- RDF Mapping Language (RML)
- Sandboxed nodejs

2.8.4 Validation

The knowledge graph provides a validation function for each data source type to verify accordance with any specified vocabularies.

Validation needs to happen after mapping the data to the internal data model. At the point of this writing it is unclear what the outcome should be if data is not valid.

It needs to be discussed if the following optional validations will be supported:

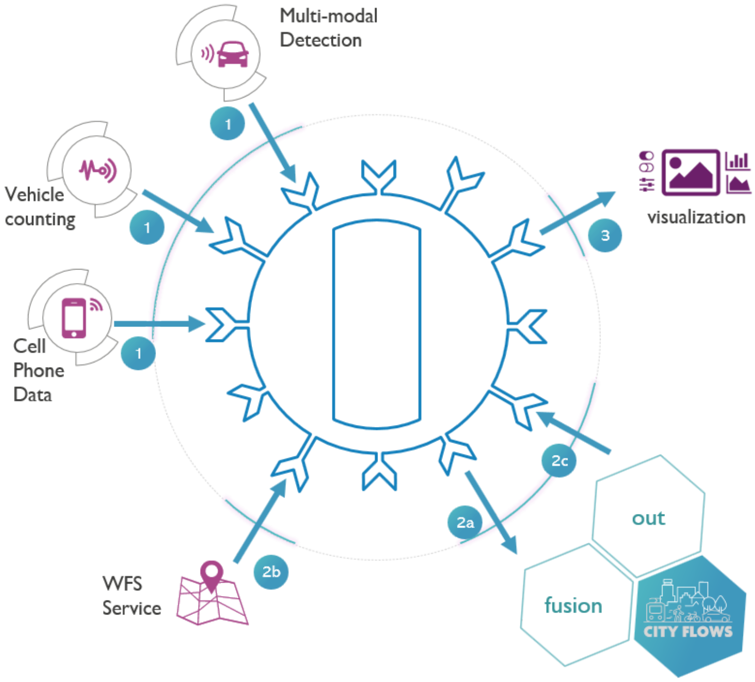
- Strict: only types defined in or imported into the DUET vocabulary are supported
- Complete: validation of completeness of certain data, i.e., that some properties are required for certain data sources
- Well formed: minimal level of compliance to ensure that correct literals and types are used in accordance with semantic specifications + required properties (refDevice, ...)

3. Use case data flows

3.1 City flows (alpha version)

The alpha version serves the purpose to demonstrate the soundness of the chosen approach and architecture. Following limitations and scoping apply to the alpha version:

- Geographical scope: Antwerp Smart Zone
- Data sources:
    - Proximus crowd management
    - Citymesh wifi scanning
    - Telraam traffic counting
    - Flanders road register
- Model: city flows model
- Visualization: Cityflows digital twin visualization



The city flows data sources will be mapped from their delivery formats onto the ngsi-v2 (later ngsi-ld) format and onboarded into the imec city of thing IoT platform that will allow to connect generically to DUET via an ngsi receptor component.

3.2 Flanders Pilot

More details on the data flows will be given as the pilot progresses.
For more information on the pilot, please visit deliverable 3.3

3.3 Pilzen Pilot

More details on the data flows will be given as the pilot progresses.
For more information on the pilot, please visit deliverable 3.3

3.4 Athens Pilot

More details on the data flows will be given as the pilot progresses.
For more information on the pilot, please visit deliverable 3.3

4. Conclusion

We have presented a conceptual architecture to onboard IoT data sources and related data sources in a scalable way into the DUET broker. This conceptual architecture builds upon prior experiences with building digital twins and tries to generalize the principles used to accommodate more diverse data sources.

In the first phase of the project, the focus will be on accommodating data coming from IoT stacks, notably NGSI-v2 compliant stacks for validating the concepts.

- We discuss three types of data (i) IoT event data (ii) IoT historical data and (iii) Geographical data
    - IoT event data for the alpha version (citflows) will be onboarded through a fiware (NGSI-v2) adapter. The data will come from an NGSI-v2 compliant context broker implemented by imec.
    - IoT historical data for the alpha versions will come from a time series database that contains the historical data for the data source mentioned below. The alpha version will include a way for DUET components to query data from that source.
    - Geographical data receptors are out of scope for the alpha version and will be discussed in subsequent versions of this deliverable
- We describe how typical IoT stacks can be wired up with DUET conceptually, more specifically for time series data and context data
- We also discuss how this data can be mapped to a unified internal data format for integration purposes at a conceptual level. A more technical discussion will follow in subsequent versions of this document.
- We introduce the DUET data catalog and knowledge graph that provide the necessary structure and handles for integrating the data.
- We conceptually discuss the potential to use the knowledge graph for validation of incoming data flows.

As discussed above in section 1.2 Roadmap, many technical details still have to be filled in. The main goal of the alpha version is to validate the data flow system. This will be done by supporting historical and live IoT data and feeding it to a model through the system DUET receptors. The aLpha version is due November 2020.

5. References

1. DUET - Digital European Urban Twins
https://www.digitalurbantwins.com/

2. Cityflows - one view on multimodal flows in the city
https://www.imeccityofthings.be/en/projecten/cityflows

3. FIWARE - The Open Source Platform for Our Smart Digital Future
https://www.fiware.org/

4. OGC - Open Geospatial Consortium
https://www.ogc.org/
Standards: https://www.ogc.org/docs/is

5. NGSI standards
NGSI-v2: https://fiware.github.io/specifications/ngsiv2/stable/
NGSI-LD:
ttps://www.etsi.org/deliver/etsi_gs/CIM/001_099/009/01.01.01_60/gs_CIM009v010101p.pdf

6. DCAT - Data Catalog Vocabulary (DCAT) - Version 2
https://www.w3.org/TR/vocab-dcat-2/

7. RDF - Resource Description Framework
https://www.w3.org/RDF/

8. RML - RDF Mapping Language
https://rml.io/specs/rml/

9. Synchronicity
https://synchronicity-iot.eu/tech/

10. EIF - The New European Interoperability Framework
https://ec.europa.eu/isa2/eif_en

11. OASC Minimal Interoperability Mechanisms (MIMs)
https://oascities.org/wp-content/uploads/2019/06/OASC-MIMs.pdf

12. European Data Portal‘s Analytical Report 6: Open Data in Cities 2
https://www.europeandataportal.eu/sites/default/files/edp_analytical_report_n6_-_open_data_in_cities_2_-_final-clean.pdf