Deliverable

D3.8 Digital Twin data broker specification and Tools v1

Project Acronym:	DUET	
Project title:	Digital Urban European Twins	
Grant Agreement No.	870697	
Website:	www.digitalurbantwins.eu	
Version:	1.0	
Date:	November 30, 2020	
Responsible Partner:	imec	
Contributing Partners:	AIV ATC	
Reviewers:	**Internal** Thomas Adolphi (VCS) Hans Cornelissen (TNO) **External** Pieter Morlion Yannis Charalabidis Andrew Stott	
Dissemination Level:	**Confidential**	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	09.10.2020	Philippe Michiels	imec	Seeding
0.2	13.11.2020	Dwight Van Lancker	AIV	OSLO tools and process
0.3	16.11.2020	Bert Van Nuffelen	AIV	Semantics, data model, CKAN
0.4	16.11.2020	Dries Staelens	imec	Mapping
0.5	17.11.2020	Philippe Michiels	imec	architecture, components descriptions
0.6	18.11.2020	Philippe Michiels	imec	Functional requirements
0.7	19.11.2020	Philippe Michiels, Koen Triangle, Bert Van Nuffelen, Dwight Van Lancker, Dries Staelens	imec, AIV	Finalizing for review
0.8	20.11.2020	Thomas Adolphi, Gert Vervaet	VCS, AIV	internal review and minor changes
0.9	24.11.2020 26.11.2020	Pieter Morlion Yannis Charalabidis	external reviewers	review
1.0	30.11.2020	Philippe Michiels	imec	final version

Editorial notes

This is the first version of the specification. A lot of work remains to be done and many things are still not entirely clear at the point of writing this version of the deliverable.

The term model has been used ambiguously throughout the project, its broad definition allows for this:

'a model is an informative representation of an object, person or system' [1]. In this deliverable we need to differentiate between **data models** and **simulation models** both of which are key elements of the DUET architecture.

A **simulation model** is a particular kind of mathematical model. It is conceived with the goal of emulating and understanding the behavior of a real-life system through software. Examples in the context of DUET are the air quality-, traffic- and noise emission models.

The term **data model** can be defined as an abstract model that organizes elements of data and standardizes how they relate to one another and to the properties of real-world entities[2].

The reader may have to glean from context which of these is applicable when we refer to the term 'model', although we should use the terms data- and simulation model explicitly if necessary.

[1] https://en.wikipedia.org/wiki/Model

[2] https://en.wikipedia.org/wiki/Data_model

Table of Contents

Executive Summary

The DUET project creates a digital twin platform for urban regions that transcends the scope of a single use case. This implies that DUET should support the following:

- Adding new (possibly third-party) data sources to an existing digital twin case
- Adding simulation models to an existing case for comparison
- Adding a simulation model to an existing case to serve as (additional) input for another model
- Adding visualization/interaction clients to enrich the reporting for a case
- Extending the digital twin ontology to support more city domains or expand existing ones

The DUET-Cell[3] architecture is designed as a plug-in interface to support all these features. The DUET data broker corresponds to the DUET-Cell. It is shielded by APIs that allow the components to connect to the T-Cell's internal *message streaming system* on which all data flows between the different components.

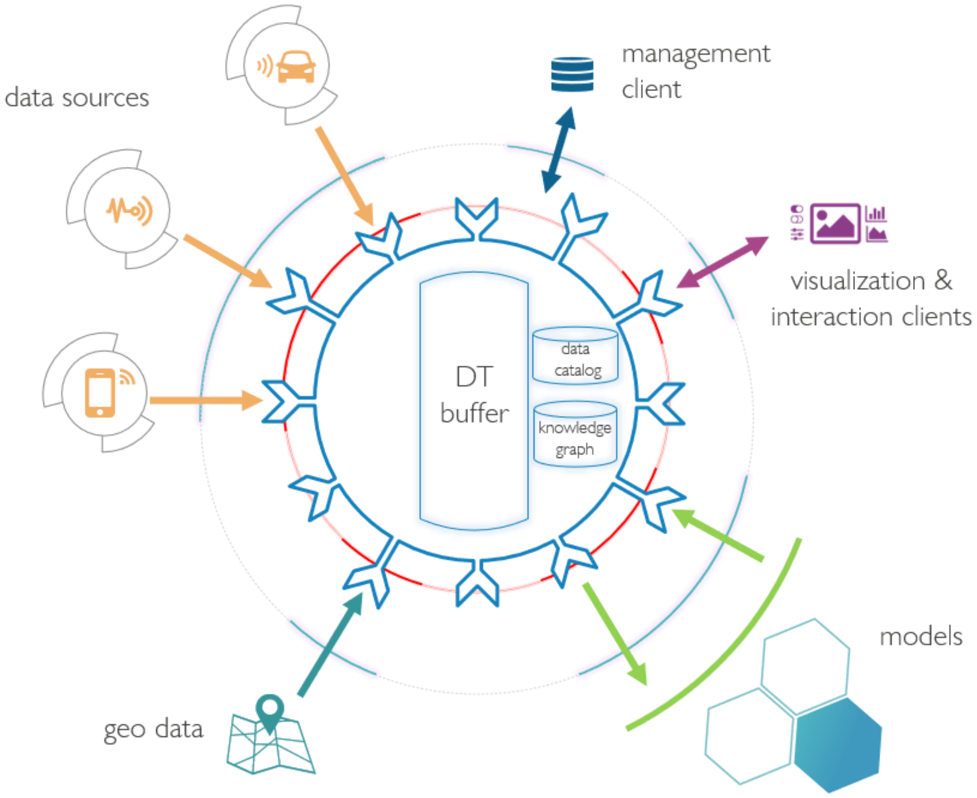


Figure: The DUET T-cell acts as a databroker connecting the data sources to the different DUET components.

The Digital Twin Data *message streaming system* lies at the core of the DUET T-cell and facilitates data streams between all components. The streaming features of the *message streaming system* allows

[3] Our conceptual architecture resembles a T-Cell lymphocyte shape, hence the name. See https://en.wikipedia.org/wiki/T_cell.

components to subscribe to data events. It also facilitates quick access to data in the case of responsive digital twin scenarios where data needs to be kept ready for use in models and/or visualizations.

One of the most essential tasks of a digital twin system such as DUET is to enable the combination of different data sources to use them in (simulation) models or visualizations.
More often than not, data sources are not fully compliant with known standards. And even if they do, it may not be the standard the user was hoping for, but a competing one. Thus, integrating data sources from different suppliers remains non-trivial. The DUET architecture should allow use case designers and users to deal with that. The proposed approach is to map data to a common language, called the DUET ontology (see below). Onboarded data needs to be mapped to that ontology at the entrance of the platform.

The DUET common ontology is stored in the knowledge graph. It reflects DUET's understanding of the smart city and its domains. This ontology will be largely inspired by existing common standards of course. But it can be extended when needed to support specific cases and scenarios.

In order to connect different models to DUET, we need mechanisms[4] to control these models from DUET on the one hand and multiple data APIs that allow these models to consume DUET data on the other. The latter are an integral part of the DUET data broker API. This API is discussed in more detail in Deliverable D3.5. Models will also produce output. As such they act as DUET data sources and they should be registered as such to make them available to other models or visualization and interaction clients.

The Data Catalog plays an essential role in registering data sources along with their metadata. This does not imply uploading data, but instead letting the DUET data broker know where to find it. Not only is this essential to make data sources discoverable, the data catalog is also instrumental in determining if a data source is compatible with a certain use case.

Visualisations and interaction clients are the most visible parts of the DUET architecture but they are clients of the DUET data broker. Consuming the data on the one hand and sending back messages (for instance interaction messages that trigger model recomputations).

Implementation Note:
The presented architecture of DUET does not rely on storing entire data sets inside the system. This will in general not even be a possibility when data sets get very large, which is often the case for IoT historical data and geo data. Although it is possible to store (smaller) data sets inside the data catalog, we do not intend to make use of that feature for now.
This implies that the data is being streamed across the platform. The message streaming platform will be used to retain some of data, depending on the retention settings and platform limitations. The retained data can conveniently be used as a cache, for instance to store relatively small results of recurring data queries.
An important implication here is that data sources may not always be available since we federate queries to them. It is important to take that into account when creating your digital twin. This can be mitigated by setting up derived data sources as proxies.

[4] The agent APIs referred to here are API specs that enable the DUET data broker to interact with models. It is up to the model provider to implement their behavior. That is, unless some standard way of model orchestration can be applied, for instance using Kubernetes.

1. Introduction

1.1 Objectives

The objectives of this deliverable are to:

- Define what a digital twin data broker is from the viewpoint of DUET as an urban digital twin platform
- Identify the different components of the data broker in DUET
- List the requirements for the data broker component and provide a high level roadmap for integration in DUET
- Specify a technical architecture for the data broker
- Identify the tools and frameworks we will use for implementing the data broker
- Discuss security from the data broker viewpoint
- Discuss some legal interoperability challenges

The data broker is at the heart of DUET. It's primary purpose is to provide a decoupled architecture that allows external components to easily connect and interact with DUET, achieving a high level of interoperability between data sources, models and digital twin clients.

1.2 Scope

This deliverable does not go into detail on the technical details of the non-broker components. These are discussed in other deliverables:

- **D3.1 IoT stack and API specifications v1** describes the components onboarding of data into DUET
- **D3.3 Smart City domains, models and interaction frameworks v1** describes the models that the different partners can provide to DUET
- **D3.5 Cloud Design for Model Calibration and Simulation** discusses how models can be run in the cloud and connected to DUET, using the potential of HPC infrastructure

This deliverable also does not discuss an overall architecture and implementation or deployment plan. This is discussed in **D5.1 System Architecture & Implementation Plan**.

For a detailed discussion of security matters, we refer to **D3.10 Multi Layered security model specification**.

The scope of the DUET data broker is aimed at establishing a decoupled digital twin architecture that can be used by third parties to integrate their data sources, models, clients and tools with as little effort as possible. Although the proposed architecture can be used to do so, the use of the data broker as a commercial platform to host and monetise data is out of scope.

The DUET project focuses on the smart city areas 'mobility' (subdomain traffic) and 'environment' (subdomain noise and air quality). That does not mean that the DUET system and architecture are not designed with a broader scope in mind but it implies that our propositions for a smart city ontology will mainly focus on those areas. The idea is that the system should eventually package an entire smart city ontology which can then be modified or extended by an administrator. However, this is out of scope for the DUET project. The same holds for other components and extensions: we will mainly focus on the scope of the project and refine & implement features and components on a call-by-need basis.

At the time of writing the 1st version of this deliverable, The Alpha Version is nearing completion. The Alpha Version is aimed at the conceptual validation of the data flows. As such, not all components of the T-Cell have been fully developed and other components are still in their design phase.

This version of the deliverable will also not elaborate on the details of the DUET ontology or specific data standards concerning traffic and environment (air quality, noise, etc.).

The remainder of this document is structured as follows:

- Section 2 defines the data broker concept in the context of an urban digital twin
- Section 3 discusses the different data broker components and their role in the overall design
- Section 4 elaborates on the two most important tools and frameworks used for realising DUET: CKAN and the OSLO toolchain
- Section 5 defines some high level requirements for the DUET data broker
- Section 6 goes into further detail on the technical architecture, the components and their dependencies
- Section 7 Discusses the security architecture and some security related matters for DUET
- Section 8 contains our conclusions

2. Defining the databroker

Data brokers are often defined as business entities that are selling or reselling information, i.e., brokering in data. In our case, a data broker is a subsystem or a component of a technical architecture. More specifically, in the context of a digital twin we define it as follows.

A Digital Twin Data Broker is a subsystem of the digital twin that:

i. Collects metadata for a variety of data sources in a (Data) Catalog in order to make them discoverable and available for models and clients,

ii. Collects metadata for simulation models in a (Model) catalog for shared use,

iii. Exposes programming interfaces (APIs) for simulation models and clients to produce or consume data,

iv. Allows users to share their data sets either publicly or in a more controlled way,

v. Facilitates data cleaning, normalization, and/or enrichment of data to maximize data interoperability,

vi. Provides data broker clearinghouse and market place functionalities that allows to track data transactions which can be conceived as an audit trail that facilitates some form of remuneration or compensation[5]. The DUET marketplace allows users to publish their DUET compliant data sources, models, visualisation and interaction clients, etc. A public-facing interface on top of the different catalogs should exist,

vii. Provides features to verify the legal interoperability of data sets for their use in digital twins and also provides tooling support for GDPR compliance.

Although the latter two points are not essential from a technical point of view, providing such functionalities may be a prerequisite for parties to onboard their data. Furthermore, it may make users aware of the legal implications and limitations of/for using certain datasets.

[5] As put forward by IDSA, see the IDSA whitepaper

3. Data broker components

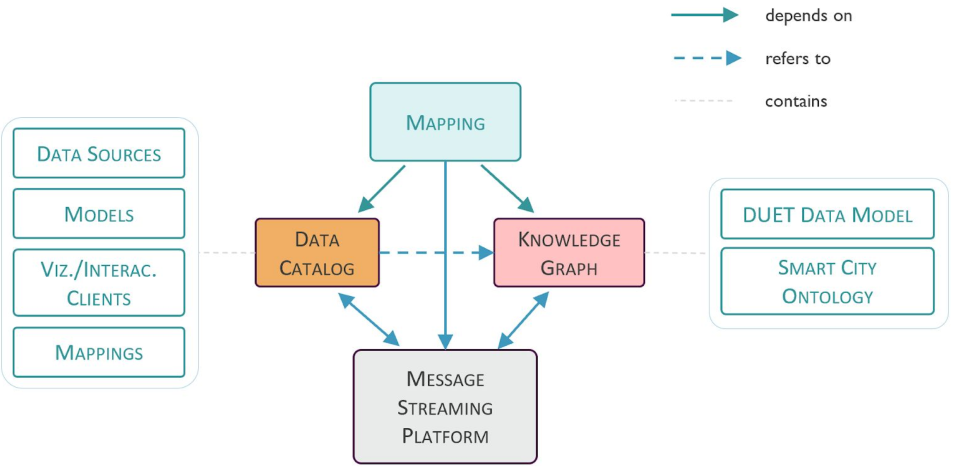


Figure: the core components of the DUET data broker[6].

3.1 Message streaming platform

The Message Streaming Platform is the heart of the DUET Data Broker. It allows the different DUET components to communicate and exchange data in an asynchronous way. Although the DUET components will be implemented in such a way that any relevant Streaming Platform can be used with the appropriate extensions, for the purposes of the project, we are going to use Apache Kafka[7].

Apache Kafka is an open source Message Streaming Platform, with characteristics like stream processing, using a highly scalable architecture, high availability and throughput, as well as a large ecosystem of open source tools around it. All these characteristics make Apache Kafka a suitable option for the DUET project.

3.2 Data catalog

The data catalog is an essential part of the data broker. It connects the data providers with the data users in DUET. A data provider can register a data source in the DUET data catalog by providing descriptive information about the data source (metadata). While supplying this metadata, the actual data streams are initiated in the DUET data bus (i.e., Kafka in our implementation). This registration process creates a data catalog which can be explored by DUET users to feed their modeling and visualization tools.

Data Catalog Features

1. Data source classification

   The system will support several types of data. The following specific data types are considered:

[6] The functionality of the data broker is made available using APIs, grouped in different logical gateways. These are discussed in more detail in the architecture section.

[7] https://kafka.apache.org/ See DUET Deliverable 5.1 for more details.

a. **IoT events** are the measurements coming from sensors entering the system as a stream of (relatively small) data records. The internal broker keeps track of them by placing them on a message queue. Again these records can be mapped to an internal data model in alignment with the central ontology. Historical data can remain available for some time depending on the retention settings for the data source. Users of the data can also subscribe to the data source.

b. **IoT historical data** is offered by time series data sources. They typically offer an API for querying a trace of the history of IoT events. DUET users can query these using a universal interface for querying historical data. The result is that - like for IoT events - the individual records are streamed via a queue to the user requesting the data.

c. **IoT context data** is data that provides context for IoT events. It gives context about measured values such as the kind of sensors involved, its location, calibration parameters, and other context that may be important for interpreting the results or for visualisation purposes. This data is available through a universal context data API and can be mapped to an internal data model.

d. **Geographical and other data** Geographical data sources may also be added to the data catalog. They too can be queried. The role of the data broker here is to facilitate the linking of data sources by their geographical location. E.g., what street is a sensor in, or even, where is a company located? This allows models to relate different data sets and visualizations to display data more accurately.

   Aside from purely geographical data, there may also be other data coming from various sources. Weather forecasts, news feeds, event calendars, ... Again the data broker's role is to allow linking of this data to other data sets.

2. Data source inventory management

   The data catalog should allow CRUD management of data source entries, i.e., metadata and not the data itself by the DUET users. Administrators of the DUET instance will have full control over the data sources. Other users can manage their own data sources. The security system should support security groups to allow shared ownership and management of data sources.

3. Data source discoverability

   The metadata provided with each data source allows users to search for data sources matching their needs. Integrating this metadata with the core DUET knowledge graph, i.e. the data models describing the supplied data by the datasources, creates a powerful knowledge base which can be queried for finding available and compatible data.

4. Data source access management/restriction

   Upon registering the data source, it is up to the publisher to indicate whether the dataset is available to all or a subset of users of the system and with what restrictions. Access can be restricted based on geography, organisation, role. It's probable that using certain datasets will be subject to paying license fees.

   For the time being, DUET will not cover contract management, remuneration schemes or sharing requests. Nor will it consider resolution of legal challenges that would occur from the usage.

5. Handling data source querying & subscriptions

When connecting a system to DUET, that system will use the data catalog to find the access channel for the datas. We distinguish 3 ways how data can reach the connecting system. 2 pass via the DUET data broker. The third option is a direct connection to the data source APIs.

- Event-based: events are flowing in as they are pushed from the source system. Raw data and mapped data (if a mapping is provided) is available in the *message streaming system*. Data retention is controlled by a setting in the data catalog. Requesting the data is transparent: the data API will map the request to the proper topic.

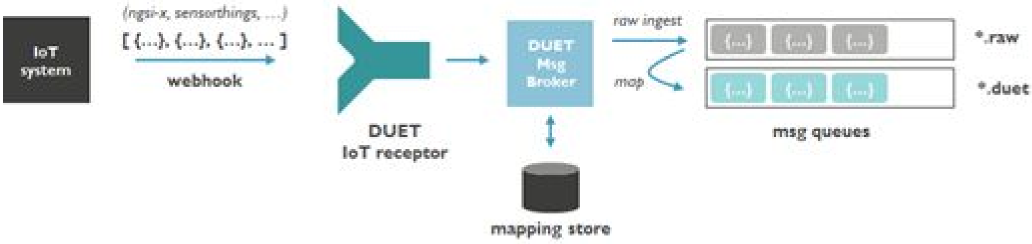


Figure: Event-based data access.

- API based (time series, geo data and other data): The data is pulled onto the platform on call by need. The data catalog resolves the data source details. The data API makes sure the data is published onto the internal *message streaming system* and that the client can get the data by means of a correlation ID.

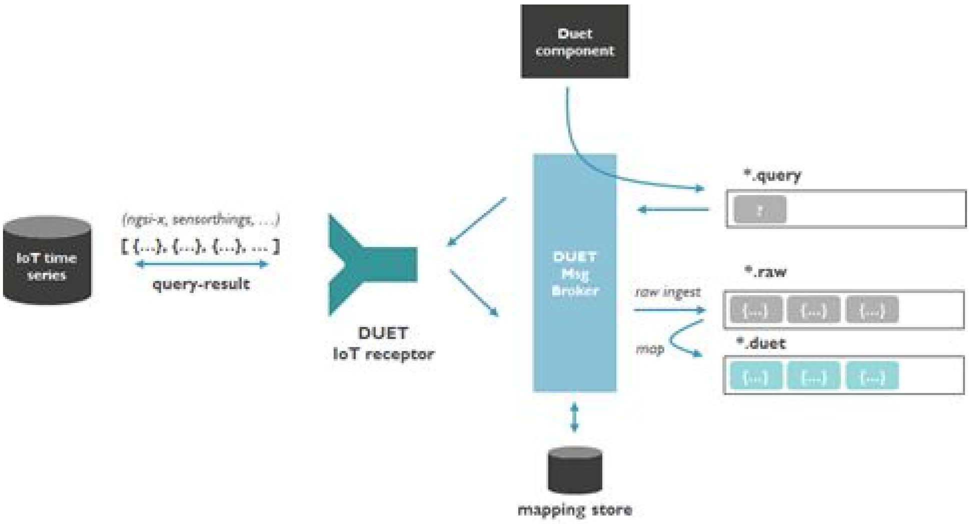


Figure: API based data access

- Direct access: Not all data sources need to be transferred across the internal *message streaming system*. Specifically, data that needs no mapping

<u>Connection with the ontology and the knowledge graph</u>

When users register data sources in the data catalog, there are three possible scenarios that apply:

1) The data source is compliant with the DUET ontology natively. The user can then refer to the ontology (by means of a link to the knowledge graph holding that ontology) to indicate what specific type of data is offered by the source system.

2) The data source is not compliant, the user can provide a mapping specification (see below) to map the source data format onto the DUET ontology. In that case, the target type can be selected from the knowledge graph.
3) the data source does not comply with the DUET ontology natively and will also not be mapped and can be used as is.

Obviously, the first two scenarios are beneficial to the ability of (a) combining different data sources from different providers and (b) reusability of the offered data set.
Note that, the knowledge graph can fulfill a broader role than just holding the DUET ontology. It can also refer back to the data catalog by keeping references to data sources and data publishers. This way, data will become more discoverable for the users.

Mappings
Incoming data will not always comply with the expected structure. There can be a mismatch in multiple ways.

1. The first kind of mismatch occurs when the source system is delivering packages (envelopes) containing one or more messages. The individual messages need to be unpacked and sent on to the internal message queue (topic) one by one.

2. The second kind of transformation is needed when mapping onto the internal data model and ontology (see section 3). This mapping makes sure that the incoming data is projected onto the structure of the internal data model and that proper semantics is applied for the data to match with the DUET ontology.

See section3 on more details concerning mapping.

Data Catalog API
The Data Catalog features impose requirements on its API. The core Data Catalog API consists of the following API calls

feature	*API calls*
Data source inventory management	*CRUD operations for data sources*
Data source discoverability	*query support for structured data* *query support for unstructured data*
Data source access management	*Grant access to a group or individual* *revoke access*
Supporting data source querying & subscriptions	*Resolve a data catalog ID to data source details in order to allow the wire-up with the source system to work*

Data Catalog DCAT model

To describe data sources one has to agree on the vocabulary to be used. Creating catalogs describing the available datasets and their associated resources has been a long standing activity within the public sector information. One of these activities, namely the Open Data movement, initiated a decade ago the creation of a W3C vocabulary called the Data Catalog Vocabulary, in short DCAT. This vocabulary has been adopted via the application profile DCAT-AP[8] as the norm for exchanging dataset descriptions in the European public sector. For instance the European Data Portal[9] federates all Open Data published in Europe.
Recently, early 2020, a new version of DCAT[10] has been published. The major improvement DCAT 2.0 is the addition of data services descriptions. The community had identified this as one of the main weaknesses of the previous version.

By basing the data catalog on DCAT we enable a potential of reusing dataset descriptions that already are made available by the public sector in Europe. Also, it will enable us to identify and describe the additional information requirements that data source providers have to fulfil for registering their data sources to DUET. In this way it becomes for governments an add-on effort on their existing data source cataloging activities.

The figure below shows the DCAT vocabulary. DUET will stick with the DCAT-AP standard and extend as required.

Regardless of the system used to manage data sources, the DCAT standard (https://www.w3.org/TR/vocab-dcat/) will be used as a base for keeping metadata. 3.3 Model Catalog.

DCAT is an RDF vocabulary designed to facilitate interoperability between data catalogs published on the Web. This document defines the schema and provides examples for its use.

DCAT enables a publisher to describe datasets and data services in a catalog using a standard model and vocabulary that facilitates the consumption and aggregation of metadata from multiple catalogs. This can increase the discoverability of datasets and data services. It also makes it possible to have a decentralized approach to publishing data catalogs and makes federated search for datasets across catalogs in multiple sites possible using the same query mechanism and structure. Aggregated DCAT metadata can serve as a manifest file as part of the digital preservation process.

[8] DCAT-AP for Portal in Europe [link]

[9] European Data Portal - https://www.europeandataportal.eu/en

[10] DCAT - https://www.w3.org/TR/vocab-dcat/

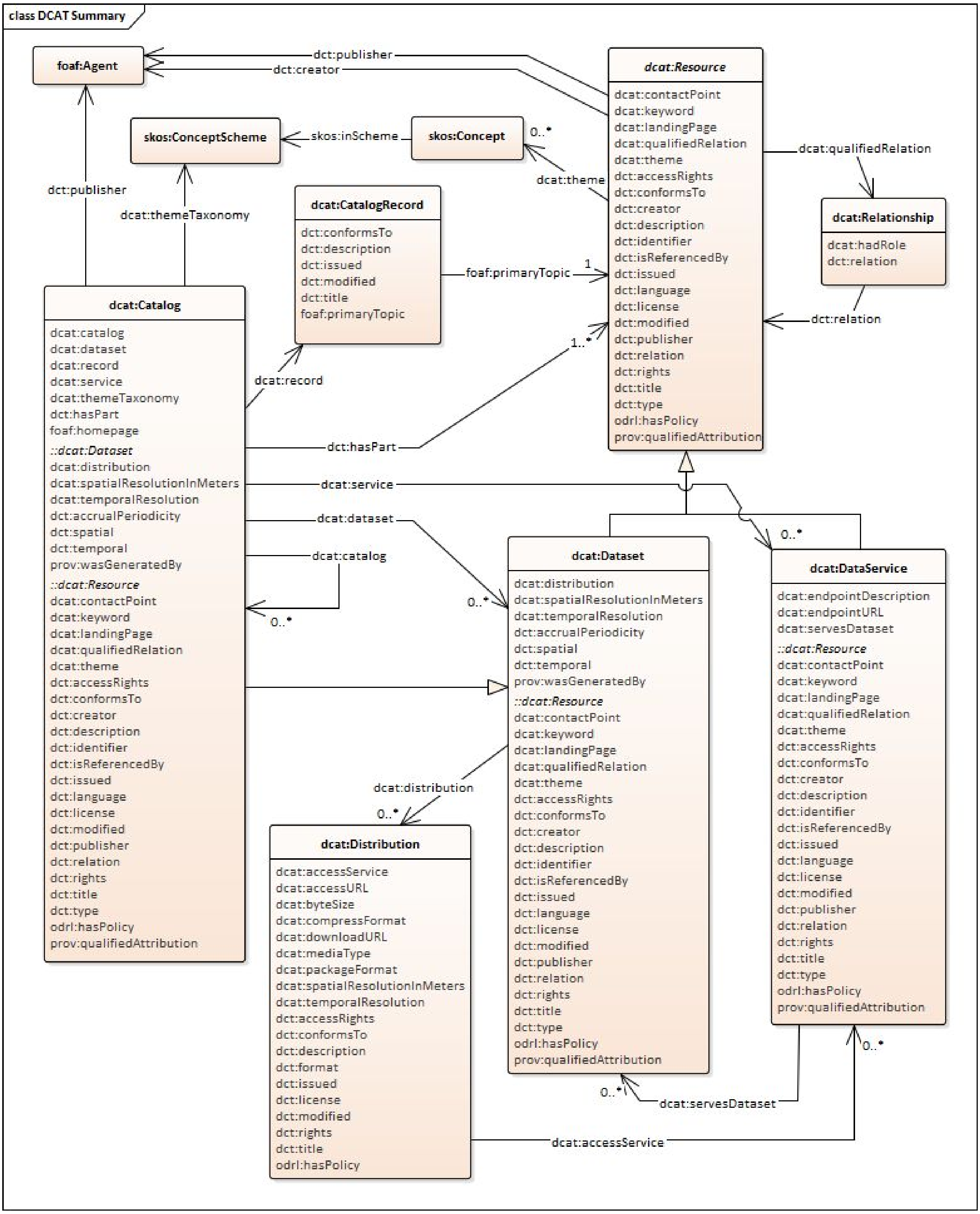


Figure: The DCAT data model. Source Data Catalog Vocabulary (DCAT) - Version 2 (w3.org)

Implementation

In the Alpha Version of DUET, we opted to use CKAN as a starting point for implementing the data catalog component. It fulfills the above API requirements. It implements the necessary operations and support access management for organizations (groups) and individual users.

It does however not support controlled sharing of data sets across organizations. This could be achieved by making use of CKAN extensions that override the different visibility of data sources for users. Alternatively, the decoupled architecture of DUET will allow us to replace the CKAN system entirely should it not adequately suit our needs.

3.3 DUET internal ontology and data model

Aside from the DCAT-based metadata, we also require semantic data models that describe the data itself. For the specification of an internal DUET ontology is foreseen in the project. The idea behind having such shared ontology is that (1) this reduces the semantic mapping efforts by data source consumers as data source providers and (2) it will enable the creation of more reusable tools and libraries in DUET.

Creating a common understanding, and formalizing this in semantic agreements, is a key activity in data interoperability programs. Partner Informatie Vlaanderen is the driving organisation being the Flemish interoperability program Open Standards for Linked Organizations (OSLO). They created a process and method for creating semantical agreements following a number of fundamental principles for the development of standards, which are based on the principles for standards development of OpenStand (https://open-stand.org/about-us/principles/). These principles apply as best practices and have already been endorsed by, among others, W3C, ISA, IEEE, IETF, IAB and Internet Society. The figure below shows the process.

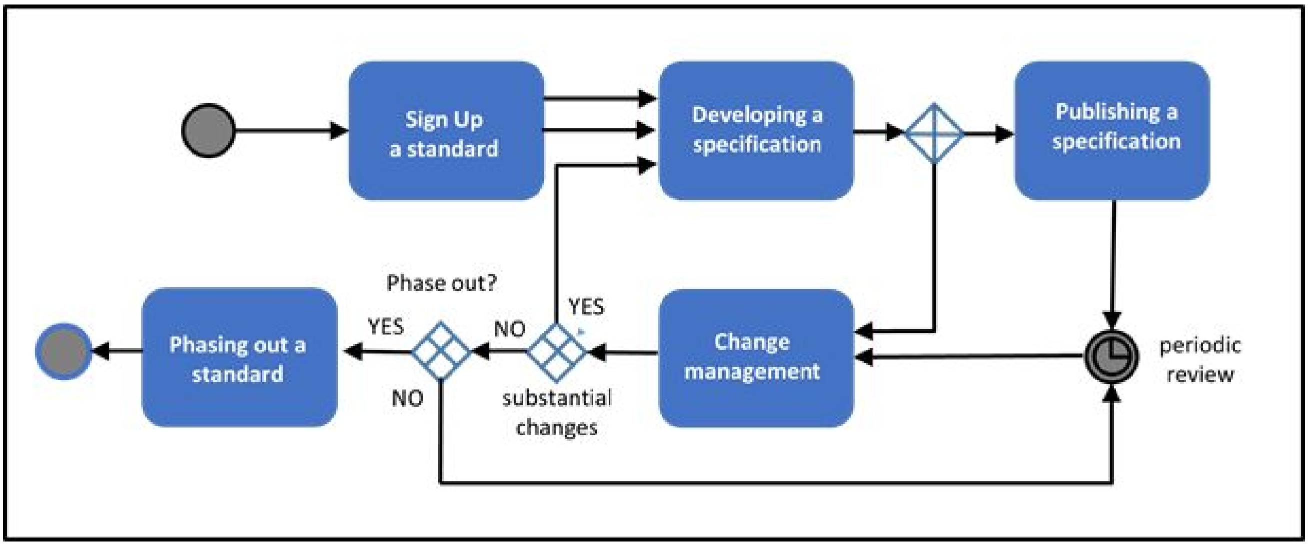


Figure: the process view for the creation of a data standard. Source OSLO[11]

For language reasons, we refer to the variant with minor changes of the OSLO process and method is being applied by the Belgian Interoperability program ICEG[12].

To support this process and method the OSLO program has invested in the creation of a toolchain and publication environment. For the DUET internal ontology a slimmed down variant of the above method is being applied. We will focus on the development and the publication phases of the method. Also the broad consensus principle is reduced. Despite that broad consensus building is by far the most important aspect of

[11] Proces_en_methode_voor_de_erkenning_van_datastandaarden_v1.0.pdf (vlaanderen.be)

[12] Process and method for the development of data standards [document link] - May 24, 2019

the OSLO process and method, it is for the DUET project sufficient to create consensus among the partners. Nevertheless connection with the outside world is not lost. The data standards are being created according to the best practices of publishing data on the web[13]**.** A key step in the development phase is therefore reusing existing vocabularies and ontologies. This creates a strong connection with the outside world. The application of the OSLO processes and methods for the design of the internal model (and also the metadata application profile) ensures that our modeling activities are done in a reusing context for future adoption beyond DUET.

Existing Data Models

For the data model, several options have been explored and the most promising vocabulary to start from is the Semantic Sensor Network Ontology (SSN). It is an ontology for describing sensors and their observations, the involved procedures, the studied features of interest, the samples used to do so, and the observed properties. The SSN ontology exists in two parts. The first part is SSN itself and follows a horizontal and vertical modularization architecture. The second part is a lightweight core ontology, called SOSA (Sensor, Observation, Sample, Actuator), which is used by SSN to describe its elementary classes and properties. Both SSN (blue) and SOSA (green) are visible on the image below, showing an overview of the complete ontology from an observation perspective.

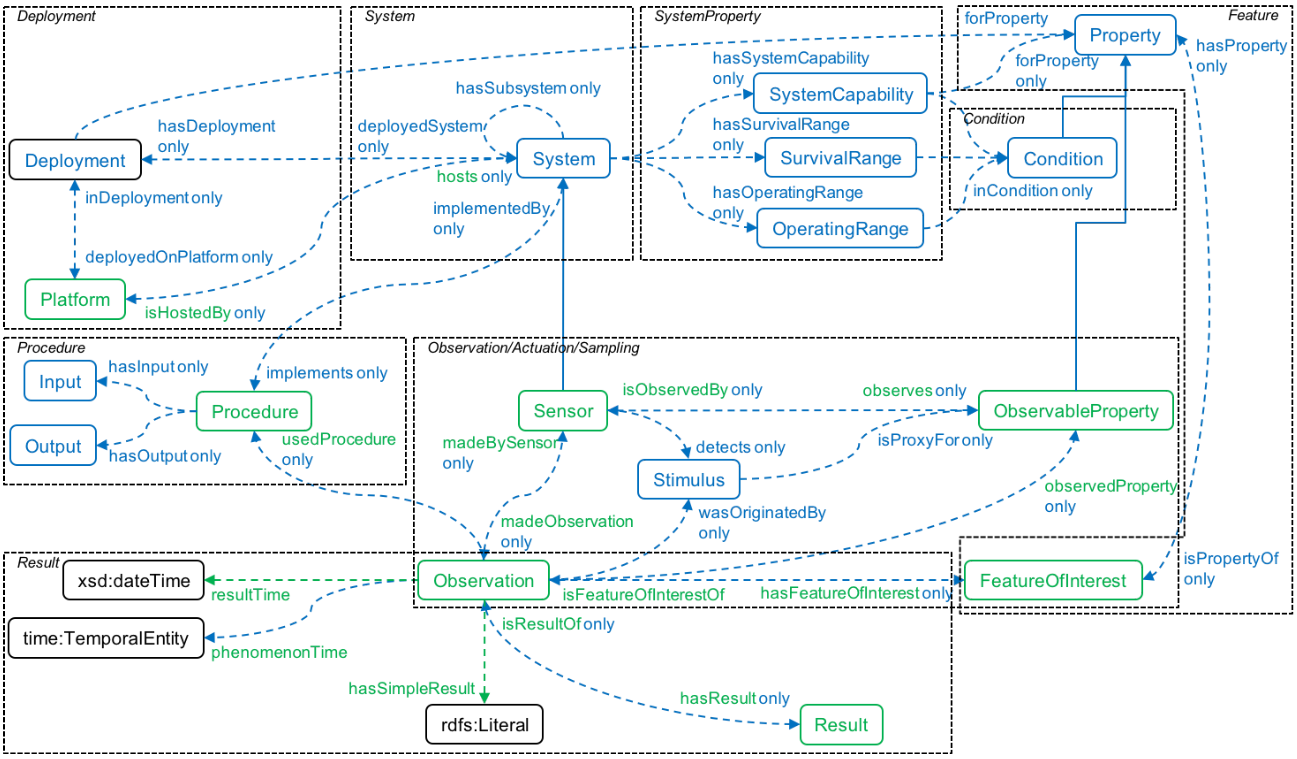


Figure: The semantic sensor network data model. Source Semantic Sensor Network Ontology (w3.org)

[13] Data on the Web Best Practices - https://www.w3.org/TR/dwbp/

Further, the Flemish Interoperability Program OSLO has initiated in October 2020 another project, Open Data Laka (ODALA), for the creation of a data standard for air and water quality. The main technical outputs of the ODALA project will be several advanced open-source data integration modules that combine and integrate legacy and real-time data into a usable framework, the so called "DataLake". The task of OSLO within this project is to define the semantic building blocks for the data that will be exchanged. It is estimated that it takes about 9 months to develop a fully qualified OSLO data standard. In a close collaboration, insights and choices will be shared with the DUET project. Where adequate, the internal DUET ontology will be aligned.

Internal Ontology

The DUET ontology will not be the first of its kind. KM4City[14] has proposed one before and it can serve as an inspiration or even as a basis for our efforts. The project clearly shows the value of a universal ontology for the smart city. A smart city ontology however is a living thing. It changes over time, mostly expanding with new domains and areas. Just as KM4City will be inspirational for building the smart city ontology, we look to the fiware smart data models[15] as a starting point for creating our ontology.

Aside from having a governance process around the standards that help define the ontology (see OSLO), we need to support versioning for the ontology. This also implies that mapping to the internal ontology and assigning types to DUET data streams will not only involve referring to smart city types, but also an ontology version.

In setting up and maintaining the DUET ontology we should stick to the following principles:

1. The ontology should not leak into the code. Changes to the ontology should not require rebuilding DUET components.
2. Modularity is key and changes to ontological modules should have no or limited impact.
3. Ontologies are extensible and can grow over time. New versions of the DUET ontology can be released.
4. Version management of ontologies/vocabularies should be supported.
5. Support a three-layered approach:
    a. Vocabulary (DUET vocabulary)
    b. Application profile (e.g. DUET core as application context: cardinalities, code lists, ...)
    c. Implementation model: including extensions in the form of (extra) fields outside the application profile and/or vocabulary

For the DUET project there is a need for an internal ontology. Considering the different data sources and how they will be measured (by sensors), only the lightweight core, SOSA, is needed as a starting point for the ontology.

The resulting data standard and thus final ontology will be discussed in the second version of this deliverable.

[14] https://www.km4city.org/

[15] https://www.fiware.org/developers/smart-data-models/

3.4 Data mapping & mapping catalog

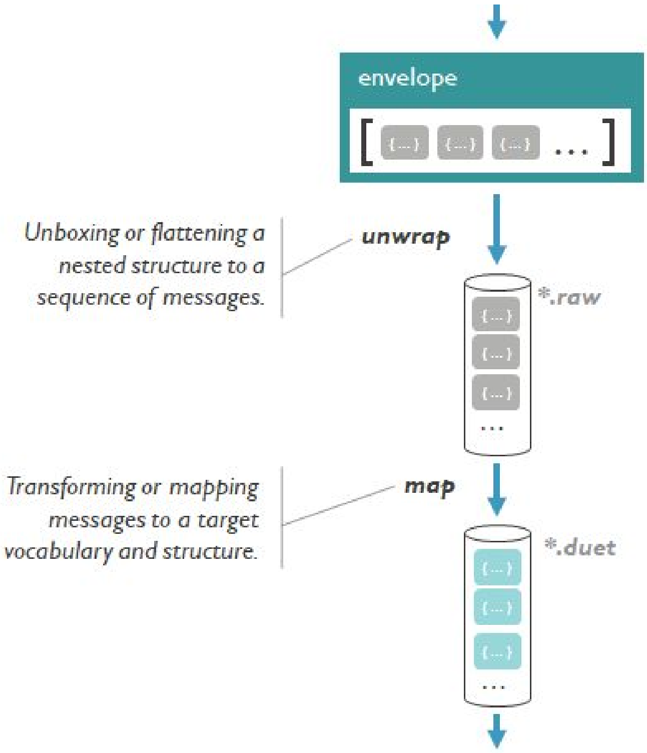
Mapping

It is clear that many of the data sources that we want to use may not be compliant to the internal ontology. After all, many data sources are not standards compliant and even when they are, competing standards are often in play. This will require us to map data from and to the DUET ontology.

Mapping from and to the ontology

For simple cases, it should be possible to describe the mapping declaratively entirely in the configuration. Kazaam[16] can be made available as a pre-existing mapping program in the DUET mapping catalog. The user can then specify how kazaam should run as a configuration argument.

A more simple approach such as flat property mapping could be used in very simple cases.


envelope
[{...} {...} {...} ...]
Unboxing or flattening a nested structure to a sequence of messages.
unwrap
*.raw
{...}
{...}
{...}
...
Transforming or mapping messages to a target vocabulary and structure.
map
*.duet
{...}
{...}
{...}
...

Unwrapping

The data entering the system might be wrapped in an envelope (e.g. a json array containing multiple events), this envelope must be unwrapped. This is typically a one-to-many transformation where one input message translates to many output messages. The resulting output will either be written to the target *message streaming topic* directly or to an intermediate topic (containing raw data) from which the data will be read by another mapping module. This allows for chaining mappings while optionally storing intermediate states as defined in the data catalog entry.

[16] https://github.com/qntfy/kazaam

The use of WebAssembly

To allow for DUET users to add maximally flexible custom mappings to DUET, the user can add WebAssembly programs that can run inside a WASI sandbox[17]. This allows DUET users to develop mappings in any language of their choosing while at the same time providing a secure runtime environment.

This can be used both for unwrapping and for mapping but possibly for other use cases as well (e.g. anonymization). A data catalog entry may then reference one or more such mappings to run when data enters the system. A data catalog entry may optionally supply some additional mapping configuration, so configurable mappings can be reused.

Mapping catalog

Since there will be commonly used input formats such as NGSI-*, OGC Sensorthings, ⋯ having a mapping catalog for reusable mappings may be convenient. Users can store and share their mappings in a mapping catalog for other users to use or DUET could provide standard mappings complying with the internal ontology. It is uncertain if the mapping catalog will fit in the scope of DUET.

Data onboarding, mapping and Validation Tooling

Onboarding data, defining and selecting a mapping, and finally selecting the matching ontology type and validating the result is a non trivial task. It is therefore essential that DUET supplies tooling for this in the form of a data management wizard with the following features:

- A wizard for registering a new data source
- Ontology type selection
- Validation of sample data against that ontology
- Mapping configuration tooling
- Mapping testing tooling for sample data

Implementation Note: We have decided to implement a simple mapping architecture starting from a fit for purpose approach.

4. Tools

4.1 OSLO Toolchain

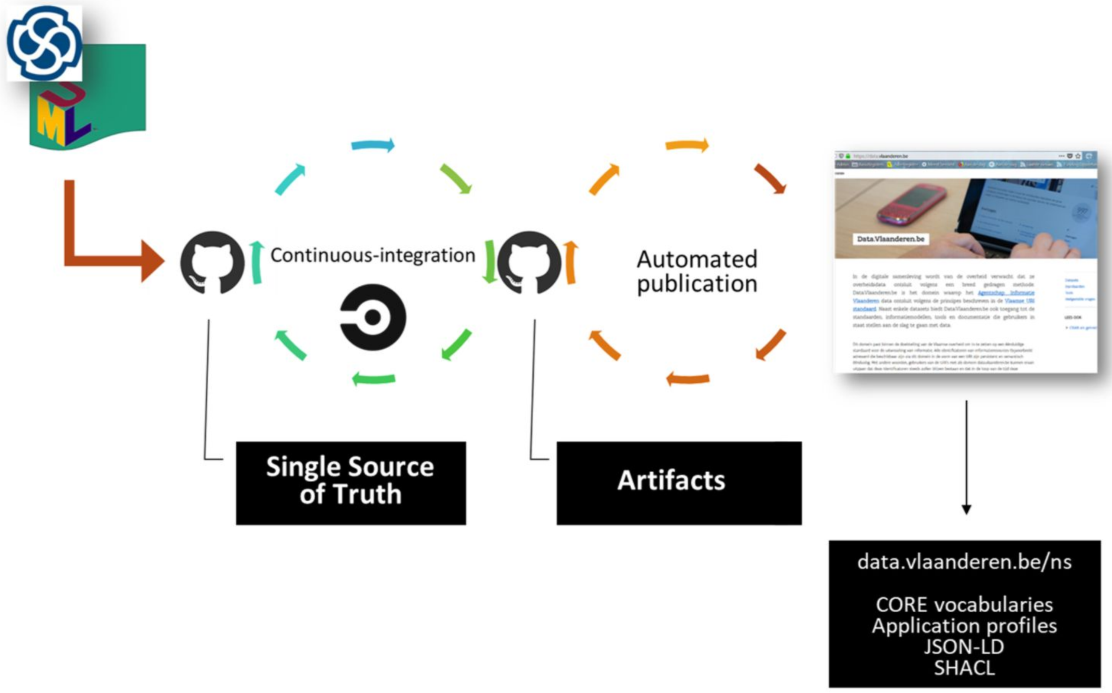
The Flemish Information Agency has developed an ecosystem of tools, processes and governance for modelling and publishing Semantic Data standards. This ecosystem relies on Open Source software and is executed entirely by open continuous integration systems. This choice has its roots in addressing the key principle of the OSLO: creating a broad consensus by maximal transparency and minimal access barriers. This openness creates additional opportunities: it creates an open community around the building and creation of data standards. And importantly the low barriers (minimal costs, setup times) to use the OSLO toolchain increase the adoption by others.

[17] https://wasi.dev

The tools within the ecosystem allow the design of data standards in a decentral approach, embedded in global governance. The setup works on the basis of separation of concerns:

- Modelling
- Provenance
- Publication with a maximal of automation to facilitate expectations for both humans and machines

The core process, usually referred to as OSLO Toolchain and visible at the image below, is as follows. First, the semantic data modelling starts with an annotated UML document. This UML document, along with some other documents, is stored in a GitHub repository, dedicated to that specific data standard. The dedicated GitHub repository is considered the single source of truth for the data standard. Via various tools the UML document is then converted to human machine-readable and machine-processable artifacts, such as HTML pages, JSON-LD context files, SHACL files, ··· The automation is done via CI/CD configuration on the involved github repositories. More information about the tools used in this ecosystem can be found here (Github) and here[18] (the latter only in dutch).



The methodology and setup has been applied in real world projects where the data standards, vocabularies for broad reuse on the one hand and application profiles for usage in a generic application context on the other, are turned into implementation models using the same Toolchain approach. These implementation models are then embedded in deployed software artifacts and APIs.

DUET Toolchain configuration

The toolchain requires setting up a minimum of 3 repositories:

- A repository for the information about the DUET semantic assets. Initially all ontologies will be maintained in this repository. In the future new repositories are created to seperate more the

[18] Only in Dutch - translation is being planned and will be made available in the course of the project.

editorial activities on the distinct ontologies. https://github.com/Informatievlaanderen/duet-ontology

- The source repository of the publication environment. Here the automation of the OSLO toolchain is set up. https://github.com/Informatievlaanderen/duet
- The result repository for storing the outcome of the Toolchain processing https://github.com/Informatievlaanderen/duet-generated

Dereferenceable identifiers

To ensure that the semantic agreements are being used in the OSLO process and method for data standards it implements persistent dereferenceable identifiers. Each term is uniquely identified by a URI. This principle is one of the cornerstones of publishing data on the web[19] and is the first principle of FAIR[20] data management. For DUET the domain **data.citytwin.eu**[21] has been assigned to support this objective.

Multilingual adaptations

Designing semantic data standards is strongly connected with the natural language being used by the participants in the process. In the public sector the natural language is even more critical as this is also the language in which legislation is expressed. Since data standards for the public sector are tools to express legislation into the digital world the use of natural language is critical and cannot be simply replaced with another lingua franca.

This business context is in contrast with the tooling support: Semantic Web technology is by far the most supportive technology for multilingual content that exists. The RDF data format is by design multilingual.

The OSLO toolchain and publishing environment have been adapted by removing the assumptions on the Dutch language for application in DUET. The toolchain now supports multiple languages. Support for multilingual data specifications will also increase the adoption of DUET in Europe. It may for instance reduce barriers for the DUET data mapping. English documentation will be created and published soon.

4.2 CKAN

We will use CKAN (https://ckan.org/) as the base for our data catalog implementation. According to Fiware, CKAN is the world’s leading open-source data portal platform. It is a powerful data management system that makes data accessible – by providing tools to streamline publishing, sharing, finding and using data. CKAN is aimed at data publishers (national and regional governments, companies and organizations) wanting to make their data open and available.

CKAN is a mature and feature-rich product and we used the following features as a motivation to adopt it:

API

CKAN features an API which is of course key for integration into DUET. For now we estimate that the built-in CKAN API will fulfill our needs. If needed we will create a wrapper API that allows us to extend the existing API if needed.

Extensions

[19] Data on the Web Best Practices - Data identifiers - https://www.w3.org/TR/dwbp/#DataIdentifiers

[20] FAIR principles - https://www.go-fair.org/fair-principles/

[21] To be configured

Some extension points - specifically about security and data set visibility - can be relevant to the DUET project. Over 200 extensions already exist. One extension, which enables DCAT support will be required by the project.

Federation

Because CKAN's harvesting functionality can be used to pull in metadata from other data portals, CKAN can be used to create a federated network of data portals which share data between each other.

Geospatial

With the spatial extension enabled, CKAN can understand a location associated with a dataset, and use this to offer geospatial search capabilities via the web interface and API. A user searching for datasets can filter the results by geographical location, specifying a bounding box to limit the area he is interested in.

Metadata

By default, CKAN provides the following metadata fields:

Identifiers

- Internal ID - Generated by CKAN and unique in the scope of the system.
- Unique identifier – a unique URL which is customizable by the publisher, but it could be that it is not unique in the context of federation. Handling such URI collisions is considered out of scope. Identifier handling will be an issue for any system that assumes an identifier which is not a URI.

Descriptive fields

- Title – allows intuitive labelling of the dataset for search, sharing and linking.
- Description – additional information describing or analysing the data. This can either be static or an editable wiki which anyone can contribute to instantly or via admin moderation.
- Licence – instant view of whether the data is available under an open licence or not. This makes it clear to users whether they have the rights to use, change and re-distribute the data.
- Tags – see what labels the dataset in question belongs to. Tags also allow for browsing between similarly tagged datasets in addition to enabling better discoverability through tag search and faceting by tags. DUET could provide some default tags to help categorize data sets.
- Multiple formats (if provided) – see the different formats the data has been made available in quickly in a table, with any further information relating to specific files provided inline.

End user functionality fields

- Data preview – preview .csv data quickly and easily in-browser to see if this is the dataset you want.
- Groups – display of which groups the dataset belongs to if applicable. Groups (such as science data) allow easier data linking, finding and sharing amongst interested publishers and users.
- Revision history – CKAN allows you to display a revision history for datasets which are freely editable by users (as is thedatahub.org)
- API key – allows to access every metadata field of the dataset and ability to change the data if you have the relevant permissions via API.
- Extra fields – these hold any additional information, such as location data (see geospatial feature) or types relevant to the publisher or dataset. How and where extra fields display is customizable.

CKAN has the ability to import and export DCAT by means of the plugin **ckanext-dcat** (see https://github.com/ckan/ckanext-dcat). However, the API does not use DCAT, but the CKAN data model. It remains to be seen if this extension fulfills our needs

Search and discovery

CKAN provides a rich search experience which allows for quick keyword and tag-based searches and browsing between related datasets. Users can quickly see what datasets are available, in which formats and with which licence, straight from the search results. All dataset fields are searchable (see above for the metadata fields).

Themability

CKAN is highly customizable. The appearance of the CKAN portal can be conveniently customized with a DUET look & feel.

Note: The use of CKAN's ability to store data sets is not foreseen. This would only be possible for relatively small data sets or smaller parts of larger data sets anyway. Given the scalability limitations, storing large data sets in CKAN does not seem a recommendable approach and it would not provide a lot of added value.

4.3 Log Analysis

As indicated in requirement 5.9.2, the audit trails of the DUET platform about sharing and accessing data, models and the management of users and their authorization need to be accessible for monitoring and alerting.

It has not yet been decided which approach will be taken. Unless a built-in tool with an integrated experience is mandated by the project, the most straightforward approach is to use a log analysis tool.

The following open source options are available to us:

- Graylog: a stream based solution for log analysis that could be linked to the DUET core,
- ELK Stack: Elastic Stack, often called the ELK Stack, is one of the most popular open source tools among organizations that need to sift through large sets of data and make sense of their system logs,
- Fluentd: a robust solution for data collection. It does not offer a full frontend interface but instead acts as a collection layer to help organize different pipelines.

The main requirement in choosing an external tool is the possibility to feed back alerts and issues to the DUET users so they can be made aware of them in the DUET client software.

5. Functional Requirements

This section describes the high-level functional requirements for the different DUET Data Broker and its components. These may evolve with time. The next version of this deliverable will elaborate further on these.

5.1 Managing data sources

Nr	Title	Description
1.1	Manage diverse data sources	Users should be able to inventorize, add, remove and update diverse data sources and manage the metadata. Any data that is not IoT related or Geographical of nature can be seen as diverse data.
1.2	Manage (IoT) event sources	A DUET user should be able to add, remove and update IoT event sources.
1.3	Manage (IoT) context data sources	A DUET user should be able to add, remove and update IoT context data sources.
1.4	Manage (IoT) historical data sources	A DUET user should be able to add, remove and update IoT historical data sources.
1.5	Manage geo data sources	A DUET user should be able to add, remove and update geographical data sources.

5.2 Managing models

Nr	Title	Description
2.1	Manage models	As a DUET user I should be able to add, remove and update models
2.2	Manage context data and metadata of models	A DUET user should be able to add, remove and update model context data, including: ● Detailed specification of input data sources (type and cardinality) ● Detailed specification of output data sources for publishing the result ● Extra expectations such as data resolution and context information about the model ● Calibration data source if applicable Additional metadata fields in analogy with the data catalog

		may be provided to facilitate searches.
2.3	Control public access (optional)	Allow public access to the model, allowing anyone to use the model.
2.4	Model access management	Allow others to use the model by sharing it with registered users through their organization or individually.
2.5	Revoke sharing	Stop sharing the model with an organization or individual.

5.3 Managing visualizations

Nr	Title	Description
3.1	Client registration	As a visualization publisher I want to be able to register my visualization/interaction client along with metadata such as the kind of visualization, required inputs, data formats, etc.
3.2	Control access to clients	As a visualization publisher I want to control public access to the visualization/interaction client
3.3	Client access management	As a visualization publisher I want to grant access to a published visualization I own to an organization or individual
3.4	Stop sharing clients	As a visualization publisher I want to revoke access to a published visualization I own from an organization or individual

5.4 Managing mappings

Nr	Title	Description
4.1	Manage mappings	A DUET user should be able to create, add, update and delete user-defined mapping engines.
4.2	Test a mapping	A DUET user should be able to validate user-defined mapping engines by applying it to samples.
4.3	Mapping configuration	It should be possible to supply different mapping configuration for each data stream the mapping engine is applied to.
4.4	Declarative mappings	A basic declarative mapping engine should be available for simple mapping cases. The declaration can be passed as configuration.

4.5	Data packages unpacking	A mapping engine should be able to do 1-to-many mappings allowing for unpacking of data packages
4.6	Mapping catalog	A DUET user should be able to have an overview of the mapping engines available.
4.7	Mapping chaining	Multiple mapping engines should be chainable writing to intermediate buffers.

5.5 Managing the ontology

Nr	Title	Description
5.1	Refer to the Ontology	The user can refer to the ontology to indicate what specific type of data is offered by the source system upon registering a data source in the data catalog
5.2	Mapping to/from the Ontology	In the case of a data source not being compliant, the user can provide mapping specifications to map the source data to the DUET ontology. The mapping can be selected from the mapping catalog. Mappings can be chained. Mappings should also allow users to map from the DUET ontology onto a more desirable format for the needs of the user.
5.3	Ontology Versioning	As an administrator, I should be able to upload new versions of the ontology to support new use cases. The DUET smart city ontology will be managed outside the DUET instance. DUET instances can download newly released versions from a central repository.
5.4	Knowledge graph	The knowledge graph database will contain all versions of the DUET smart city ontology. Deploying a new instance of DUET will include a recent version of the DUET ontology.
5.5	Manage dereferenceable identifiers	To ensure semantic agreements for data standards, the use of dereferenceable identifiers is encouraged.
5.6	Multilingual Ontology	Multi-language ontology specifications should be supported.

5.6 Authentication and Authorization

Nr	Title	Description
6.1	Register users and organizations	It should be possible to register as a DUET user via an email validation process. As a user I should be allowed to set up an organization or logical group. In this case I will become the owner of that group. I can pass ownership to other group members.
6.2	Adding/removing users to organizations	As an organization/group owner (referred to as registered users in D5.1), I can invite existing users to my organization/group. The invited user receives an email with a link to accept membership. Users can belong to multiple organizations. Users can remove themselves from a group or they can be removed from it by the group owner.
6.3	Request access to data, models or visualizations	A DUET user should be able to request access to catalog items. The owner receives an email with a link to grant access.
6.4	Granting/revoking access to data, models, visualizations	A DUET user should be able to grant access to a catalog item upon request or spontaneously. Revoking access for organizations/users with access should equally be possible.
6.5	Log activities to an audit trail	All activities should trigger an event that is registered in an audit trail
6.6	Manage users and groups and access	As a DUET admin, I should be able to manage users and groups and grant /revoke administrative privileges. Administrators have full control over users, groups/organizations and access rights.

5.7 Accessing Data

Nr	Title	Description
7.1	IoT Event based access	As a data producer, I should be able to publish IoT event data to event based data sources (and their corresponding data streaming topics). As a data consumer, I should be able to subscribe to IoT events from event-based data sources (on the corresponding data streaming topics).
7.2	IoT time series access	As a data consumer I should be able to request IoT time series data from an IoT time series data source on demand by sending a parameterized query request to the DUET data API

7.3	IoT context data access	As a data consumer I should be able to request IoT context data from an IoT context data source on demand by sending a parameterized query request to the DUET data API
7.4	Geo Data Access	As a data consumer I should be able to request geo data from a geo data source on demand by sending a parameterized query request to the DUET data API
7.5	API data access	As a data consumer I should be able to request data from an API data source on demand by sending a parameterized query request to the DUET data API
7.6	Data sending	As a data producer I should be able to send data to a DUET data source via the DUET data API. For instance, a model should be able to publish its result onto the message streaming topic corresponding to the output data source of the model.
7.7	Data source inventory	As a DUET user, it should be able to receive an inventory of the available data sources
7.8	Data source discoverability	As a DUET user, I should be able to discover data sources by navigating the knowledge graph
7.9	Data source querying	As a DUET user I should be able to query the data catalog based on the metadata, smart city domain / ontology, geographical bounds or to browse through data sources based on smart city domain / geography.

5.8 Running & Searching Models

Nr	Title	Description
8.1	Start a model run	As a DUET component or user, it should be possible to initiate a model run via the data broker. The call should include specific parameter data: ● input data sources ● an output data source(s) ● limitations in time and geographical boundaries Starting a model run should return an identifier for the run that permits getting the status of the run or stopping it.
8.3	Stop a model run	As a DUET component or user, it should be possible to stop a model run via the data broker (through the Model API).
8.4	Get the status of a model run	As a DUET component or user, it should be possible to receive the status of a model via the data broker (through the Model API).
8.5	Subscribe to events	As a model, it should be possible to use an event stream,

		historical data sources and geographical data sources via the IoT Data API.
8.6	Publish events and historical data	As a model provider (cfr. both data producer and data consumer roles in D5.1), it should be possible to publish an event stream and historical data via the IoT Data API.
8.7	Search models	As a user it should be possible to search for models by all possible metadata fields, including but not limited to model name, data types, smart city domain, publisher, ...
8.8	Model output	As a model user I want to be able to record and archive the output of the model run to persistent storage

5.9 Managing the DUET platform

Nr	Title	Description
9.1	Manage configuration	As a DUET administrator I want to access and manage all configuration settings of a DUET instance.
9.2	Consult Audits	As a DUET administrator I want to consult the audit logs of a DUET instance.
9.3	Update Ontologies	As a DUET administrator I want to manage the internal ontology by updating to the latest version of the DUET ontology.

6. Technical Architecture

DUET also uses an event-driven architecture that is powered internally by a message streaming platform (Kafka in our case). The use of such a data broker allows us to stream data across the platform with the following benefits:

- **Streaming**: streaming data allows us to implement an event driven approach where different components can interact;
- **Mapping**: by streaming data we can implement mapping efficiently without blocking the message processing,;
- **Decoupling** via pub/sub and eventing: A decoupled system helps to build a maintainable and scalable platform. Limiting dependencies will prevent changes from rippling through the architecture;
- **Responsiveness**: not only is the use of a message streaming platform scalable in itself by supporting horizontal scale-out, but the independent handling of message streams prevents bottlenecks from choking up the system, which is essential in building scalable responsive systems.

6.1 Components

This section discusses the different components of the databroker and their interconnections. The figure below depicts the data broker from the viewpoint of security. Everything inside the red oval is shielded from outside access. Most components are not accessible directly. The different gateways provide secure authenticated access to exposed functions.

For a more elaborate discussion of the architecture, the components and their connectors we refer to DUET Deliverable 5.1: *System Architecture & Implementation Plan*.

Note: For the sake of simplicity we assume in what follows that the **mapping, model and visualization catalogs** are part of the data catalog. This disregards the fact that it is unclear if it is technically possible to use the same component for that purpose. These needs to be clarified later in the next version of this deliverable.

Note: Although we refer frequently to Gateways (e.g., IoT Data Gateway, App Gateway, ...) and APIs (e.g., Model API, Data API, ⋯) these are not actual components. Instead they represent different logical groupings of components- usually micro services - that each offer a small part of functionality.

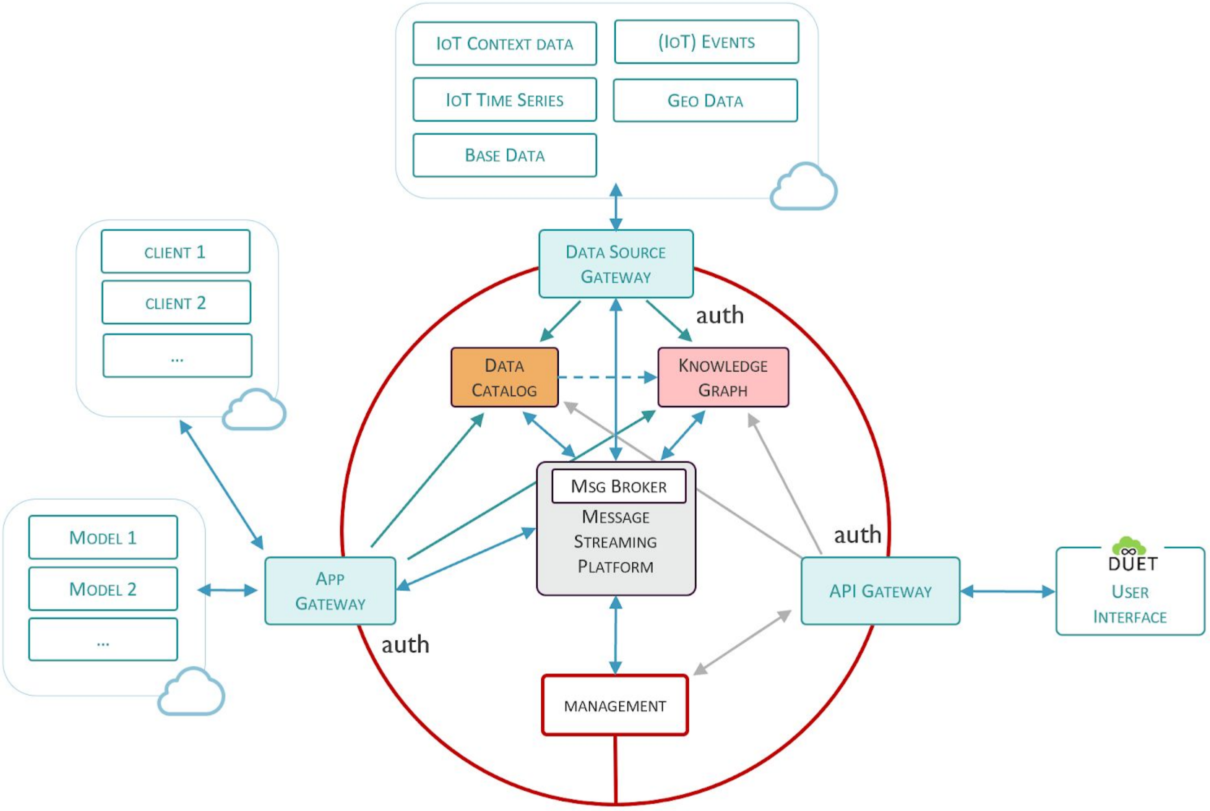


Figure: DUET security diagram. Everything inside the red boundary is part of the DUET data broker system.

Data Source Gateway

The Data Source Gateway is a logical grouping of following components:

- IoT streaming receptors: for connection IoT data sources that push data to the message streaming system
- IoT polling receptors
- IoT time series API receptors
- IoT Context API receptors
- Geo data API receptors

App Gateway

The logical component App Gateway covers the following components:

- The Message Receiver that accepts messages / data from the models and relays them into the cell, through the streaming platform, after performing data validation and authorisation
- The Message Sender that allows consumers of data such as visualizations and models to retrieve data from DUET data sources or subscribe to DUET data streams
- The Model API allows the DUET data broker to control model execution and monitor model instance status

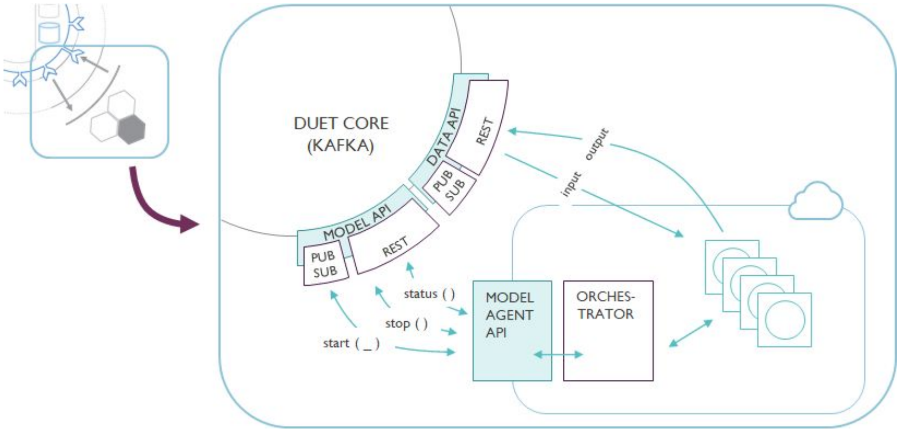


Figure: the app gateway is a logical grouping of components to interact with models and visualization / interaction clients of DUET

API Gateway

The API Gateway groups all externally accessible APIs, and provides a single point of access for external applications, such as:

- The Digital Twin Administration and management API
- The data catalog, model catalog and visualization catalog APIs
- The ontology management API

Message Streaming Platform

The Message Streaming Platform is the central component around which the data broker is built. It fulfills the important role to enable a decoupled and scalable publish/subscribe system for digital twin data, whether it is fast moving (IoT/sensor measurements) or slow moving (occasional changes in the street layout of a city).

The **Message Broker** provides operations related to the Message Streaming Platform like retrieval of topics, topic creation, etc that can be triggered by an Administrator manually or by the system automatically upon the reception of relevant events.

Management Module

The Management module provides an API to manage shared digital twin settings. Most of these settings will be accessible to administrators only.

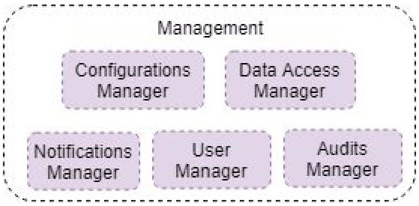


Figure: components of the management module

6.2 Dependencies

The DUET data broker components are dependent on one another in the following way:

- The Data Catalog depends on the knowledge graph in the sense that it needs references to entities in that graph to indicate to what type the native or mapped entities correspond.

- The Data Catalog will also refer to mappings in the mapping catalog since users will be able to specify which mapping should be used to map their source system data onto data that conforms to the internal ontology.
- The IoT Data Gateway depends on the message streaming platform for receiving, mapping and sending data.
- The IoT Data Gateway also relies on metadata from the Data Catalog to make sense of data requests it receives and the Knowledge Graph for validation.
- The API gateway depends on the internal components (Data Catalog, Knowledge Graph, Management Module) to expose some or all of their features.
- All the internal components (Data Catalog, Knowledge Graph, Message Broker, Management Module) depend on the message streaming platform for publishing messages about updates in the respective component.
- The App Gateway depends on the Message Streaming platform for the following actions:
    - Sending events that are the result from interactions in DT clients
    - Subscribing to events from data sources (including models)
    - Sending data requests to API connectors and receiving the result of those requests
- The App Gateway depends on the Data Catalog and Knowledge Graph to resolve data source names and validate input data respectively.

7. Security

All the aforementioned components will be securely accessed and managed using a unified Identity and Access Management solution. Aiming at a centralised management solution, we plan to build the DUET Identity and Access Management (D3.10) upon Keycloak, which is an open source solution and provides the capabilities of user and role management, defining of authorisation policies and off-the-self clients supporting multiple applications and programming languages.

Details of security requirements and relevant measures to address them are analytically described in D3.10.

7.1 Authorization and authentication

Authorization and authentication in the DUET data broker is done at the level of the gateways. Communication inside the broker, bounded by the red circle in the architecture drawing, is considered to be trusted.

For every request, the requestor is authenticated and authorization to access to the requested resources is verified by a central security service.

7.2 Data Source, Model, Visualization Client sharing

By default, data sources, models and visualization clients that are published/registered by the user are not accessible to other users. It is up to the users to either

- make their work public and free to use for all
- or grant access to specific users or groups for the use of their work.

Every action to grant or revoke access is logged in an audit trail.

7.3 Auditing

The following actions are subject to a log entry in the audit trails:

- Registration/creation, update or deactivation of users or user groups
- Granting or revoking administrative privileges
- Any CRUD action on a catalog item (data source,mapping, model or visualization client registration)
- Updating the metadata of a catalog item
- Upgrading the ontology or manipulating it in any way
- Changing the sharing settings of any catalog item (data source, mapping, model, visualization client)
- Starting a model run

7.4 Notifications

Any of the above auditing events will also trigger notifications to the DUET administrators or any data owner impacted by the action itself unless the data owner specifically indicates otherwise.

8. Data Licensing

Mixing and matching data from different sources is the core business of the Urban Digital Twin. This potentially poses different problems with data licences and the limitation they incur on the use, reuse and redistribution of the data. Given the complex legal consequences of mixing different licence models, determining what limitations apply to which data received directly or indirectly can be a daunting task.

For now, we foresee no automated compatibility check that ensures integrations only use the data in accordance to the expressed licenses. It remains unclear how a component checking the compatibility would feature in DUET. Does it act as a transparency component by logging all interactions so that legal audits are facilitated? Or does it actively intervene in the connection processes when researchers and users are experimenting with the data.

DUET will facilitate the recording of applicable licenses using the DCAT metadata descriptions in the Data Catalog: https://www.w3.org/TR/vocab-dcat-2/#license-rights

Existing work on data license compatibility can be found in the following references.

- https://joinup.ec.europa.eu/collection/eupl/solution/joinup-licensing-assistant/jla-compatibility-checker
- http://ceur-ws.org/Vol-905/VillataAndGandon_COLD2012.pdf
- https://www.comsode.eu/index.php/2015/02/licence-comparison-and/
- http://licentia.inria.fr/

9. Conclusion

Upon writing the conclusion of this deliverable, the Alpha Version is slowly materializing as well. The Alpha Version validates the important data flow principles that we have discussed in this document. Defining and refining The DUET architecture is a step by step process based on an architectural concept that will gradually become more refined and crisp based on needs of practical implementations.

At the core of DUET we use a scalable and responsible-by-design message streaming platform that will enable us to realise powerful responsive digital twin use cases driven by events and responsive models i.e., models that run in real-time. The components around have supporting tasks to achieve this:

- **The component catalogs**: The data catalog, model catalog, mapping catalog and visualization/interaction client catalog help users to find their way around in the jungle of components to build their own digital twin. They allow them to share their components and data with other users as well.
- **The knowledge graph & ontology**: The DUET internal ontology that is stored in the knowledge graph helps to normalize data streams coming from different providers so it can be combined in models and visualizations.
- **The gateways**: The gateways are APIs provided by separate components (microservices) that regulate traffic from and to the broker. Data providers and consumers, management portals, models and visualisation/interaction clients use these gateways to interact with each other through the message streaming platform.
- **The mapping**: Mappings ensure that data can be uniformized to the internal ontology - the *lingua franca* of DUET.
- **Security & management**: The DUET data broker is bounded by a security layer that shields the message streaming platform and other internal components from direct access by users. The message streaming platform should be abstracted away and the inner broker operation that controls data traffic should remain opaque for external users and components.

As pointed out frequently in this and other deliverables, the use of and support for open standards, standardized interfaces and open technology is key to achieve an open and maximally interoperable digital twin platform. Hence we are looking at existing standardization efforts such as W3C, OGC, Fiware, etc. as a main driver for our design decisions.

The DUET architectural outline is getting more clear. However, some questions are still left unanswered and some topics are only briefly touched. A lot of work lies ahead of us in fleshing out the architecture even more and in implementing these principles in a working system as well.

10. References

Ghent data broker (dutch)
https://stad.gent/nl/over-gent-en-het-stadsbestuur/stadsbestuur/wat-doet-het-bestuur/gent-internationaal/samen-internationaal-werken/europese-subsidies-en-projecten-13

Ghent data broker end report (pdf document)
https://stad.gent/sites/default/files/media/documents/Vlaio%20City%20Of%20Things%20project%20DataBroker%20eindrapport_versie_12032020.pdf

CKan - The open source data portal software
https://ckan.org/

Apache Kafka - a distributed streaming platform
https://kafka.apache.org/

OSLO - Open standards for linking organizations (dutch)
https://overheid.vlaanderen.be/producten-diensten/oslo

Process and method for the development of data standards - May 24, 2019
https://github.com/belgif/review/blob/master/Process/201906-ICEG%20-%20process%20and%20method.docx

W3C SSN - SOSA - Semantic Sensor Network Ontology
https://www.w3.org/TR/vocab-ssn/

A Lightweight Ontology for Sensors, Observations, Samples, and Actuators
ttps://www.researchgate.net/publication/326335033_SOSA_A_lightweight_ontology_for_sensors_observations_samples_and_actuators

VLOCA - Flemish open city architecture
https://vloca-kennishub.vlaanderen.be/

Fiware NGSI-v2
https://fiware.github.io/specifications/ngsiv2/stable/

NGSI-LD
https://fiware-datamodels.readthedocs.io/en/latest/ngsi-ld_howto/index.html

OGC Sensorthings
https://www.ogc.org/standards/sensorthings

DCAT - Data Catalog Vocabulary
https://www.w3.org/TR/vocab-dcat/

DCAT-AP - Data Catalog Vocabulary

https://joinup.ec.europa.eu/collection/semantic-interoperability-community-semic/solution/dcat-application-profile-data-portals-europe

European Data Portal
https://www.europeandataportal.eu/en

Smart architecture: why a smart city can be so much more than just the sum of its smart parts
https://www.imeccityofthings.be/en/blog/hoe-een-slimme-stad-meer-moet-worden-dan-de-som-der-slimme-delen

Snap4City (aka KM4City) - scalable Smart aNalytic APplication builder for sentient Cities and IOT
https://www.snap4city.org

WASI - The WebAssembly System Interface
https://wasi.dev/

JLA - Joinup License Assistant (JLA - Compatibility Checker)
https://joinup.ec.europa.eu/collection/eupl/solution/joinup-licensing-assistant/jla-compatibility-checker

Licenses Compatibility and Composition in the Web of Data
http://ceur-ws.org/Vol-905/VillataAndGandon_COLD2012.pdf

Licence Comparison and Compatibility Assessment
https://www.comsode.eu/index.php/2015/02/licence-comparison-and/

Licentia - a suite of services to support you in looking for a suitable license for your data
http://licentia.inria.fr/

Fiware Smart Data Models
https://www.fiware.org/developers/smart-data-models/

OSLO - Open Standards for Linking Organizations
https://github.com/Informatievlaanderen/duet/tree/master/documentation