

OSLO LDES:

Thematische werkgroep 1

Welkom!

Donderdag 20 januari Virtuele werkgroep – Microsoft Teams

We starten om 09:05



Doel van vandaag

Voorstelling sneuvelmodel en een overzicht van de bestaande standaarden die van toepassing kunnen zijn.



Samenvatting van de business werkgroep



Voorstelling sneuvelmodel & capteren van input a.d.h.v. interactieve oefening



Vastleggen van een basisdefinitie van een LDES en zijn kenmerken, nl.:

- Collectie
- Lid
- Structuur
- Versie van een object (OSLO Generiek)

Agenda

| | Welkom en introductie | 09:05 - 09:15 |
|---|---|--------------------------------|
| | Overzicht: wat hebben we gedaan in de vorige werkgroep? | 09:15 - 09:25 |
| ひ | Korte herhaling OSLO & LDES | 09:25 - 10:00 |
| | | |
| | Bestaande standaard: Semic | 10:00 - 10:25 |
| | Bestaande standaard: Semic Een eerste poging tot een data model: feedback en brainstorming | 10:00 - 10:25 10:35 - 11:50 |

Praktische zaken



Opname?





Welkom en introductie

Mural

Wie is wie?



Link in de chat:

https://app.mural.co/t/beadvtc7549/m/beadvtc7549/1642058003282/c4fc00efe 50465f570d5149b04acd586f51c6a39?sender=u048a1117151baed084666519



Samenvatting: Wat hebben we gedaan in de vorige werkgroep?

Wat hebben we gedaan in de vorige werkgroep?



OSLO introductie

- Semantische interoperabiliteit
- Technische interoperabiliteit
- Uitwisselen van data
- Hergebruiken van data





Introductie tot LDES

- Context: Vlaamse Sensor Data Space
- Wat is een LDES en waarvoor kan het gebruikt worden?



Brainstorm oefeningen

- Capteren van de verwachtingen
- Oplijsten welke problemen / verwachtingen eventueel opgelost kunnen worden door een LDES
- Welke terminologie dient opgenomen te worden in de specificatie?

Scope van het project

Ontwikkelen van een semantisch framework voor een Linked Data Event Stream

Omvat een duurzaam vocabularium en applicatieprofiel voor een Linked Data Event Stream

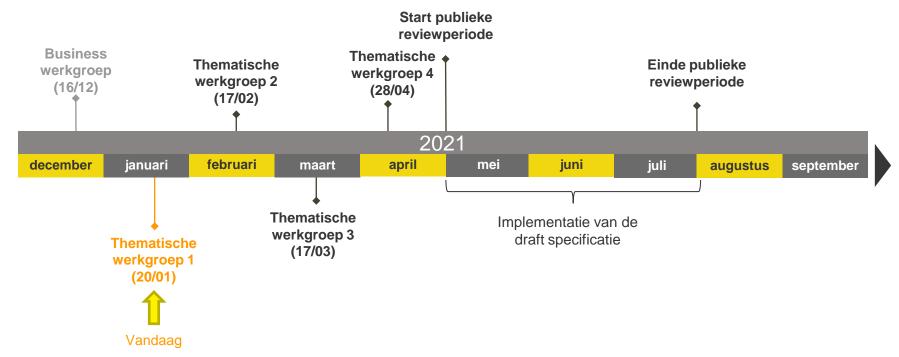
We volgen de OSLO methodiek, wat betekent dat:

- We starten van use cases
- We zoveel mogelijk aligneren met bestaande standaarden
- We zelf zaken definiëren waar nodig

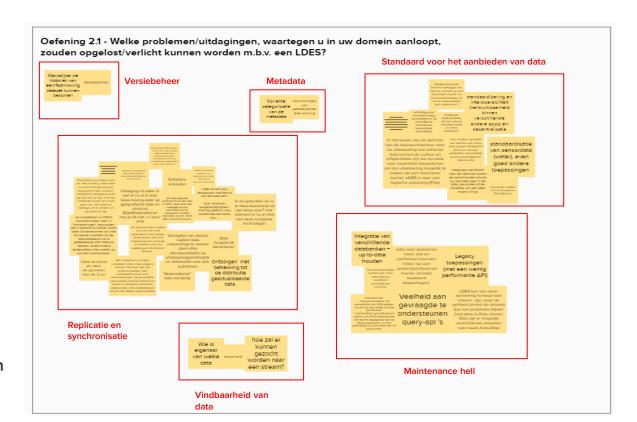
Overzicht van de planning

Startdatum: 16 december 2021

Duur: 10 maanden

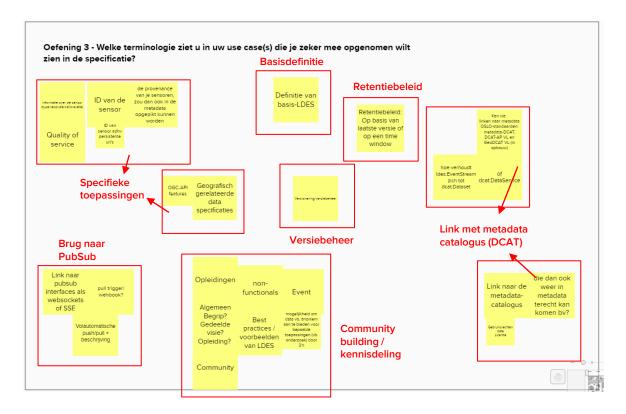


Welke problemen/uitdagingen zouden opgelost kunnen worden door een LDES?





Welke terminologie dient zeker mee opgenomen te worden?

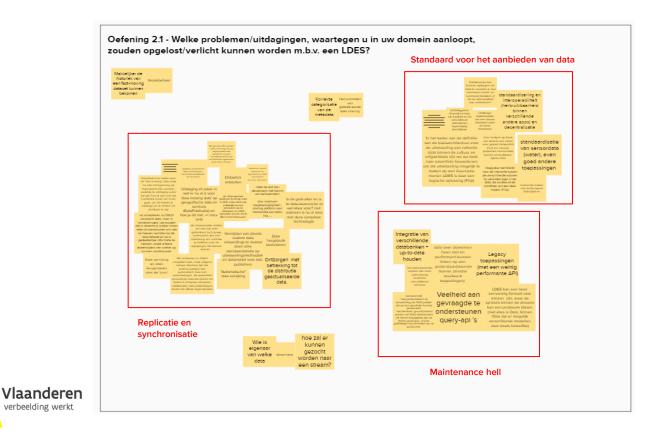




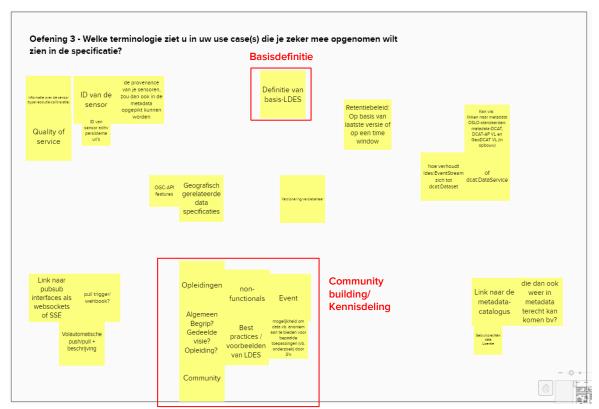
Korte recap OSLO & Linked Data Event Streams (LDES)

Thijs Hegge & Arne Van Der Stuyft

In dit deel focussen we op:

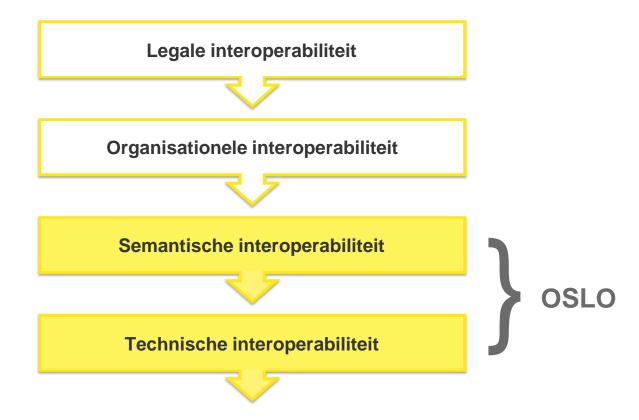


In dit deel focussen we op:





Levels van interoperabiliteit





OSLO



Semantische interoperabiliteit

Technische interoperabiliteit





Tools

Support & Governance

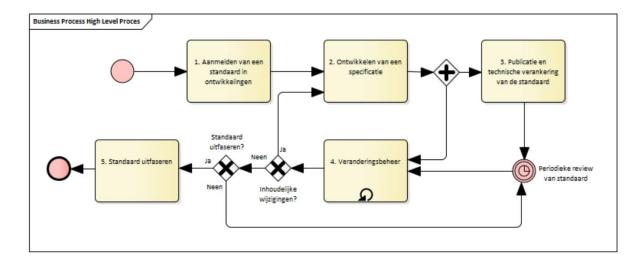




Use cases

Context van de werkgroepen

- → Werkgroepen kaderen binnen breder proces
 - Doel: Consensus rond data standaard gedragen door verschillende stakeholders
 - Proces en methode voor het ontwikkelen van een data standaard





Linked Data Event Streams

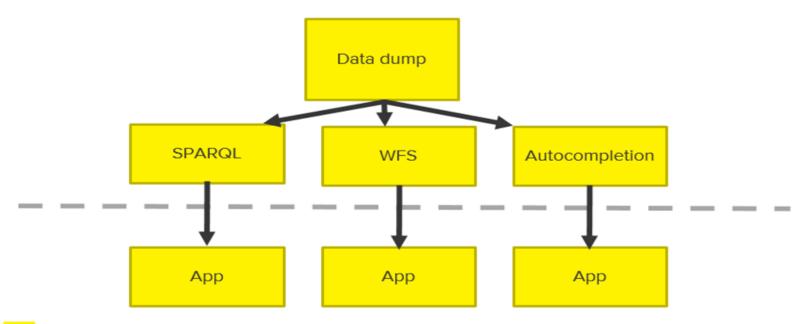
zorgen ervoor dat iedereen kan repliceren en in-sync kan blijven met objecten gedocumenteerd mbv Linked Data

Al een EU specificatie: https://w3id.org/ldes/specification

- Uit te breiden voor de "Vlaamse Data Space"
- Een LDES = een altijd groeiende collectie van objecten



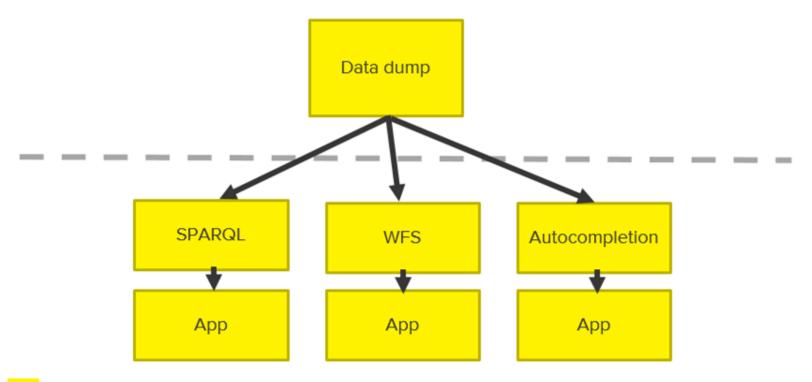
In plaats van een hele querying API





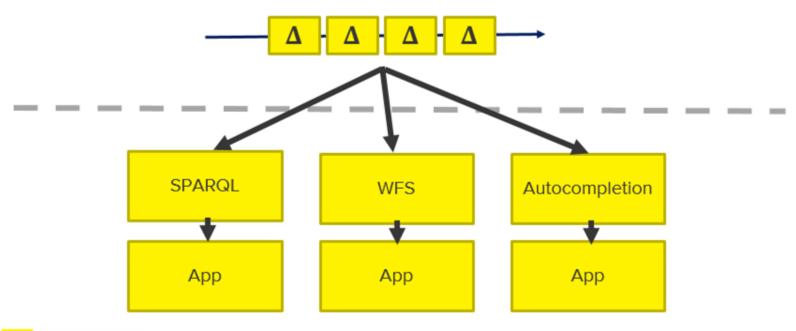
Ideale setting voor een "maintenance hell"

Of in plaats van een hele data dump





Een Linked Data Event Stream Blijft altijd in-sync





LDES: een high-level voorbeeld van sensorobservaties

```
<C1> a ldes:EventStream ;
     tree:member <0bservation1, <0bservation2>, <0bservation3> .
<Observation1> a sosa:Observation ;
                sosa:resultTime "2021-01-01T00:00:00Z"^^xsd:dateTime ;
                sosa:hasSimpleResult "..." .
<Observation2> a sosa:Observation ;
                sosa:resultTime "2021-01-01T00:10:00Z"^^xsd:dateTime ;
                sosa:hasSimpleResult "..." .
<Observation3> a sosa:Observation ;
               sosa:resultTime "2021-01-01T00:20:00Z"^^xsd:dateTime ;
               sosa:hasSimpleResult "..." .
     Vlaanderen
      verbeelding werkt
```

Use cases

Slow moving data

- Een adressenregister dat alle straatnamen van de gemeente beheert.
- Het basisregister bevat de basisinformatie (naam, geslacht, geboortedatum, ...) van alle inwoners van België. Een afgeleide LDES van het basisregister op nationaal niveau, is het basisregister van alle inwoners van Antwerpen.
- Het Generiek Informatieplatform Openbaar Domein (GIPOD) brengt alle informatie over (grond)werken, evenementen en hinder op het openbaar domein samen. Het platform zorgt ervoor dat er meer afstemming komt tussen nuts- en wegenwerken.

Fast moving data

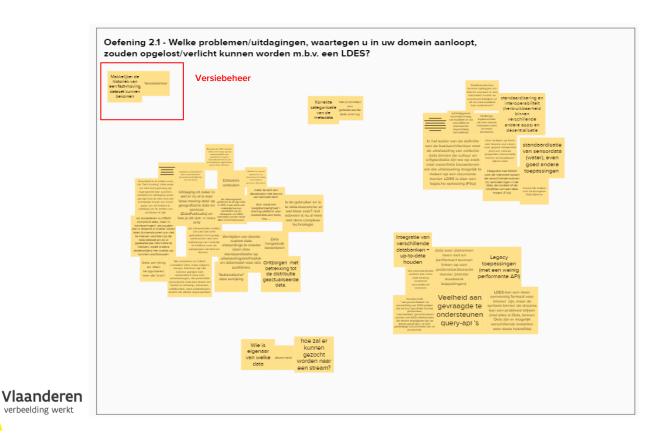
Voor de kwaliteit van de Schelde in de gaten te houden, wordt er op vaste tijdstippen d.m.v.
 sensoren observaties gemaakt inzake de waterkwantiteit.



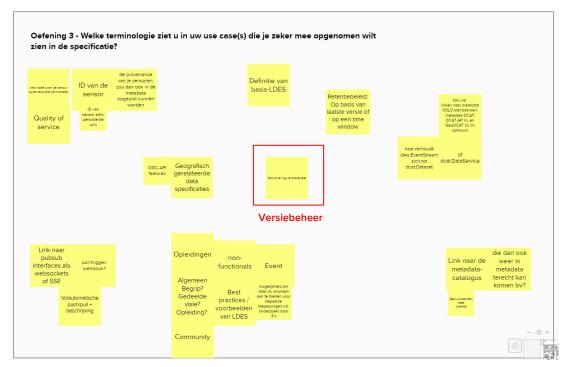
Zijn er nog andere use cases waaraan u denkt na deze sessie?

Gelieve dit <u>na de sessie</u> op de voorziene locatie (oefening 1) aan te geven in de <u>Mural</u>.

Welke problemen/verwachtingen hebben jullie?



Welke terminologie dient zeker mee opgenomen te worden?





Versiebeheer

van objecten



Voorbeeld: versiebeheer van straatnamen

→ Straatnamen zijn veranderlijke objecten, maar LDES verwacht onveranderlijke objecten → publiceren van <u>versies in de tijd</u> van een straatnaam!

```
"@id": "https://smartdata.dev-vlaanderen.be/base/straatnaam#64ef478f5c843021d05ebaa8c44a0769390ee7bf",
"isVersionOf": "https://data.vlaanderen.be/id/straatnaam/1142",
"generatedAtTime": "2021-09-07T15:48:51.851Z",
"created": "2011-04-29T13:34:14+02:00",
"eventName": "StreetNameBecameCurrent",
"memberOf": "https://smartdata.dev-vlaanderen.be/base/straatnaam",
"@type": "Straatnaam",
"straatnaam": [
      "@value": "Heirmanstraat",
      "@language": "nl"
"isToegekendDoor": "https://data.vlaanderen.be/id/gemeente/11002",
"homoniemToevoeging": "02",
"status": "https://data.vlaanderen.be/id/concept/straatnaamstatus/inGebruik"
```

We moeten echter een verwijzing hebben naar dit entiteit waarvan dit een versie is

→ Extra triple dct:isVersionOf met als waarde de persistente URI van de entiteit

```
"@id": "https://smartdata.dev-vlaanderen.be/base/straatnaam#64ef478f5c843021d05ebaa8c44a0769390ee7bf",
"isVersionOf": "https://data.vlaanderen.be/id/straatnaam/1142",
"generatedAtTime": "2021-09-07T15:48:51.851Z",
"created": "2011-04-29T13:34:14+02:00",
"eventName": "StreetNameBecameCurrent",
"memberOf": "https://smartdata.dev-vlaanderen.be/base/straatnaam",
"@type": "Straatnaam",
"straatnaam": [
      "@value":
               "Heirmanstraat"
      "@language": "nl"
"isToegekendDoor": "https://data.vlaanderen.be/id/gemeente/11002",
"homoniemToevoeging": "02",
"status": "https://data.vlaanderen.be/id/concept/straatnaamstatus/inGebruik"
```

Als mijn straatnaam wijzigt, bijvoorbeeld → wordt gehistoreerd

ightarrow Dan publiceren we een nieuw versie-object die een verwijzing bevat naar de entiteit

```
"@id": "https://smartdata.dev-vlaanderen.be/base/straatnaam#9027f0e43983893becfa19c125e80d1245568b46",
"isVersionOf": "https://data.vlaanderen.be/id/straatnaam/1142",
"generatedAtTime": "2021-09-07T15:48:51.882Z",
"created": "2012-01-31T14:59:14+01:00",
"eventName": "StreetNameWasRetired",
"memberOf": "https://smartdata.dev-vlaanderen.be/base/straatnaam",
"@type": "Straatnaam",
"straatnaam": [
      "@value": "Heirmanstraat",
      "@language": "nl"
"isToeqekendDoor": "https://data.vlaanderen.be/id/qemeente/11002",
"homoniemToevoeging": "02",
"status": "https://data.vlaanderen.be/id/concept/straatnaamstatus/gehistoreerd"
```



Versiebeheer: wat met observaties van bv: lochtkwaliteit?

→ Een observatie kan geen update ontvangen (is iets wat bestaat in de tijd)



Versiebeheer: conclusie

- Voor zaken die het concept van tijd niet kennen, zoals straatnamen, adressen, gemeentes, gebouwen, ...
 - worden versie-objecten gepubliceerd → die een verwijzing bevatten naar de entiteit waarvan ze een versie zijn (bv: dct:isVersionOf of foaf:isPrimaryTopicOf)

 Dingen die het concept van tijd wel al reeds kennen, zoals observaties, hebben die verwijzing niet nodig



Waarop hebben we ons gebaseerd?

Pieter Colpaert

Semic specificatie - Introductie

§ 1. Introduction

A Linked Data Event Stream (LDES) (ldes:EventStream) is a collection (rdfs:subClassOf tree:Collection) of immutable objects, each object being described using a set of RDF triples ([rdf-primer]).

The Ides: EventStream instance SHOULD have these properties:

- tree:shape: the shape of the collection defines its members. It tells clients all old and new members of the stream have been and will be validated by that shape. As a consequence of the immutability of the members, this shape MAY evolve, but it MUST always be backwards compatible to the earlier version.
- tree:member indicating the members of the collection.

The Ides:EventStream instance MAY have these properties:

- Ides:timestampPath indicating how you can understand using a timestamp (xsd:dateTime) a member precedes another member in the LDES
- Ides:versionOfPath indicating the non-version object (see example bellow).

Semic specificatie - tree:Collection

§ 1. Collections

The TREE specification introduces these core concepts:

- a tree:collection is a collection of elements that adhere to a certain shape. It typically has these properties
 when described in a node:
 - tree:member indicates the object is a member of the collection
 - tree:view indicates a root node from where all members can be reached
 - tree:shape indicates the SHACL [SHACL] shape to which each member in the collection adheres

Een LDES van sensor observaties

```
<C1> a ldes:EventStream ;
      tree:member <0bservation1, <0bservation2>, <0bservation3> .
 <Observation1> a sosa:Observation ;
                sosa:resultTime "2021-01-01T00:00:00Z"^^xsd:dateTime ;
                sosa:hasSimpleResult "..." .
<Observation2> a sosa:Observation ;
               sosa:resultTime "2021-01-01T00:10:00Z"^^xsd:dateTime ;
               sosa:hasSimpleResult "..." .
<Observation3> a sosa:Observation ;
               sosa:resultTime "2021-01-01T00:20:00Z"^^xsd:dateTime ;
               sosa:hasSimpleResult "..." .
```

Semic specification - Retentiebeleid

§ 3. Retention policies

By default, an LDES MUST keep all data that has been added to the tree:Collection (or Ides:EventStream) as defined by the TREE specification. It MAY add a retention policy in which the server indicates data will be removed from the server. Third parties SHOULD read retention policies to understand what subset of the data is available in this tree:View, and MAY archive these member.

In the LDES specification, two types of retention policies are defined which can be used with a ldes:retentionPolicy with an instance of a tree:View as its subject:

- Ides:DurationAgoPolicy: a time-based retention policy in which data generated before a specific time is removed
- 2. 1 des:LatestVersionSubset a version subset based on the latest versions of an entity in the stream

Different retention policies MAY be combined. When policies are used together, a server MUST store the members as long they are not all matched.



Wees transparent: een tijd-gebaseerd retentiebeleid

```
<C1> a ldes:EventStream;
    tree:view <> .
<> ldes:retentionPolicy <P1> .

<P1> a ldes:DurationAgoPolicy;
    tree:path prov:generatedAtTime;
    tree:value "P1Y"^^xsd:duration . # Keep 1 year of data
```



Opmerking: het is deze specifieke view die een retentiebeleid heeft, niet de LDES

Wees transparant: een op 'aantal versie-objecten' gebaseerd retentiebeleid

```
<C1> a ldes:EventStream;
    tree:view <> .
<> ldes:retentionPolicy <P1> .

<P1> a ldes:LatestVersionSubset;
    ldes:path prov:generatedAtTime;
    ldes:amount 2;
    ldes:versionKey ( dcterms:isVersionOf ) .
```



Semic specificatie - Versie materialisatie

§ 4.1. Version Materializations

A version materialization can be defined only if the original LDES defines both ldes:versionOfPath and ldes:timestampPath.

A version materialization replaces the subject of a member with its Ides:versionOfPath IRI, and selects a certain version of this object. It also translated created style timestamp predicates to modified-style predicates.

A version materialization is thus a tree:collection instance that has two predicates set:

- Ides:versionMaterializationOf: points to the orginal LDES
- Ides:versionMaterializationUntili optionally gives a timestamp (xsd:dateTime) until when the materialization was made.

Note: We see versionMaterializationUntil mainly useful for historical and static datasets that deliberately will not be updated to the latest state of the LDES.



Oplossing 1a: Gebruik enkel een referentie

De stad Gent is een tijdloos concept en zal niet veranderen zolang je LDES in gebruik is.



Oplossing 1b: Gebruik een versie specifieke referentie

Als de beschrijving van Gent toch zou kunnen veranderen en die beschrijving werd in een andere LDES gedocumenteerd



Oplossing 2a: Includeer een geversioneerd object

Wanneer je de beschrijving van de stad Gent onder versie controle wenst te houden van deze LDES

```
<C1> a ldes:EventStream ;
    tree:member <streetname1-v1>, <streetname1-v2> .
<streetname1-v1> rdfs:label "Station Road"@en ;
                 dcterms:isVersionOf <streetname1> ;
                 prov:wasAttributedTo <CityOfGhent-v1> ;
                 dcterms:created "2020-01-01T00:10:00Z"^^xsd:dateTime .
<CityOfGhent-v1> rdfs:label "Ghent"@en ;
                 dcterms:isVersionOf <CityOfGhent> ;
                 dcterms:created "2020-01-01T00:10:00Z"^^xsd:dateTime .
```



Oplossing 2b: Includeer een Blank Node in RDF

Wanneer je de correcte versie identificator niet kan vinden op de LDES creatie tijd

```
<C1> a ldes:EventStream ;
    tree:member <streetname1-v1>, <streetname1-v2> .
<streetname1-v1> rdfs:label "Station Road"@en ;
                 dcterms:isVersionOf <streetname1> ;
                 prov:wasAttributedTo :CityOfGhent-v1 ;
                 dcterms:created "2020-01-01T00:10:00Z"^^xsd:dateTime .
:CityOfGhent-v1 rdfs:label "Ghent"@en ;
                 dcterms:isVersionOf <CityOfGhent> ;
                 dcterms:created "2020-01-01T00:10:00Z"^^xsd:dateTime .
```



Een kleine pauze...





Welkom terug!





Een eerste poging tot een datamodel

Feedback en brainstorming

In dit deel focussen we op





Replicatie en synchronisatie

Bestaande API's bieden soms een hoge query expressiviteit aan waardoor er geen schaalbaarheid is om rechtstreeks applicaties op te laten draaien. Mogelijkheld om datastromen statische, quasi commercieel Undates automatisch aan te bleden laten doorstromen statische en (verantwoordelijkheid dynamische data van de bron) Datasilo's consistent Voordeel is er zeker voor kunnen afnemen de 'fast moving' data waar ontsluiten nu niet snel genoeg op meta: te kort aan Ingespeeld kan worden; Uitdaging zit zeker in developers met kennis waarbij de uitdaging weer wat er nu al is voor van semweb tech zal zijn hoe je dan met die als data-eigenaar 'slow moving data' op metadata ervan om moet publiceer ik al miin data Is de gebruiker en is dus: nood aan geografische data en gaan om dit helder in in API's maar wil ik ook le data-leverancier er makkelijk kunnen laagdrempelighied + catalogi om te zetten om services aansluiten op de sharing platform voor wel klaar voor? niet vindbaar te zijn (DataPublicatie) en dataspace en LDES colaboratie aan tools. edereen is nu al mee standaard zonder dat ik wii accepteren nu OSLO hoe ie dit ziet. => once libs. ... met deze complexe alles moet herbouwen compliant data, maar in only technologie 'aanleveringen', we zouden als dataverdeler stellen die in streams al sneller willen we vast dat veel laten binnenstromen om niet gebruikers toch graag te hoeven wachten op de Vermiiden van steeds vasthouden aan een hele dataset en zo al Data custom data datadump om controle gedeeltelijke informatie te heraebruik onboardings te moeter te hebben over de hebben, zodat andere bevorderen wijzigingen die binnen doen dmv stakeholders hier sneller op komen. standaardistatie op kunnen voortbouwen uitwisselingsmethodiek We ontsluiten nu OSLO en datamodel voor data Ontzorgen met Data verriiking compliant data, maar volgens en laten publishers betrekking tot dumps, hierdoor zijn die teruqvloeien externe partilen niet de distributie "Automatische" naar de 'bron' automatisch mee met veranderingen, die potentiele data verrijking geactualiseerde verouderde data kan leiden tot data. fouten in ontwerp, uitvoeren, collaboratie, data dubbelingen, assets die elkaar tegenspreken

Standaard voor het aanbieden van data

Dataleveranciers kunnen opleggen om data te voorzien in een standaard model, en standaardisering en eventueel bekiiken of dit de data kwaliteit interoperabiliteit kan verbeteren? (herbruikbaarheid achterliggende binnen brondata te laag Challenge: Opportunity verschillende Implementation niversele manier van van kwaliteit en via verschillende van een nieuwe werken voor tal var andere apps) en standaard naast standaarden allerhande domeine de reeds tegenstrijdig decentralisatie en sectoren beschikbaar bestaande In het kader van de definitie door analyse op basis van diverse use cases standaardisatie van de basisarchitectuur voor over gepast dataprofie de uitwisseling van collectie (met evt. nieuwe van sensordata properties sensordata) data binnen de cultuur en (water), even komen tot bruikbare erfgoeddata zijn we op zoek riikere data goed andere naar essentiele bouwstenen om die uitwisselina moaeliik te toepassingen Integratie met SOLID maken op een duurzame voor de interactie tussen manier. LDES is daar een de verschillende actoren bii veranderingen in de logische oplossing (Filip) data de context of de Connectie maken condities van een deel

traject. (Filip)

Maintenance hell

Integratie van verschillende databanken + up-to-date houden

> Via standaardisatie jestandaardiseerde werken aan meer data sharing overheen verschillende sectoren

formaten-hell: aan productiekant: de verwachting van 1000 partijen dat we hun specifieke formaat produceren aan leeskant: geconfronteerd worden met 1000 dataformaten die slechts begrijpbaar zijn via (beste geval spec, of zeer

gebrekkige documentatie van de

producent)

Veelheid aan gevraagde te ondersteunen query-api 's

data over domeinen

heen vlot en

performant kunnen

linken op een

manier. (minder

maatwerk

koppelingen)

performante API)

Legacy

toepassingen

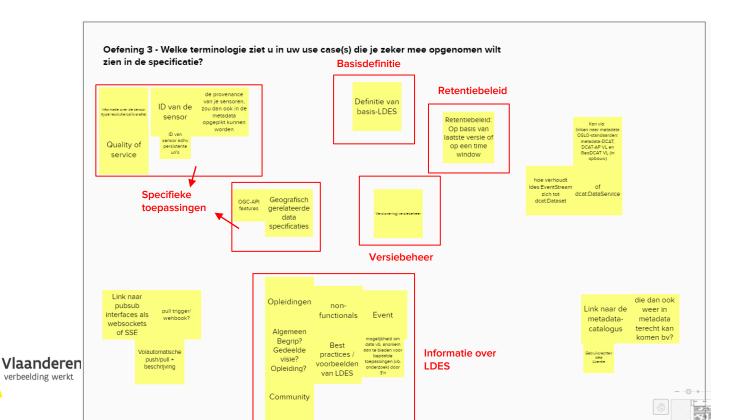
(met een weinig

LDES kan een meer eenvormig formaat voor inlezen ziin, maar de syntaxis binnen de streams kan een probleem blijven (niet alles is Oslo, binnen Oslo zijn er mogelijk verschillende modellen voor deels hetzelfde)

met de Europese

Data Spaces

Welke terminologie dient zeker mee opgenomen te worden?



Oefening: sneuvelmodel



Link in de chat:

https://app.mural.co/t/beadvtc7549/m/beadvtc7549/1642058003282/c4fc00efe 50465f570d5149b04acd586f51c6a39?sender=u048a1117151baed084666519



Praktische afspraken!

- Voor dit deel zullen we werken met drie breakout rooms, in combinatie met de plenaire ruimte.
- De oefening werd onderverdeeld in 3 delen, per deel hanteren we de volgende structuur:
 - o 5' toelichting van het datamodel/de terminologie
 - 10' discussie in de breakout rooms
 - 10' om gezamenlijk de feedback te consolideren
- ledereen mag post-its met feedback plaatsen in het vak van je respectievelijke breakout room; zet a.u.b. wel graag je naam op de post-its voor een efficiënte consolidatie.

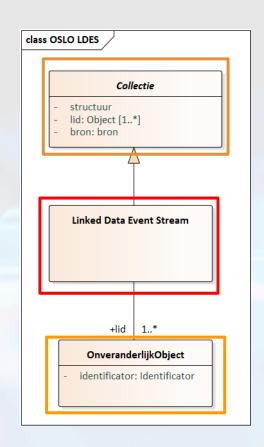


Basisdefinitie LDES

Een Linked Data Event Stream (LDES) is een collectie van onveranderlijke objecten (zoals versie-objecten, sensor-observaties of gearchiveerde representaties).

• Lid LDES: zie slide 55

https://w3id.org/ldes/specification

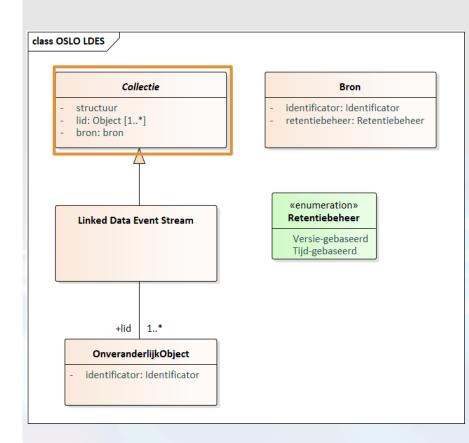


Collectie

Een collectie is een verzameling van objecten die een bepaalde structuur naleven. Elke collectie wordt gepubliceerd op één of meerdere servers (de bron(nen)).

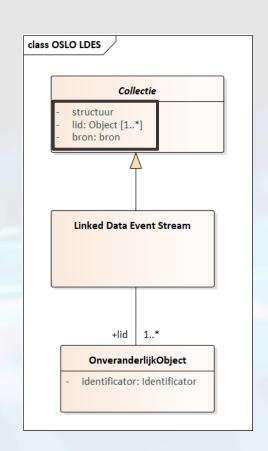
Een collectie volgt een bepaalde vastgelegde structuur, en heeft een aantal (één of meerdere) leden.

https://treecg.github.io/specification/#collection



Collectie

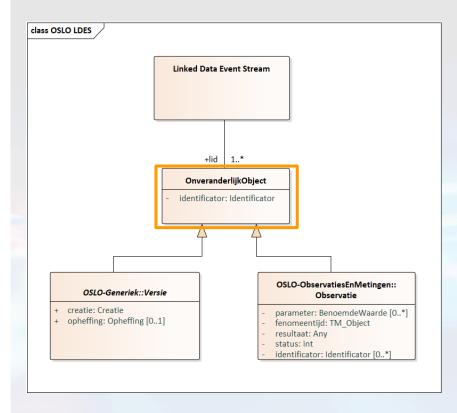
- De 'bron' duidt de bron (i.e. server) aan waarop alle leden geraadpleegd kunnen worden.
- De 'structuur' van een collectie definieert de vormvereisten van de leden. Het geeft weer welke oude en nieuwe leden zijn en gevalideerd zullen worden door deze structuur. Als een gevolg van de onveranderlijkheid van de objecten, KAN deze structuur veranderen, maar MOET deze altijd achterwaarts compatibel zijn.
- Het 'lid' geeft weer welke
 OnveranderlijkeObjecten deel uitmaken van
 de collectie (en wordt later gespecificeerd
 naar welke onveranderlijkeObjecten deel
 uitmaken van de LDES).



OnveranderlijkObject

Een OnveranderlijkObject maakt deel uit van een LDES (en dus van een collectie).

 De 'identificator' bevat Informatie die gebruikt kan worden om een object uniek te identificeren

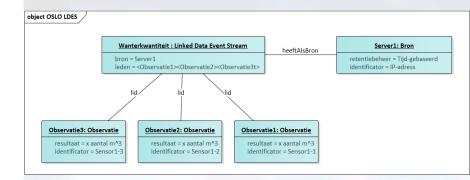


Een use-case

Fictief!

Door middel van een sensor (sensor 1) wordt de waterkwantiteit in een waterobject, bv. de Schelde, gemeten over verschillende tijdstippen. De sensorobservaties worden in de vorm van een LDES ontsloten.

De verschillende observaties vormen de leden van deze specifieke LDES. De LDES wordt bijgehouden op server1 en heeft een tijd-gebaseerd retentiebeleid (na 1 jaar zullen de observaties van de server verwijderd worden).

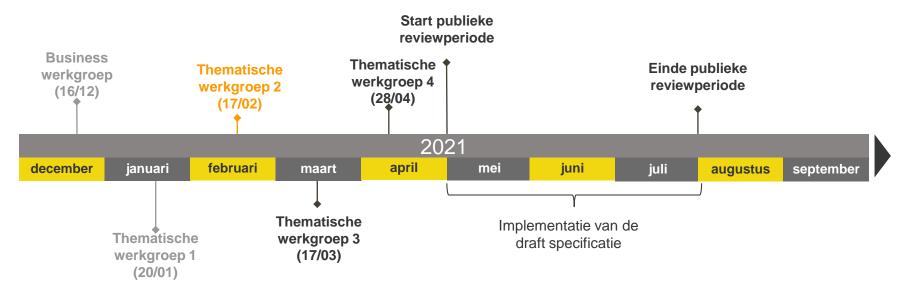




Volgende stappen

Thematische werkgroep op 17 februari (9h00-12h00)

Vergeet niet in te schrijven! https://overheid.vlaanderen.be/informatie-vlaanderen/agenda/thematische-werkgroep-2-oslo-ldes



Volgende stappen – in de tussentijd...



Herzien van het model en indienen van feedback/vragen via mail of GitHub.



Herzien van de definities en indienen van feedback/vragen via mail of GitHub.



Zijn er nog andere use cases (zoals op slide 25) waaraan u denkt na deze sessie? Gelieve dit op de voorziene locatie (oefening 1) aan te geven in de Mural.



Feedback & samenwerking



Feedback kan per e-mail worden gegeven aan de volgende personen:

- oslo@vlaanderen.be
- arne.vanderstuyft@vlaanderen.be



Feedback/input kan gegeven worden via GitHub:

https://github.com/Informatievlaand eren/OSLOthema-Ides/issues

Via het aanmaken van issues

Vragen?





Bedankt!