Deliverable

D3.7 OSLO Extensions for the Digital Twin - Final

Project Acronym:	DUET	
Project title:	Digital Urban European Twins	
Grant Agreement No.	870697	
Website:	www.digitalurbantwins.eu	
Version:	1.0	
Date:	29 July 2022	
Responsible Partner:	AIV	
Contributing Partners:	IMEC, TNO	
Reviewers:	**Internal** Walter Lohman (TNO) Philippe Michiels (IMEC) **External** Pieter Morlion (MORE LION) Yannis Charalabidis Andrew Stott	
Dissemination Level:	Public	X
	Confidential – only consortium members and European Commission	

Revision History

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

Revision	Date	Author	Organization	Description
0.1	18.05.2022	Lieven Raes	AIV	Initial structure
0.2	22.06.2022	Dwight Van Lancker	AIV	LDES section
0.3	05.07.2022	Bert Van Nuffelen	AIV	OSLO Toolchain
0.4	16.07.2022	Philippe Michiels	imec	MIMs
0.5	18-19.07.2022	Dwight Van Lancker	AIV	OSLO Toolchain, Exec summary & Conclusion
0.6	20.07.2022	Dwight Van Lancker	AIV	References
0.7	25-26.07.2022	Dwight Van Lancker	AIV	Adopt feedback
1.0	29.07.2022	Dwight Van Lancker & Bert Van Nuffelen	AIV	Final version

Table of Contents

1. Executive Summary

One of the key pillars of a Digital Twin is data, as it can be used to enrich another dataset, can be displayed on a map or simulate certain scenarios. To integrate data in a Digital Twin, it is necessary to make agreements about the data models and their semantics. With the Flemish Interoperability program, called Open Standards for Linked Organizations (OSLO), Digital Flanders has the necessary tools and processes to develop a semantically meaningful data model. These tools, collectively referred to as the OSLO Toolchain, were thoroughly discussed in the previous deliverable D3.6. To be able to use the OSLO Toolchain at an international level, it was necessary to extend it with multilingualism in order to develop data models in any language. This extension was prototyped by developing an English data model based on an air quality dataset in Pilzen, and also discussed in the previous deliverable. In the meanwhile, multilingualism has been integrated into the latest version of the OSLO Toolchain. Additionally, the necessary effort has also been made to be able to deploy a new instance of the OSLO Toolchain anywhere very quickly and easily and will be discussed in this deliverable. Along with the OSLO toolchain, DUET also uses other ICT standards, which will also be discussed in this deliverable.

In addition to the data model, the integration of the data itself is also an important part of the Digital Twin. The data publisher usually offers one or multiple Application Programming Interfaces (APIs) on top of their datasets, allowing the data consumer to integrate these APIs into their backend, and that way use the data in their application. However, all these integrations often require custom code which is expensive to develop and often not reusable. In this deliverable, we discuss a new data publishing strategy called Linked Data Event Streams (LDES), where data consumers such as DUET, have to create the needed functionality (APIs) themselves, but have access to the full dataset to realise this. We prototyped a pipeline where we replicate/synchronize mobility hindrance data from Flanders for a WFS service within DUET.

2. Introduction

2.1 Objective of this document

Each government level uses its own different information system, but at the same time, citizens expect these government levels to adopt a user-centric approach and provide instant access to their data or to open government data. Take for example the local governments in Flanders, which provide over 800 products and services. To support their processes and service delivery, they use back-office applications from different software vendors. These domain-specific applications are organised as vertical processes, having their own data model, ensuring that the data can not be reused by other applications, resulting in data silos and no interoperability. If it is necessary that two of those applications must exchange data, the data must be translated and transformed, which is complex and expensive. To raise interoperability, Flanders started the Open Standards for Linked Organisations (OSLO) initiative to create new or align with existing data standards for different domains at a Flemish level. By developing these data standards, Flanders aims to provide a horizontal layer on top of the vertical data silos, making data easier to exchange between various systems.

The same principles can be applied to Digital Twins, as it should be possible to work with data coming from different parties (e.g. air quality), which is why it is important to reuse existing data standards or to adopt the process and methodology of OSLO to create a data model that is supported within the DUET community. Previously, however, it was only possible to develop data standards in Dutch. To use OSLO on an international level, it was necessary to extend our tooling with multilingualism. This document discusses how the OSLO Toolchain, can be set up and configured in different (international) environments.

Flanders has also the intention to create a Smart Data Space in the coming years. At the core, there is a need for a sustainable and cost-effective way to publish data. Therefore, Flanders fully commits to Linked Data Event Streams (LDES) and sees it as the core task of data publishers. By publishing data as an LDES, data consumers are in complete control to develop the functionality they need on top of the data set. This way of publishing data is also discussed in this document, along with a prototype that shows how to consume a dataset that is published as an LDES and is used by a WFS service within DUET.

2.2 Previous work

The process and methodology were already explained in a previous deliverable (D3.6 — OSLO Extensions for the Digital Twin), along with the experimental extension of the OSLO Toolchain, multilingualism. This deliverable builds on top of the previous and explains the further improvements that were made. In D3.6, LDES was already introduced and discussed at a high level. In this deliverable, we discuss LDES more in-depth, and how an LDES can be integrated into DUET. Deliverable D3.8 — “Twin Data Broker Specifications and Tools” also mentions OSLO and discusses it very briefly.

2.3 Structure of this document

This document provides a detailed description of how OSLO (the OSLO Toolchain) can be used on an international level to create data standards and discusses a new data publishing strategy called Linked Data

Event Streams (LDES). Besides the OSLO data standards, this document discusses other ICT standards that are used within DUET.

In section 3, the improvements of the OSLO Toolchain are discussed, along with the challenges to set up a new instance of the Toolchain in another environment. This section is concluded by discussing the different levels on which the OSLO Toolchain is already being used.

Section 4 discusses the novel data publishing strategy, LDES, and explains how an LDES was integrated in DUET. Section 5 gives a brief overview of other ICT standards within DUET.

3. Improvements of the OSLO Toolchain

In the first part of the DUET project the OSLO Toolchain has been extended with internationalisation capabilities. A first version has been deployed on https://duet.dev-vlaanderen.be/. During the next phase, the internationalisation has been integrated into the main branch of the OSLO toolchain. Additionally, improvements were made to make the OSLO Toolchain ready to be used by other parties. These contributions are documented in this section.

3.1. Challenges

Technically, the OSLO Toolchain is built as a CI/CD automation on top of GitHub repositories. Deploying a new OSLO Toolchain instance for another publication domain and other responsible parties could be summarised in one sentence: "Create two new GitHub repositories, then copy the CI/CD automation from an existing OSLO toolchain and adapt the CI/CD configs to the new publication domain."
Although technically this is a reasonably straightforward task, it does not guide the deploying partner.

It also indicates that the deployment process has to be mature to ensure others will adopt the OSLO toolchain. This includes among others:

- Version management: it should be clear which version of the OSLO toolchain the instance is running
- Version deployment: it should be possible to deploy a new version with minimal impact on the instance configurations
- Documentation: the design and documentation of the OSLO toolchain should be accessible and in English.

Another challenge encountered in the discussion is the need for working in a non-public environment. Sometimes one prefers to restrict access to the data specifications to a limited audience.

3.2. Realisations

To address the most critical aspects of version management and the documentation challenges, the generic building blocks for the OSLO toolchain are collected in a template repository. Using these templates, the deployment of a new toolchain is simplified and facilitated. The drawback of this solution is that when an existing instance wants to align with a new version, a smart merge has to happen. The template repository contains scripts that are being used by the CI/CD workflow. Whenever a new version of the toolchain is available, and an existing instance wants to update to that new version, there is a good chance that these scripts will have to be adjusted, among other things. This can be quite a time-consuming task, especially if the difference between the existing instance and the new version is quite large. In addition, when performing an update, a check must also be carried out for all specifications to ensure that the same result is obtained with the new version. To reduce the effort for the scripts, a solution based on CircleCI Orbs could be studied. Orbs are considered to be reusable snippets of code that help automate repeated processes and can be developed and managed in a central location. By using CircleCI Orbs, scripts no longer would have to be part of the template repository and therefore no longer have to be managed by a local instance. Performing an update would be tantamount to adjusting the version number of an orb in the CircleCI configuration.

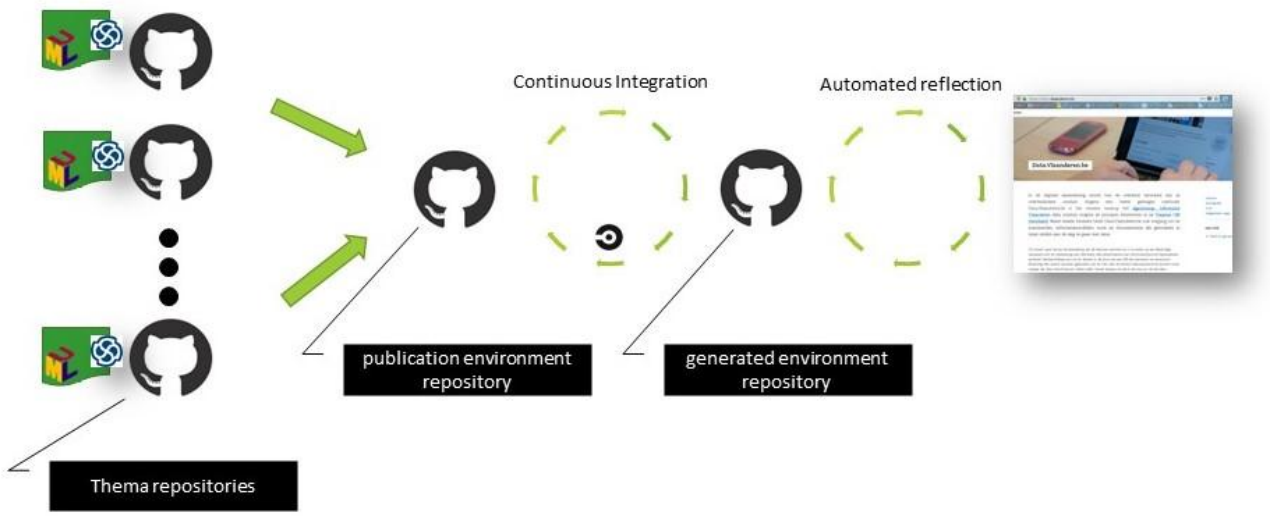

Figure 1 — Overview of the OSLO Toolchain. All data models (Enterprise Architect files) are stored in independent GitHub repositories. All publication configuration is stored in the publication environment repository. Making changes to this repository triggers the CI/CD and starts the workflow, which results in multiple artefacts being built, and stored in the generated repository. The repository containing the artefacts (static pages) is used by the proxy to serve it on the website.

The objective of an instance of the template is to automate the publication of the data specifications in a static website as shown in figure 1. The data specifications are managed in separate thema repositories. Separating the data specifications content from the publication environment creates flexibility and scaling potential, without losing central control. The content of the static website is available in the generated repository. To speed up configuration, we made template GitHub repositories for the theme repositories and the publication environment repository. These templates ensure that all needed information and structure is present as expected by the OSLO Toolchain and the CI/CD.

After the template has initiated a publication environment, the setup has to be completed with additional configuration. The process that must be followed to complete the configuration can also be found in the template repository.

3.2.1 Support for private repositories

As already mentioned above, access to data specifications can be restricted to a limited audience and can be obtained by making GitHub repositories private. There are two levels on which access to data specifications can be limited.

The first level that can be made private is the *generated environment* repository, which is used by the OSLO Toolchain to write the generated specifications to. On this level, extra configuration is needed for the CI/CD workflow and is described in the annex of this document. The second level on which access can be restricted are thema repositories. When a thema repository is private, then a similar approach as for the private generated environment repository should be followed. Again, this requires additional configuration for each thema repository that is private. The process that must be followed to process a private thema repository is also described in the annex of this document.

3.2.2 Generated specifications deployment support

The deployment of the static website containing the generated specifications, is independent from and beyond the configuration of the publication environment repository. The setup for data.vlaanderen.be is shown in figure 2. Everything is hosted in the cloud and as cloud provider, Azure was chosen. On top of the cloud provider, there is the infrastructure layer, and Terraform is used here. The Terraform configuration describes a docker swarm setup, which is responsible for the execution of the services of data.vlaanderen.be. Docker swarm also does health checks on every service. Whenever a service is malfunctioning, docker swarm will try to respawn the service to restore the application. The service itself is controlled by a docker-compose file, which acts as a neutral description of the application, and contains details of how the services are connected to each other. When adding new features to the service or fixing bugs, the code is pushed into the source control system (github.com and a local GitLab instance). The automation triggers a build to create a new docker instance of the service and also tests that build. When the build passes all checks, it can be committed to the master branch and tagged. Then, the tagged docker service can be activated in the docker-compose. Any changes made to the docker-compose are automatically deployed on the infrastructure. The use of branches and tags allows to precisely document which service is running on which environment. A more detailed description of each layer can be found on GitHub.

Take in mind that this setup for the static website is not mandatory. One is free to choose his own setup, and could for example choose to publish the static files through GitHub pages, which is free.

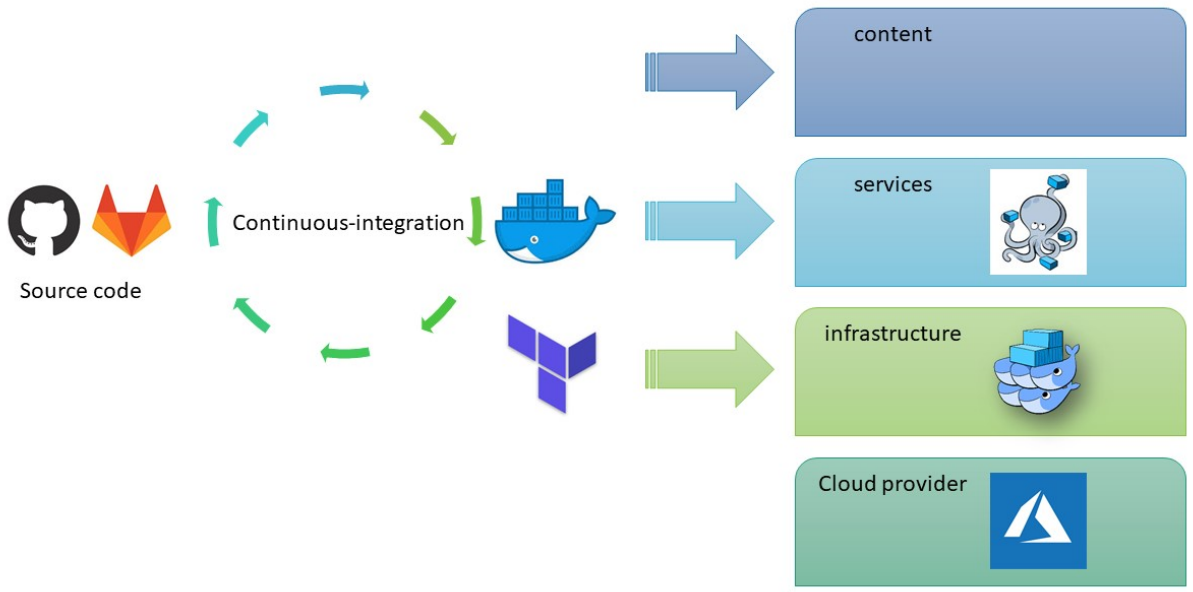

Figure 2 — Overview of the setup of data.vlaanderen.be.

3.3. Reuse of the OSLO Toolchain

Extending the OSLO Toolchain to support multiple languages to create data models, and simplify the setup and configuration, has ensured that the Toolchain can be (re-used) in different operational contexts. This section provides an overview of the usage since the contributions by DUET are made.

3.3.1 Reuse at the Belgian national level

At Belgian national level, the OSLO process and methodology were adopted by the Interfederal Collaboration for E-Government (ICEG) initiative, which is a cooperation agreement between the Belgian federal, regional, and community governments for the harmonization and alignment of the initiatives aimed at realizing integrated e-government. The domain on which the data models are published is https://belgif.github.io/thematic/models. At the time of writing this report, there are 2 standards published:

1. Public Service → Alignment with CPSV-AP (Core Public Service Vocabulary) which was designed to support the exchange of basic information about public services
2. Public Organisation → Alignment with CPOV (Core Public Organisation Vocabulary), and designed to support the exchange of basic information about public organizations

All information about the working groups (reports, presentations, models, …) is publicly available on GitHub: https://github.com/belgif/thematic.

Due to the multilingualism of the Toolchain, it can easily be used within European projects. For non-Flemish data standards, we do deviate from our standard domain on which we publish the standards and use a different domain, namely purl.eu. Already multiple standards have been published on purl.eu coming from different European projects:

- OSLO Air & Water → Data standard to exchange observations of air and water quality made by sensors. Was developed as part of the ODALA project, where the goal is to improve data management in cities.
- OSLO Passenger Transport Hubs → Data standard to exchange data about transport hubs. Was developed as part of the GreenMov project, and is based on the Flemish version, called OSLO Mobiliteit: Trips & Aanbod (Dutch).
- OSLO Consent → A data standard for supporting consent mechanisms and checking that data is used in the right way. The aim is to restore citizens' trust in the digital economy by implementing log integrity, non-repudiation and building data lineage and transparency by design. Three domains were considered within scope: government, telecom and the financial sector. This is done with the support of the TRAPEZE project.

The toolchain is also used by the AI Proficient project for expressing implementation models for exchanging data about manufacturing sensor readings. These semantic models are based on the OSLO Air & Water Core model and published on the project's own domain.

3.3.2 Reuse at the European level

The Semantic Interoperability Community Europe (SEMICeu) also adopted the OSLO Toolchain at the European level to generate the Core Vocabularies, such as Core Person Vocabulary, Core Location Vocabulary,

Core Business Vocabulary, and Core Public Organization Vocabulary. All specifications and generated artefacts are available on the GitHub of SEMICeu.

Recently, also Norway showed interest in reusing the OSLO Toolchain. Soon, the setup and configuration will be initiated, with technical support being provided by the OSLO team.

3.3.3 Reuse for implementation models

Previously, in Flanders the OSLO Toolchain was only used to generate vocabularies and application profiles. As discussed in the previous deliverable (D3.6), an application profile intends to be as generic as possible within its domain in order to create the greatest possible coverage/support. Today, more and more parties want to align themselves with existing OSLO standards, but the models are often too broad for their use case or certain concepts are missing for their implementation. For that reason, the toolchain is increasingly being used to develop implementation models. These implementation models are still aligned with the OSLO application profiles, but their scope is narrower and implementation-specific concepts can also be added. As these models are built for a specific implementation, the choice was made to not publish it on the official OSLO environment, data.vlaanderen.be, but on a subdomain or on a domain of the requesting party's choice. Once the domain has been chosen, a new instance of the toolchain is deployed to publish the specification(s) on the chosen domain.

3.4 Future work

The growing interest and adoption of the toolchain is an indication that many organisations and data ecosystems want to formalise their data exchange (publicly). The OSLO approach, of which the OSLO toolchain is one, yet a key aspect, has reached a maturity level that it can be adopted by others with a minimal effort. This creates a network effect, the more semantic agreements are published, the stronger the data networks become and thus the easier data can be shared.

Recently, the European Commission has launched the notion of *data spaces*: ecosystems with interoperable data exchanges. Within data spaces, the need for managing semantic agreements becomes even more important. The OSLO toolchain is ready to play a role here.

These different usage context will provide new ideas for future improvements. Some already identified topics are:

- Better integration with a SHACL data validation service
- Example generation in connection with a playground, to create an immediate live experience of the data standard
- An improved data standard registry: providing a more penetrable history of each data standard decision making history in the context of the whole development trajectory (past and future).
- An LDES event stream of the data standards published

4. Linked Data Event Streams

Today, data publishers often make their data available in one of two ways. The first way is through APIs that are hosted on their own servers. However, different data consumers often have different needs in terms of functionality. For example, one data consumer has a Geographic Information System (GIS) application, meaning that preferably he integrates with a WFS service provided by the data publisher, while another consumer likes to work with Linked Data and wants to integrate a SPARQL endpoint into his graph-based application. To meet the various desired functionalities of their consumers, data publishers have to continuously expand their existing APIs or host a completely new API. This ever-increasing number of functionalities/APIs means that data publishers have to spend more and more resources on keeping online/maintaining the functionalities/APIs, and results in a so-called *maintenance hell*. The second way is the complete opposite, as data publishers only make a data dump of their dataset available. The consumers can download the data dump and create the necessary functionality (API) on their own servers. This way of publishing data has pros and cons for both data publishers and consumers.

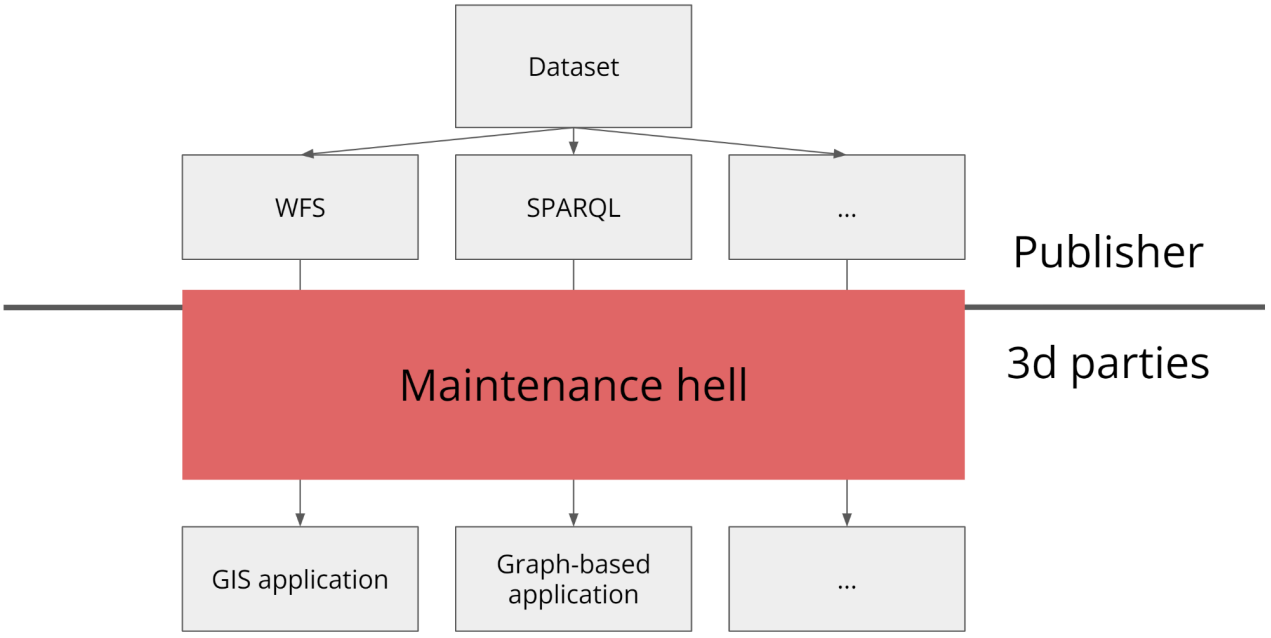

Figure 3 — By hosting every API possible, data publishers end up in a maintenance hell as they have to spend resources to maintain an increasing amount of APIs

On the one hand, the data publishers have fewer costs because they have to keep fewer APIs online and are able to focus more on publishing qualitative data (as data dumps). On the other hand, the data consumer has to put in more effort because he has to create the API on his own server, but he also has the complete freedom to develop all the functionality that is needed for his application as all the data is available for use.

However, the problem with publishing data dumps is that data consumers all too often have the idea that they only need to download the data once and then don't have to look at it again. Another common phenomenon is that data consumers download the data dump once and then start to make changes to their local data dump because this is apparently easier than having to download the entire data dump every time.

This results in local datasets being completely out of sync with the original dataset. To stay up to date with the dataset, consumers will have to periodically download the latest data dump. Having to download to entire data set as a data dump over and over again just to stay up to date, even if there are just a few changes, is not sustainable and cost-efficiënt (e.g. bandwidth), which is why this option is called a *replication hell.*

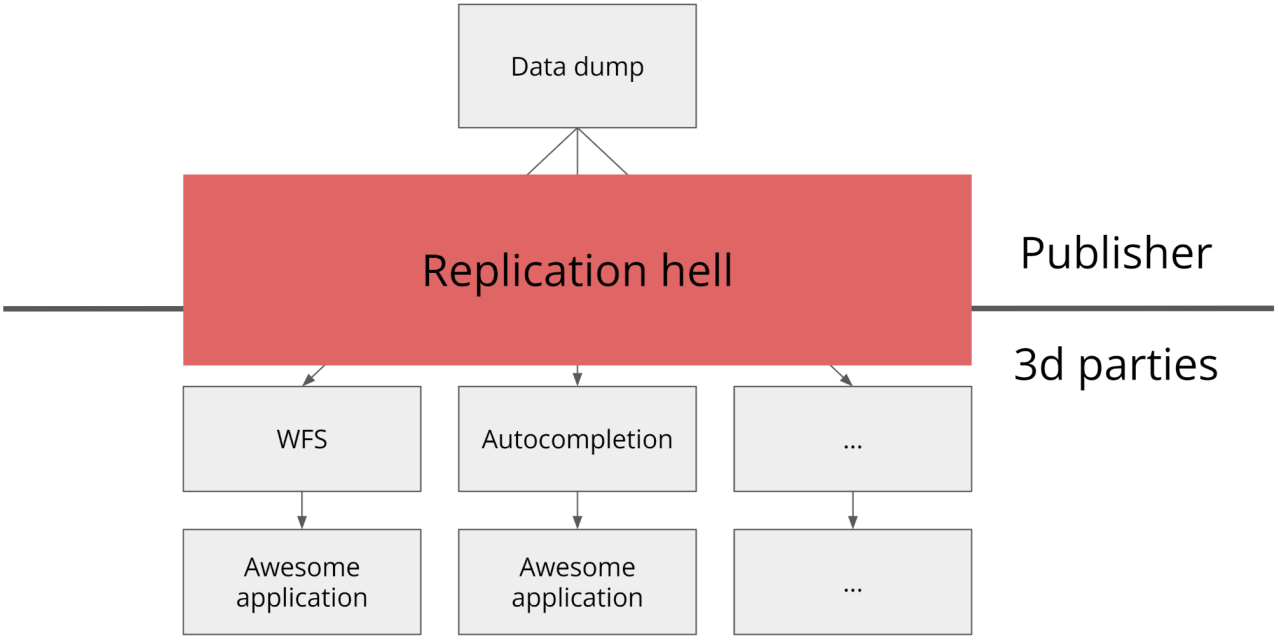

Figure 4 — By making data dumps available, the data consumers can create the needed functionality themselves, but need to regularly download the latest data dump to stay in sync with the dataset, which results in a replication hell.

To solve this problem, we introduced the concept of Linked Data Event Streams (LDES) in the deliverable D3.6 OSLO Extensions for the Digital Twin. With LDES, the data publisher ensures that all data is available as an 'event stream', including the history. This way, an automated replication/synchronization pipeline can be set up, such that the data consumer can always stay in sync with the data publisher's dataset. LDES is a European specification, available at https://w3id.org/ldes/specification.

Digitaal Vlaanderen also wants to be a pioneer in this, and this has been expressed in a new project called "Flemish Smart Data Space" (FSDS), which will run until 2024. This project is part of the plan called "Vlaamse Veerkracht" of the Flemish government and should help to strengthen the Flemish prosperity and well-being of the citizens after the pandemic. The intention of FSDS is to create a data space at the Flemish level in which as many datasets as possible are made available as Linked Data Event Streams. Multiple datasets have already been published as LDES, such as a prototype of the address registry, as well as mobility hindrance data coming from the Generic information platform public domain (GIPOD) and various datasets outside Digitaal Vlaanderen.

Another goal of FSDS is that the LDES specification not only becomes a standard at the European level but at the Flemish level as well by going through the OSLO trajectory. At the time of writing this report, OSLO-LDES is a candidate standard and is now in public review for a period of six months. Once this period is over, the necessary efforts will be made to make OSLO-LDES an officially recognized Flemish standard.

4.1 Mobility hindrance data in DUET

Within Flanders, the Generic Information Platform Public Domain (GIPOD) is a central exchange platform that brings together as much information as possible on operations and events in the public domain. GIPOD ensures more coordination between utilities and road works by optimizing information flows, making it easier to avoid operations on diversion routes and to detect/reduce conflicts between operations and events. One of the datasets that can be queried through GIPOD is mobility hindrance data, containing data about hindrances, and their location, along with additional information such as the period of the hindrance. In the context of the FSDS project, the mobility hindrance dataset was published as an LDES, making all mobility hindrances from the past, present, and future (upcoming mobility hindrances are also published) available. The LDES of mobility hindrances is available in beta at https://private-api.gipod.beta-vlaanderen.be/api/v1/ldes/mobility-hindrances. The mobility hindrances objects are also aligned with an OSLO data standard Public Domain Occupancy (only Dutch version available).

In the context of DUET, integrating the mobility hindrance dataset is an added value as it can be used to show the hindrances on a map or to calculate how much the impact is/will be on the environment (traffic, air quality, …). The figure below shows the workflow of how the GIPOD LDES will be integrated into the DUET environment. The different components of this figure are discussed below. All code is available at https://github.com/ddvlanck/gipod-ldes-demo.

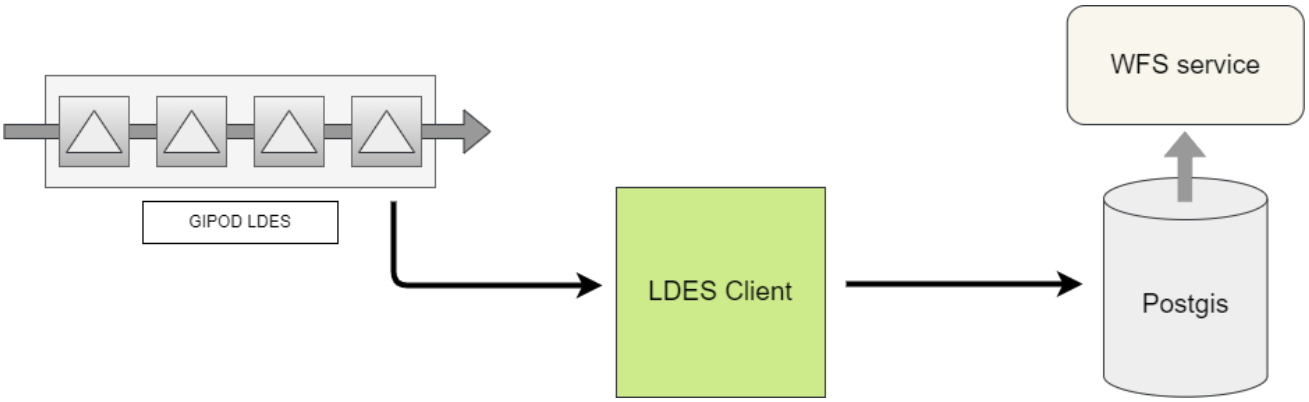

Figure 5 — Overview of the replication/synchronisation pipeline of mobility hindrance data in the GIPOD. By using the LDES client, we are able to get all mobility hindrances and insert them into a database, which acts as input for a WFS service within DUET.

An important part of the LDES specification that is not a part of this demo, but could potentially be relevant for DUET, are fragmentations. Fragmentations of an LDES could be considered the same as an index in the database, but then published on the Web. As with an index in a database, query performance will be higher for certain queries, by creating one or more fragmentations on top of an LDES. In general, fragmentations have two advantages, which we will discuss on the basis of a geospatial fragmentation, which, by the way, could be very interesting for the GIPOD mobility hindrances LDES. For a geospatial query, query clients would be able to gather their data much faster using the geospatial fragmentation compared to the basic LDES,

where a query client must iterate the whole LDES to be sure it has all the data that is needed to answer the question. The other advantage is that data consumers can synchronize with that part of the dataset that they are actually interested in. For example, if a Digital Twin is only interested in the mobility hindrances located in the area of Brussels, it only has to find the fragment/geospatial tile containing the mobility hindrances in Brussels and synchronize with that tile, while the other tiles can be ignored. This reduces the amount of data that must be replicated to the local backend. As stated, fragmentations are not part of the demo, which means that all the mobility hindrances (in Flanders) will be replicated to the backend. Of course, client-side filtering on the WKT string of the mobility hindrances could also be implemented but is beyond the scope of this report and demo.

4.1.1 The LDES specification and GIPOD LDES

The specification defines a Linked Data Event Streams as a collection of immutable objects, called *members*. Within a database, this means creating a new record when an object changes and inserting the changed object into the new record, instead of updating the existing record. All these records can be considered immutable, as they do not change anymore, because whenever something changes, an insert happens instead of an update. Take for example listing 1, which shows an LDES of street names in Turtle format (Linked Data serialization). To simplify the examples, the HTTP URIs which are normally used in Linked Data to uniformly identify objects are replaced with <C1>, a collection, and <streetname1>, a street name. This example shows that <streetname1> originally had the label “Station Road” and at some point, the label has been changed to “Station Square”. However, listing 1 is not compliant with the LDES specification, because the specification states that all members of an LDES must be immutable, and <streetname1> is not immutable in this example. The problem of mutable objects shown in listing 1 occurs for objects that do not know the concept of time, such as street names, addresses, and municipalities. However, an observation of a sensor does know the concept of time, because that observation was measured at a certain time and is only valid then. To solve this problem, the LDES spec states the version objects should be used, as shown in listing 2. So, for objects that do not understand the concept of time, versions of objects should be published, because versions are valid at a certain point in time and can be considered immutable. This means that listing 2 shows an example of an LDES of street name versions (<streetname1-v1> and <streetname1-v2> represent the HTTP URI of a street name version object). However, additional information is needed to indicate to which object a version belongs. The LDES specification leaves this up to the data publisher to decide how this extra information is added. Within Flanders, the decision was made to use *dcterms:isVersionOf*, which indicates of which object (<streetname1>) the described object (<streetname1-v1> and <streetname1-v2>) is a version. For example, the Netherlands have the NEN3610 standard, which acts as the base model to exchange geographic information, and in that standard, they use *foaf:isPrimaryTopicOf* to link an object to its versions.

```
<C1> a ldes:EventStream;
     tree:member <streetname1> .
<streetname1> rdfs:label "Station Road" .
<streetname1> rdfs:label "Station Square" .
```

Listing 1 — Invalid example in Turtle format of adding objects to an LDES that do not know the concept of time which would make them not immutable, which is against the LDES specification. **Note:** <...> represents an HTTP URI

```
<C1> a ldes:EventStream;
     tree:member <streetname1-v1> <streetname1-v2> .
<streetname1-v1> rdfs:label "Station Road" ;
                 dcterms:isVersionOf <streetname1> ;
                 dcterms:created "2022-01-01T00:10:00Z"^^xsd:dateTime .
<streetname1-v2> rdfs:label "Station Square" ;
                 dcterms:isVersionOf <streetname1> ;
                 dcterms:created "2022-01-10T:00:20:00Z"^^xsd:dateTime .
```

Listing 2 — Valid example in Turtle format showing an LDES of street name versions. For objects that do not know the concept of time, versions of objects must be published, because a version does understand the concept of time, and thus is immutable. **Note:** <...> represents an HTTP URI

The same principle applies to the GIPOD mobility hindrances as they also do not know the concept of time. Listing 3 shows a mobility hindrance version object and indicates (through dcterms:isVersionOf) that this version object, with id *https://private-api.gipod.beta-vlaanderen.be/api/v1/mobility-hindrances/10228530/3*, is a version of *https://private-api.gipod.beta-vlaanderen.be/api/v1/mobility-hindrances/10228530*. The most important part of this mobility hindrance is the geometry of the mobility hindrance and can be found at zone > geometry > wkt or by following the property path "(https://data.vlaanderen.be/ns/mobiliteit#zone http://www.w3.org/ns/locn#geometry http://www.opengis.net/ont/geosparql#asWKT)" on RDF level.


```
{
   "@id":"https://private-api.gipod.beta-vlaanderen.be/api/v1/mobility-hindrances/10228530/3",
   "@type":"MobilityHindrance",
   "gipodId":10228530,
   "identifier":[
      {
         "@type":"Identifier",
         "Identifier.identifier":{
            "value":"10228530",
            "type":"gipodId"
         },
         "assignedByName":"https://gipod.vlaanderen.be"
      }
   ],
   "isConsequenceOf":[
      {
         "@id":"/api/v1/works/3779167",
         "@type":"Work",
         "gipodId":3779167
      }
   ],
   "description":"8420 De Haan, Grotestraat 159: Stelling",
   "owner":{
      "isVersionOf":"/api/v1/organisations/fedab33f-792a-029c-9b34-9d9cfe7d6245",
      "@type":"Organisation",
      "preferredName":"EagleBe Smartcity"
   },
   "contactOrganisation":[

   ],
   "zone":[
      {
```

```
"@id":"/api/v1/mobility-hindrances/10228530/zones/991fa7c1-a530-44f7-a185-98ba45b69ce0",
         "@type":"Zone",
         "consequence":[

         ],
         "geometry":{
            "@type":"Geometry",
            "wkt":"<http://www.opengis.net/def/crs/EPSG/9.9.1/31370> POLYGON
((57626.8718169853 218733.768758968, 57626.8718169853 218731.768758968, 57639.8718169853
218731.768758968, 57639.8718169853 218733.768758968, 57626.8718169853 218733.768758968))"
         },
         "zoneType":{
            "@id":"/api/v1/taxonomies/zonetypes/0fb72ef7-6ac9-4a70-b295-a30ea215d250",
            "prefLabel":"HinderZone"
         }
      }
   ],
   "period":[
      {
         "@type":"Period",
         "start":"2021-01-14T05:00:00Z",
         "end":"2021-01-27T19:00:00Z"
      }
   ],
   "timeSchedule":null,
   "permittedBy":[

   ],
   "status":{
      "@id":"/api/v1/taxonomies/statuses/a411c53e-db33-436a-9bb9-d62d535b661d",
      "prefLabel":"Onbekend"
   },
   "generatedAtTime":"2020-12-28T09:36:09.72Z",
   "eventName":"MobilityHindranceWasImportedFromLegacy",
   "isVersionOf":"/api/v1/mobility-hindrances/10228530",
   "memberOf":"https://private-api.gipod.beta-vlaanderen.be/api/v1/ldes/mobility-hindrances",
   "lastModifiedOn":"2020-12-24T09:44:05.58Z",
   "lastModifiedBy":{
      "isVersionOf":"/api/v1/organisations/fedab33f-792a-029c-9b34-9d9cfe7d6245",
      "@type":"Organisation",
      "preferredName":"EagleBe Smartcity"
   },
   "createdOn":"2020-12-24T09:44:05.58Z",
   "createdBy":{
      "isVersionOf":"/api/v1/organisations/fedab33f-792a-029c-9b34-9d9cfe7d6245",
      "@type":"Organisation",
      "preferredName":"EagleBe Smartcity"
   }
}
```


Listing 3 — Overview of what a mobility hindrance version object looks like. The property *dcterms:isVersionOf* is used to indicate of what entity this is a version.

4.1.2 LDES Client

To retrieve all the immutable objects of an LDES, an LDES client was developed by IDLab (Ghent University), which understands the LDES spec and is able to traverse an LDES to get all the members. The source code is publicly available on GitHub.

4.1.3 PostGis and WFS service

The LDES client is integrated into code and it is possible to subscribe to a stream on which the LDES client puts the members of the LDES. The program subscribes to the stream on which the LDES client puts the members, receives them, and then puts them into a PostGis database. This database is used as input for the WFS service, which can be integrated into DUET to visualise the mobility hindrances on a map.

5. Minimal Interoperability Mechanisms (MIMs)

ICT standards are an integral component of the DUET architecture and play an essential role in the ongoing development and wider implementation of DUET. It is important that DUET complies with existing leading open ICT standards, including the MIMs, to make widespread take-up and adoption easier. DUET also has the opportunity to help shape the new standards that will be needed, which will benefit the move towards LDTs more generally and open up new opportunities for take-up of the DUET solution. The opportunity to help shape International Standards, will also help open up opportunities for European companies globally.

DUET relies on existing MIMs such as MIM1 Context Information Management, MIM2 Data Models, and MIM3 Data Ecosystem Management. It is feeding into some of the MIMs that are in the process of being developed, especially to MIM7 Geospatial, but also to MIM6 Security, MIM9 Analytics, and MIM10 Resource Management. MIM4 on Personal Data Management and MIM5 on Fair AI are also relevant.

DUET also implies/suggests a new MIM, namely Abstract Orchestration for the facilitation of orchestrating federated components in a data pipeline as described in DUET Deliverable 3.9 (Michiels & Vervaet, 2021). This suggestion is based on the concrete need for Digital Twin scenario management in all DUET pilot cases materialised by orchestrated and federated components.

Currently, DUET made use of the following existing standards:

Standard Name/abbr.	Description and role in DUET	Standardisation Organisation	DUET component use
DCAT	The Data Catalog Vocabulary is a W3C standard for describing datasets and data services in a catalogue. Within the Public Section Information domain, a European profile DCAT-AP is used to harmonise the descriptions to realise a network of (Open) Data portals throughout Europe. In DUET, DCAT is used as the vocabulary to describe the catalogue of datasets and data services the Digital Twin has access to or provides.	W3C, EC	Data Catalog
CityGML	Standard for exchanging city models. All city models of pilots are based on CityGml, to be visualised in the viewing component.	OGC	Viewing component
GeoJSON	GeoJSON is a format for encoding a variety of geographic data structures	IETF - RFC 7946	Viewing component

3D Tiles	3D Tiles are designed for streaming and rendering massive 3D geospatial content such as Photogrammetry, 3D Buildings, BIM/CAD, Instanced Features, and Point Clouds. It defines a hierarchical data structure and a set of tile formats that deliver renderable content. 3D Tiles does not define explicit rules for visualisation of the content; a client may visualise 3D Tiles data however it sees fit.	OGC	Viewing component
WMS	The OpenGIS® Web Map Service Interface Standard (WMS) provides a simple HTTP interface for requesting geo-registered map images from one or more distributed geospatial databases.	OGC	Viewing component
WMTS	WMTS complements earlier efforts to develop services for the web-based distribution of cartographic maps.	OGC	Viewing component
TMS	A Tile Map Service (TMS) provides access to cartographic maps of geo-referenced data, not direct access to the data itself.	OSGEo	Viewing component
WFS	This International Standard specifies discovery operations, query operations, locking operations, transaction operations, and operations to manage stored, parameterized query expressions on vector features.	OGC	Viewing component
Vector tiles	Vector Tiles are packets of geographic data, packaged into pre-defined roughly-square shaped "tiles" for transfer over the web.	Mapbox	Viewing component

Table 1 — DUET Overview of used ICT standards overview

6. Conclusion

To ensure that a Digital Twin is widely supported, interoperability is a key aspect that must be covered. Without uniform data models for the domains of which data must be integrated into the Digital Twin, with the necessary semantics attached to them, the Digital Twin will not be sustainable. With the Flemish Interoperability program, OSLO, we have shown that tooling is available to create semantically meaningful data models. To realise this, the OSLO Toolchain was extended with multilingualism, which provides the possibility to not only use the Toolchain in Flanders but also at an international level. On top of the tooling, the OSLO process and methodology provide a governance structure for the data models that can be used to develop data models that are widely supported within a Digital Twin community. In addition, a great effort has been made in providing documentation and templates that simplify setting up a new instance of the OSLO Toolchain in a different environment. Next to the data standards, we discussed other ICT standards that are being used by DUET, which will only simplify the adoption of DUET in various environments.

Data integration into a Digital Twin is also an important aspect that has been discussed throughout this report. When multiple data sources must be integrated into a Digital Twin, chances are high that custom code will need to be written for each of those APIs, in order to integrate them into the backend. We introduced a novel data publishing strategy, called Linked Data Event Streams, that offers advantages for both data publisher and consumer. Rather than facilitate any API possible to meet the needs of consumers or provide data dumps, data publishers should make their datasets available as an LDES. This results in a decrease of maintenance and hosting costs, which lets the data publishers focus on publishing qualitative data. This way of data publishing benefits the data consumers in a way that they have complete freedom to build any functionality possible on top of the dataset. Of course, custom code will need to be written, but will be reusable as we will be able to code against the LDES specification. The data consumer can setup an automatic replication/synchronisation pipeline to stay up-to-date with the dataset as demonstrated with the GIPOD mobility hindrances LDES. Whenever there is a need for extra functionality, the data consumer can just implement that functionality himself, which was also demonstrated by creating a WFS service on top of the GIPOD mobility hindrance LDES. However, as the data consumer now has to host APIs on his own environment, the associated costs are for him, which results in a trade-off that has to be made by the data consumer. Is he willing to pay more for extra functionality?

Another interesting opportunity that is also supported by LDES, is replaying certain scenarios. As an LDES is able to keep track of history, it will be easy to publish a time-based fragmentation of an LDES containing a collection of certain events (objects changing over time) and iterate it chronologically. For example, every time a road opens or closes, a new version of the road is added to the LDES, and that LDES can be used within simulations or to visualise the changing of closed roads on a map. This opportunity is yet to be explored.

In the future, from OSLO we will be working on re-writing the OSLO Toolchain to make it more flexible and configurable. Now, the OSLO Toolchain is a workflow that allows little configuration and is quite complex to extend with new functionalities. The goal is to let users configure their instance of the Toolchain according to their own taste. For example, the current version always generates multiple artefacts such as HTML pages of the data models. If a user is not interested in the HTML pages, he should be able to disable the generating of them, which is now not easily possible. Within OSLO, we also have the intention to create a GitHub

repository where the OSLO community can easily report bugs or make suggestions for the Toolchain. Doing so will allow us to interact and further expand the OSLO community. From the LDES perspective, as it is a key part of the Flemish Smart Data Space, multiple building blocks will be developed in the coming years that should facilitate/simplify the use of LDES for both the data publisher and data consumer.

7. References

1. OSLO — http://dx.doi.org/10.1145/3014087.3014096
2. DUET — https://www.digitalurbantwins.com
3. CircleCI — https://circleci.com/
4. Azure — https://azure.microsoft.com/nl-nl/
5. Terraform — https://www.terraform.io/
6. Linked Data Event Stream specification — https://w3id.org/ldes/specification
7. Linked Data Event Stream paper — https://link.springer.com/chapter/10.1007/978-3-030-74296-6_3
8. OGC (CityGML, 3D Tiles, WMTS, WFS, and WMS) — https://www.ogc.org/docs/is
9. DCAT (Data Catalog Vocabulary) — https://www.w3.org/TR/vocab-dcat-2/
10. MIMs (Minimal Interoperability Mechanisms) — https://oascities.org/minimal-interoperability-mechanisms/
11. GIPOD — https://overheid.vlaanderen.be/en/producten-diensten/generic-information-platform-public-domain-gipod

8. Annex

8.1 Additional configuration private generated environment repository

The generated environment repository is the repository in which the OSLO Toolchain will write the generated artifacts. If this repository is private, then one should create a deploy key as described in the documentation for github.com. The private key should be added to CircleCIconfig in the additional ssh keys, having "github.com" as hostname. The fingerprint of this private key should be placed in the .circleci/config in the create-artefact step configuration. This will insert that key into the create-artefact step. The public key should be installed as a deploy key with read/write rights on the 'Generated' repository in GitHub.

8.2 Additional configuration private thema repository

As stated above, a similar approach can be followed as for a private generated environment repository. However, due to the key insertion of CircleCI into a container (step), the following has to be considered:

- Create a new deploy key for the private 'Thema' repository
- Enable the public key as deploy key for the 'Thema' repository (read access is sufficient)
- Insert the private key in CircleCI project as additional ssh key with as (dummy) hostname Thema-private
- Enable the update of the ssh configuration for this (dummy) hostname Thema-private
- Use instead of https://github.com/<ORG>/<REPO>, git@Thema-private:<ORG>/<REPO>.git as the value of the repository in the configuration of a publication point.

These steps have to be executed for each private thema repository.

8.3 Background information on private repositories and deploy keys

1. https://docs.github.com/en/authentication/connecting-to-github-with-ssh/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent#generating-a-new-ssh-key
2. https://circleci.com/docs/2.0/gh-bb-integration/#best-practices-for-keys
3. Location within the CircleCI web app of the ssh key configuration for a project:
    a. Login into CircleCI
    b. Select the repository (called project in CircleCI web app)
    c. Go to project settings (button located on the right top)
    d. Select the tab ssh settings (in the menu on the right)
    e. Additional ssh key configuration is at the bottom of the page