Best of Both Worlds

Lazo Bozarov (Gemeente Utrecht)
Joeri Bekker (Maykin Media)

DiS Geo
Doorontwikkeling in Samenhang

Agenda

- Introductie — Administratieve wereld en geo wereld bij elkaar

- ‘Waarom?’ — door DiS Geo (Bart-Jan de Leuw)
- ‘Hoe?’ — door Gemeente Utrecht (Lazo Bozarov)
- ‘Wat?’ — door Maykin Media (Joeri Bekker)

- Interactie — vragen via Menti
  in gesprek over jullie vragen en voorbeelden ☺

Waarom?

- In de wereld van de overheid (“Administration”) willen we meerwaarde van locatie uit de geo wereld benutten

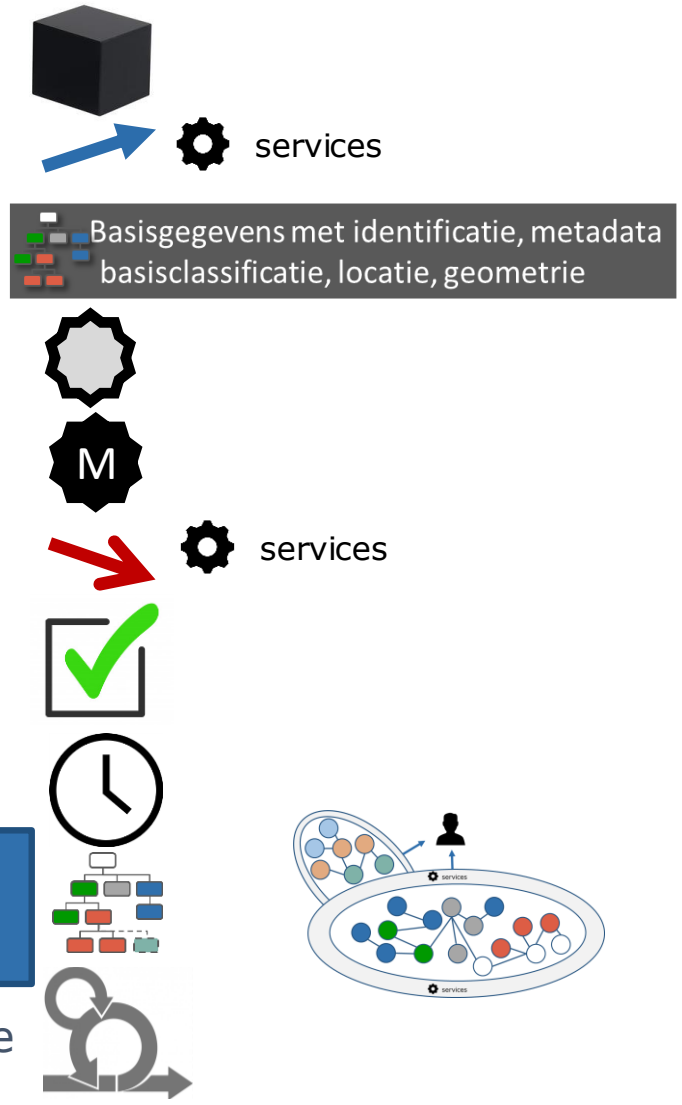
- We willen geodata als stroom uit het stopcontact gebruiken
- We willen geodata combineren met andere data uit alle toepassingsgebieden binnen de overheid: “geo meets admin”

- Architectuurvisie DiS Geo schetst een toekomst
- Bart-Jan de Leuw vertelt…

“Als stroom uit het stopcontact”

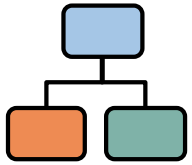
Waarden omtrent basisgegevens op een rij:

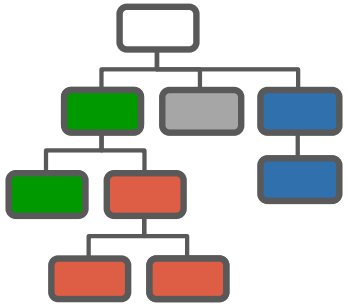
1. Basisgegevens zijn van en voor iedereen
2. Basisgegevens zijn laagdrempelig beschikbaar en bruikbaar voor iedereen
3. Basisgegevens voldoen aan vereisten
4. Bronhouders zijn verantwoordelijk voor basisgegevens
5. Bronhouders kunnen leveranciers machtigen
6. Gegevens aanpassen kan makkelijk en goed
7. Gebruikers en bronhouders werken samen aan de kwaliteit van gegevens
8. Basisgegevens zijn zo actueel en volledig als redelijkerwijs mogelijk
9. **Gegevens passen bij elkaar: relaties tussen gegevens zijn voor gebruikers duidelijk, en gegevens zijn in samenhang bruikbaar**
10. De gegevensstructuur kan snel genoeg meegroeien met de gebruiksbehoefte

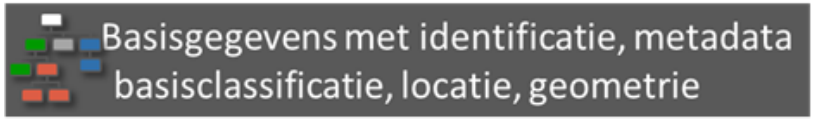


Gegevens passen bij elkaar en bij andere gegevens: relaties tussen gegevens zijn voor gebruikers duidelijk, en gegevens zijn in samenhang bruikbaar

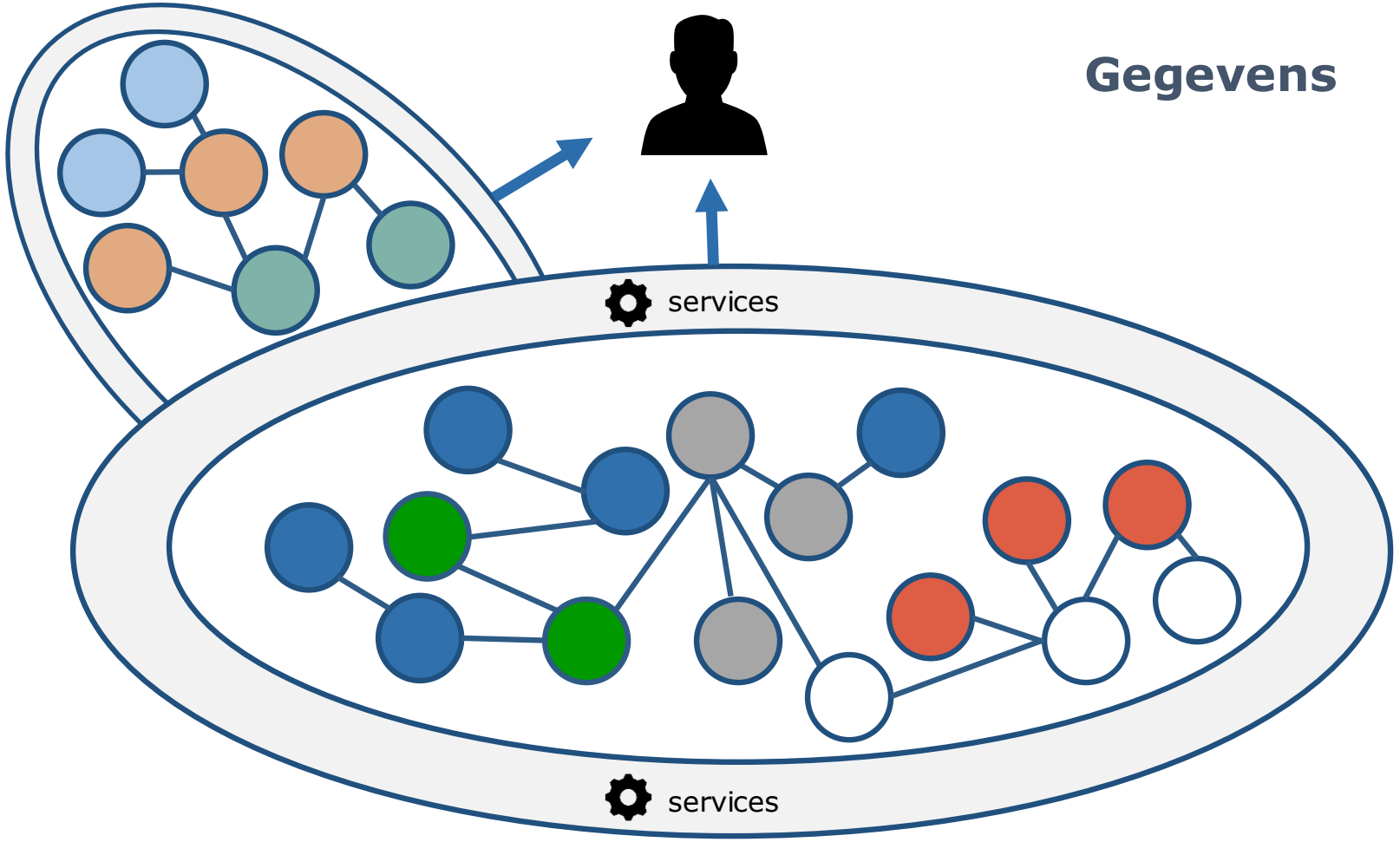
Spelregels





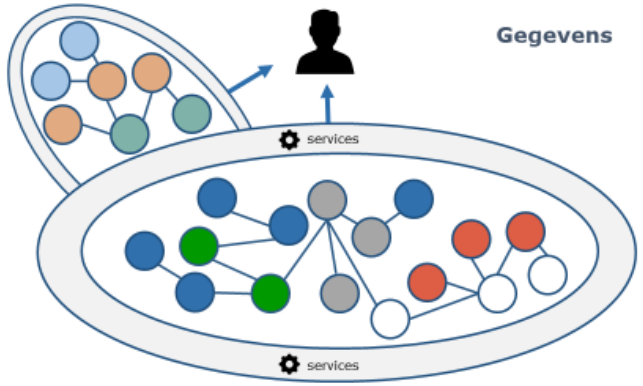


Gegevens



Vraag

- Ga naar www.menti.com met code: 7275970
- **Welke (geo) databronnen zijn belangrijk?**



Voorbeeld: Voor klimaatadaptatie van de stad

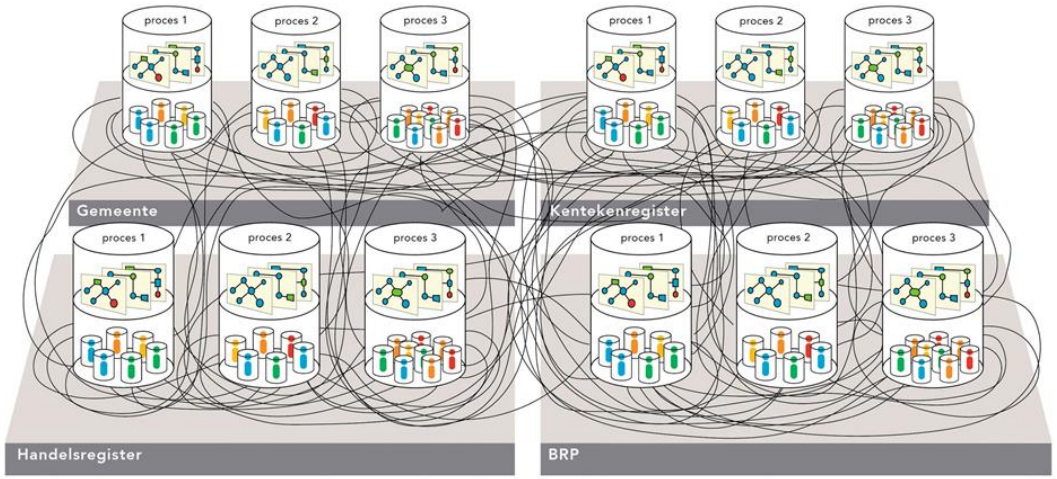
- Gegevens stadsbomen
  - locatie
  - leeftijd
  - groeisnelheid
  - capaciteit CO_2 -> O_2
  - waterbehoefte
- Contactgegevens boombeheerders
- … etc.

Hoe?

- Door: Lazo Bozarov
- Van: Gemeente Utrecht

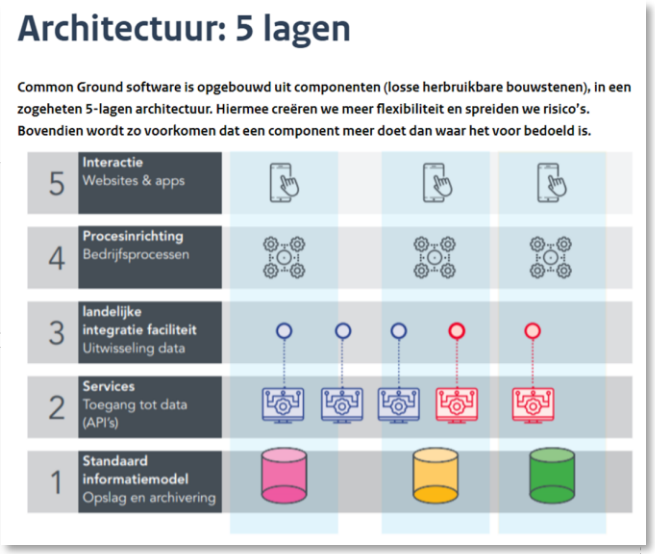
- Common Ground: herbruikbare open source componenten maken
  - Die iedereen kan gebruiken en doorontwikkelen
- Nuttig en nodig om zelf stopcontacten te kunnen maken op je datasets
- Dit doet Gemeente Utrecht met behulp van Maykin Media

Enthousiast over Common Ground



Gemeente Utrecht

Utrechtse aanpak



API

Applicatie

Laag 5
Interactie

Laag 4
Proces

Laag 3
Gegevens uitwisseling

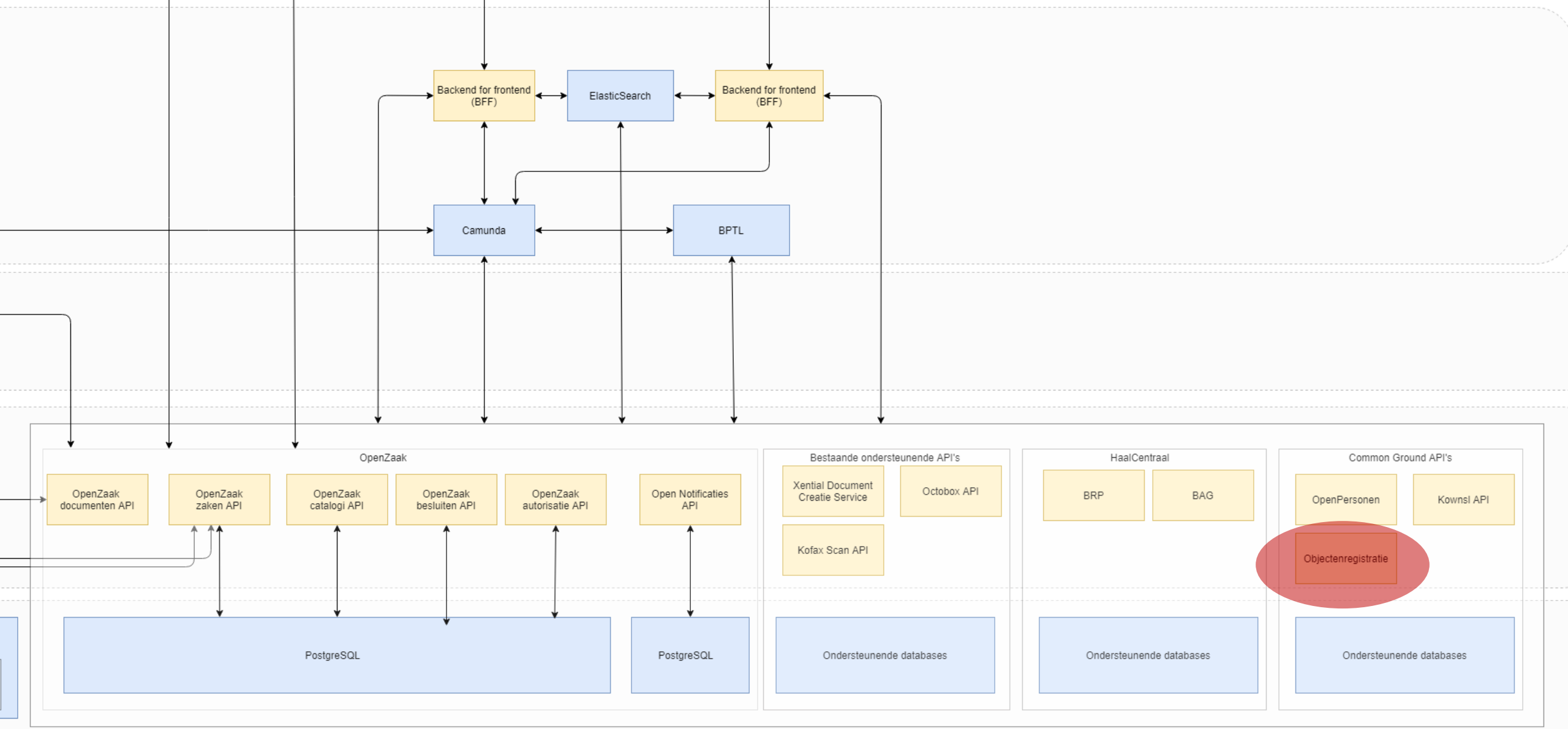
Laag 2
Services

Laag 1
Gegevens

Tezza (importmodule + zoeken)
OpenForms (prototype)
iBabs
iBabs Jail
Archief-Vernietigings-component
Generieke Zaakafhandel-component
Externe Inzage component
Backend for frontend (BFF)
ElasticSearch
Backend for frontend (BFF)
Camunda
BPTL
Contezza Documenten API
Alfresco content services API (OpenAPI en open standaarden)
OpenZaak
OpenZaak documenten API
OpenZaak zaken API
OpenZaak catalogi API
OpenZaak besluiten API
OpenZaak autorisatie API
Open Notificaties API
Bestaande ondersteunende API's
Xential Document Creatie Service
Octobox API
Kofax Scan API
HaalCentraal
BRP
BAG
Common Ground API's
OpenPersonen
Kownsl API
Objectenregistratie
Alfresco Storage)
Dossiers
DRC opslag
PostgreSQL
PostgreSQL
Ondersteunende databases
Ondersteunende databases
Ondersteunende databases

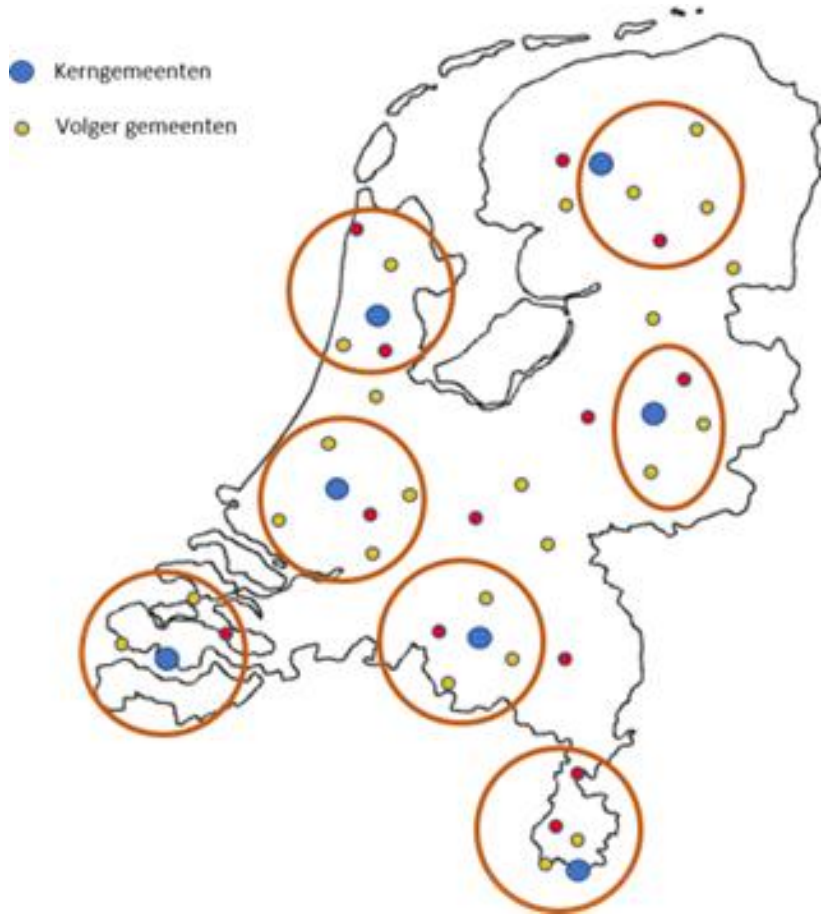
Laag 0
Infra HAVEN (Linux Kubernetes etc)

Gemeente Utrecht



Gemeente Utrecht

Succesfactoren samenwerking



- Open E-formulieren
- Record Management
- Objecten & Objecttype API
- Open Zaak
- OpenZaak Brug

- Open Personen
- Generieke Zaak afhandel component
- Huwelijksplanner
- Haven en BIO compliant orkestratie platform
- NLx

Gemeente Utrecht

Vervolg

- Vandaag bekijken we met elkaar de API’s en hoe die kunnen helpen om geo objecten en admin zaken bij elkaar te brengen

Gemeente Utrecht

Vraag

- Ga naar www.menti.com
- Voer de code in: 7275970

We zijn benieuwd naar:

- **Wat is jouw functie/rol?**
- **Was je al bekend met Common Ground?**

Wat?

- Door: Joeri Bekker
- Van: Maykin Media

- In opdracht van gemeente Utrecht
- API ontwikkeld waarmee je je eigen stopcontact kan maken
- In actieve (door)ontwikkeling
- Hoe het werkt met voorbeeldgegevens
  - Nog niet met gebruiker, dat komt op 30 maart

Achtergrond

- Project komt oorspronkelijk uit “zaakgericht werken” hoek
  - Alles bij een gemeente is een zaak (vergunningsaanvraag, melding openbare ruimte, schuldhulpverlening, etc.)

- Elke zaak is gekoppeld aan een of meer “objecten”, bijv:
  - Een *kapvergunning* wordt afgegeven voor een *boom*
  - *Melding* van een kapotte *laadpaal* moet behandeld worden

- Applicaties kunnen omgaan met (zaken en) gerelateerde objecten

Definitie: Objecttype (binnen deze context)

"Een in JSON-schema vastgelegde definitie van een object tezamen met metadata. Elk objecttype vertegenwoordigt een verzameling objecten met vergelijkbare vorm en/of functie."

Dit zijn de **spelregels**.

Voorbeelden:

"Boom" (), "Melding" (), "Vordering", "Laadpaal"

Definitie: Object (binnen deze context)

"Een op zichzelf staand geheel van gegevens met een eigen identiteit"

Dit zijn de **gegevens**.

Voorbeelden:

"Boom UTR-1958396" (🌳),

"Melding AMS-2020-000015" (📣),

"Vordering 550e8400-e29b-41d4-a716-446655440000"

"Laadpaal SolarPole-000005"

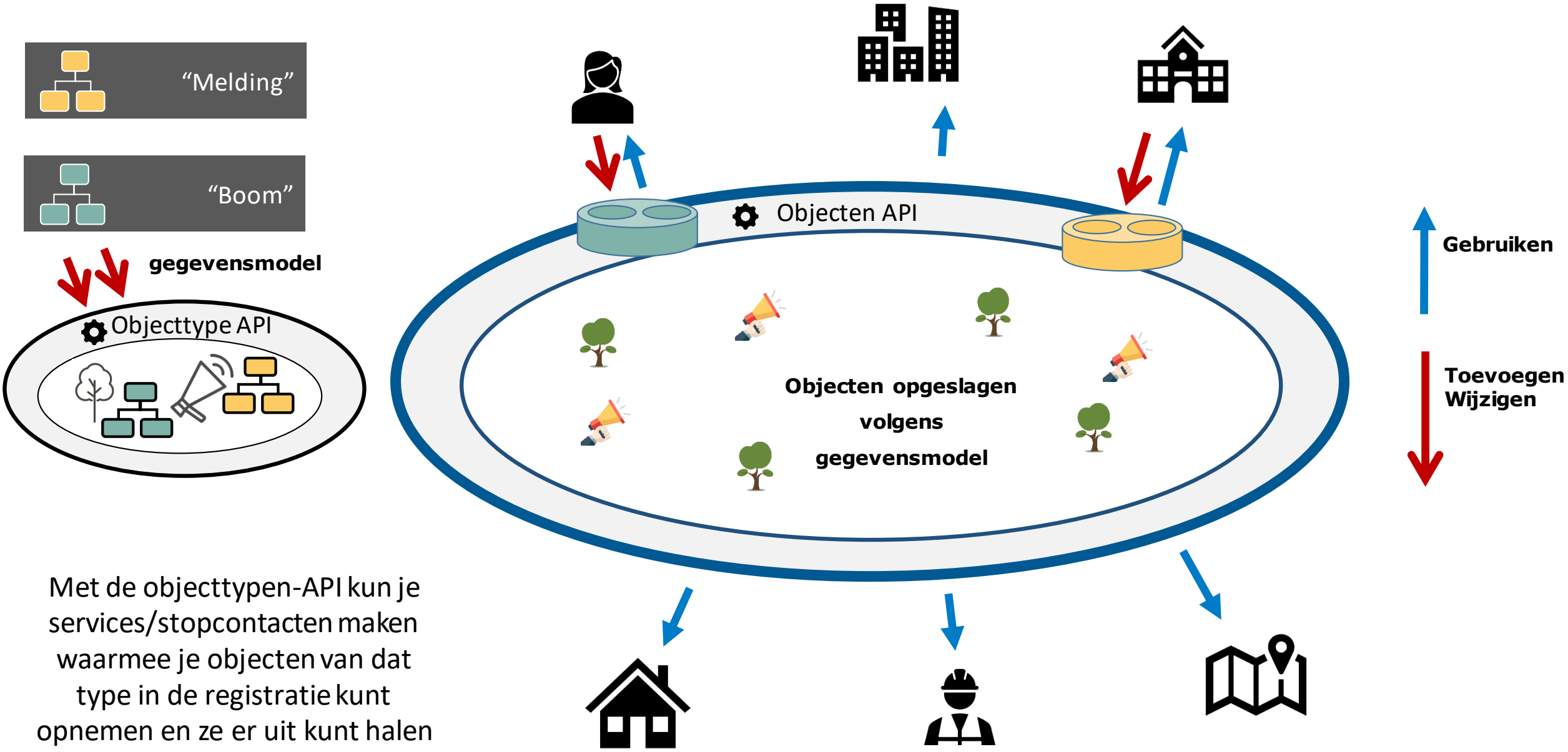
Objecttypen API en Objecten API

- Een **Object** (*gegevens*) wordt opgeslagen en ontsloten volgens het bijbehorende **Objecttype** (*spelregels*).
- Applicaties kunnen Objecten gebruiken via de **Objecten API** (*stopcontact*).
- Applicaties weten hoe een object er uit ziet door de **Objecttypen API** te raadplegen.

Wat kan je met de Objecten en Objecttypen APIs?

- Allerlei type objecten (**objecttypen**) definiëren
- Direct allerlei **objecten** registreren en ontsluiten
- Objecten zijn **open-** of **niet-open** data, **geo** of **niet-geo**
- Zowel **eigen** objecttypen als **landelijke** objecttypen gebruiken
- Sneller **standaardiseren** en direct **in productie** gebruiken

“Objecten als stroom uit het stopcontact”



Met de objecttypen-API kun je services/stopcontacten maken waarmee je objecten van dat type in de registratie kunt opnemen en ze er uit kunt halen

Waarom Objecttypen en Objecten als API?

- Functioneel beheer op je gegevens landschap wordt mogelijk
  - Functioneel beheerders kunnen zelf nieuwe objecttypen toevoegen
- Geen aparte realisatie/implementatie van een nieuwe API
  - Geen ontwikkelbureau nodig voor elk nieuw objecttype
- Data wordt toegankelijk: Democratisering van de data

Vraag

- Ga naar www.menti.com
- Voer de code in: 7275970

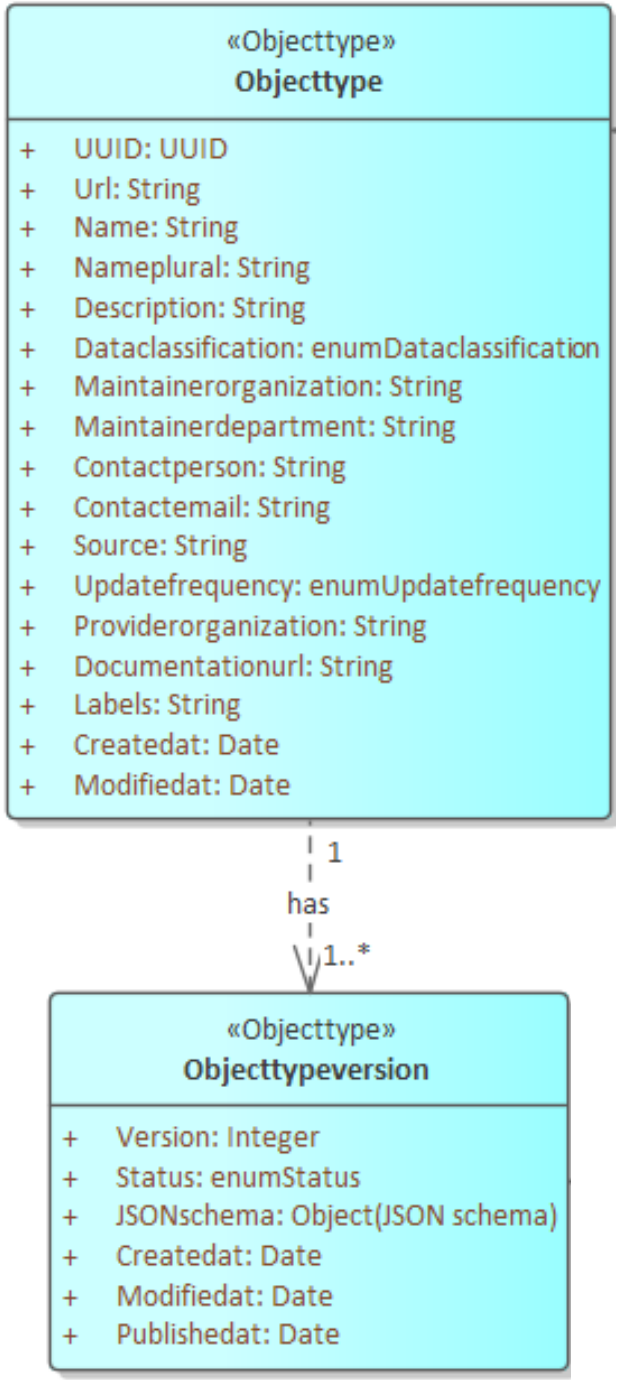
- **Welke gegevens zijn kandidaat om te worden ontsloten op de “Common Ground” manier?**

Objecttypen API features

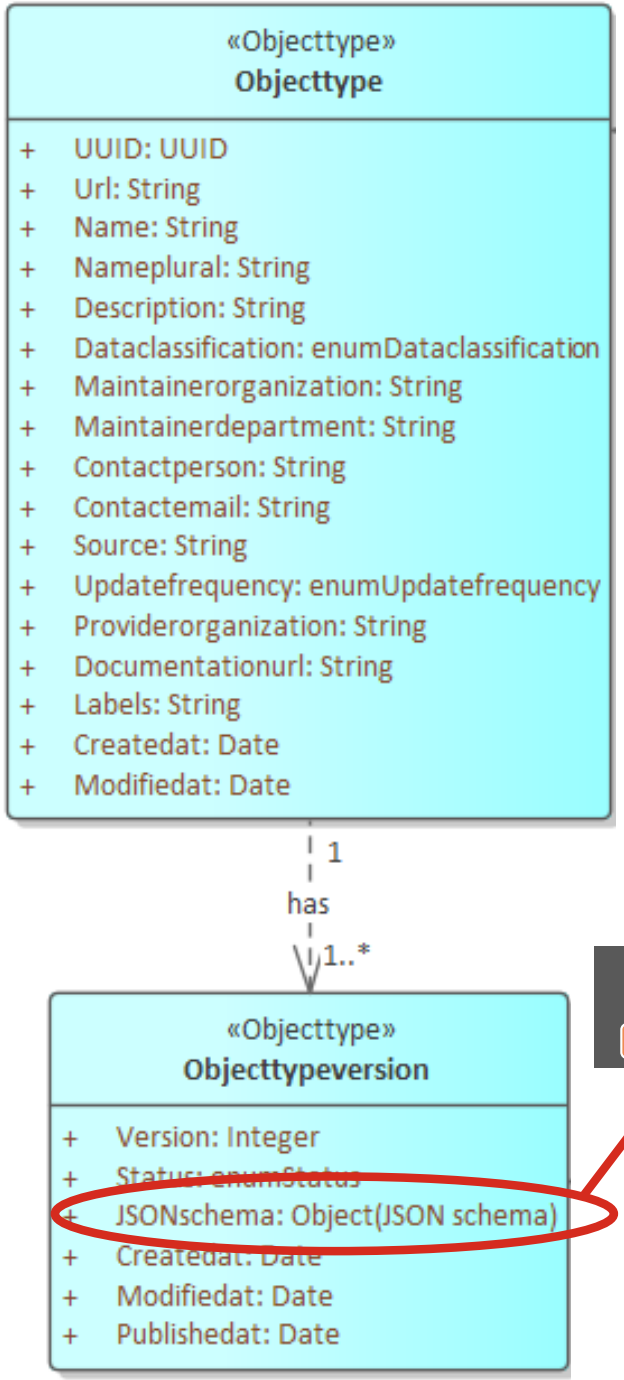
- Beheerders:
  - Objecttypen aanmaken
  - Objecttypen bewerken zolang deze niet gepubliceerd is
  - Objecttypen verwijderen zolang er geen gepubliceerde versies zijn.
  - Nieuwe versies van een Objecttype opvoeren

    En eventueel oude versies depreciëren
- Voor iedereen beschikbaar (lees-rechten)

IM: Objecttypen



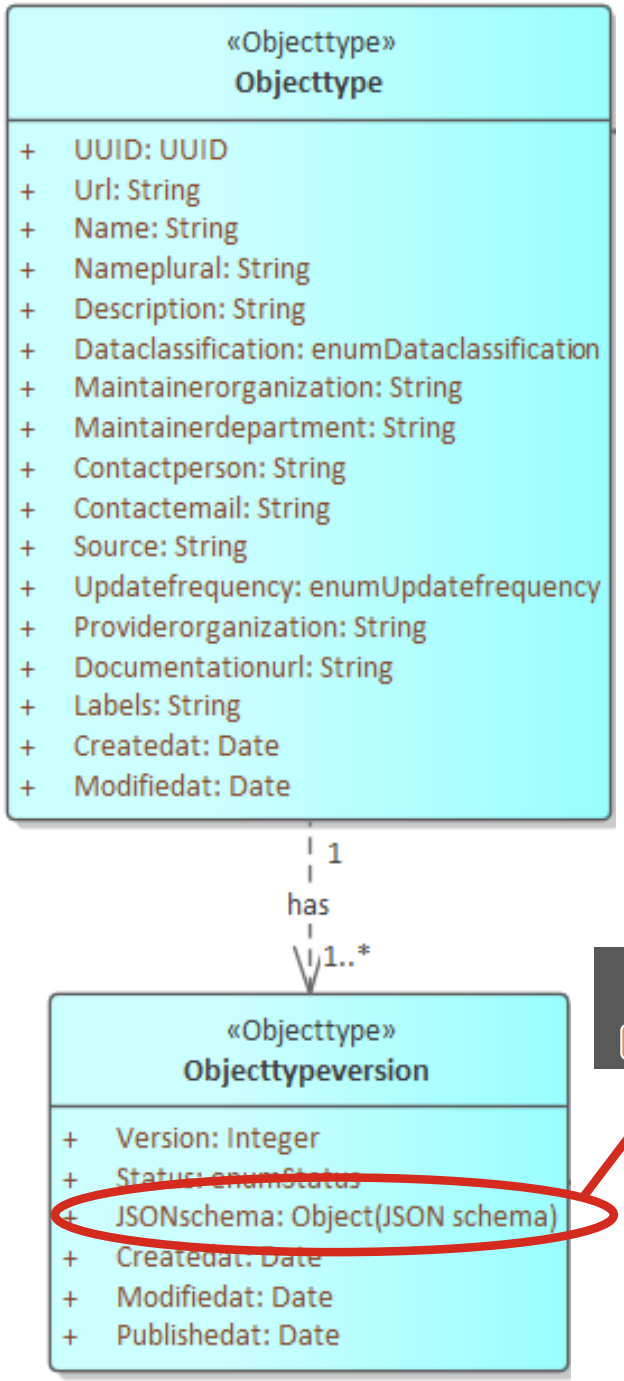
IM: Objecttypen



Spelregels

Sectorgegevens

IM: Objecttypen



Spelregels

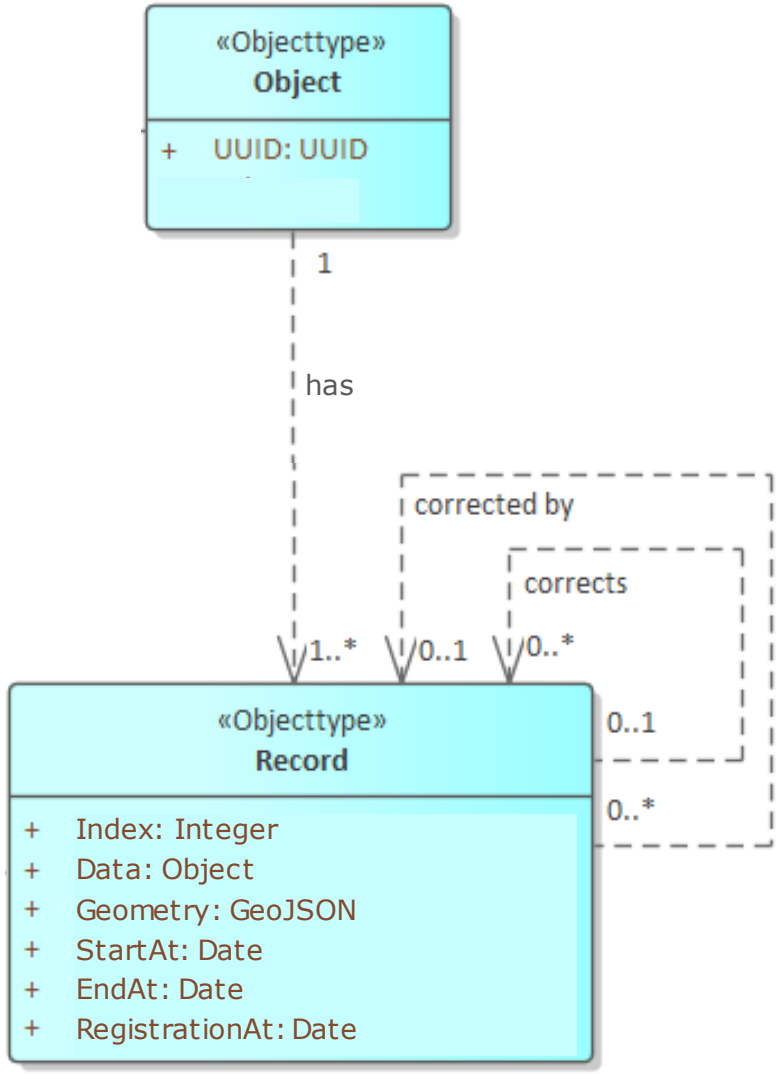
Sectorgegevens

```
{
  "title": "Boom",
  "type": "object",
  "$schema": "http://json-schema.org/draft-07/schema#",
  "required": [
    "diameter"
  ],
  "properties": {
    "diameter": {
      "type": "integer",
      "description": "Diameter op 0,3m hoogte (cm)."
    },
    "datumGepland": {
      "type": "string",
      "format": "date",
      "description": "Datum waarop de boom is geplant."
    },
    ...
  }
},
```

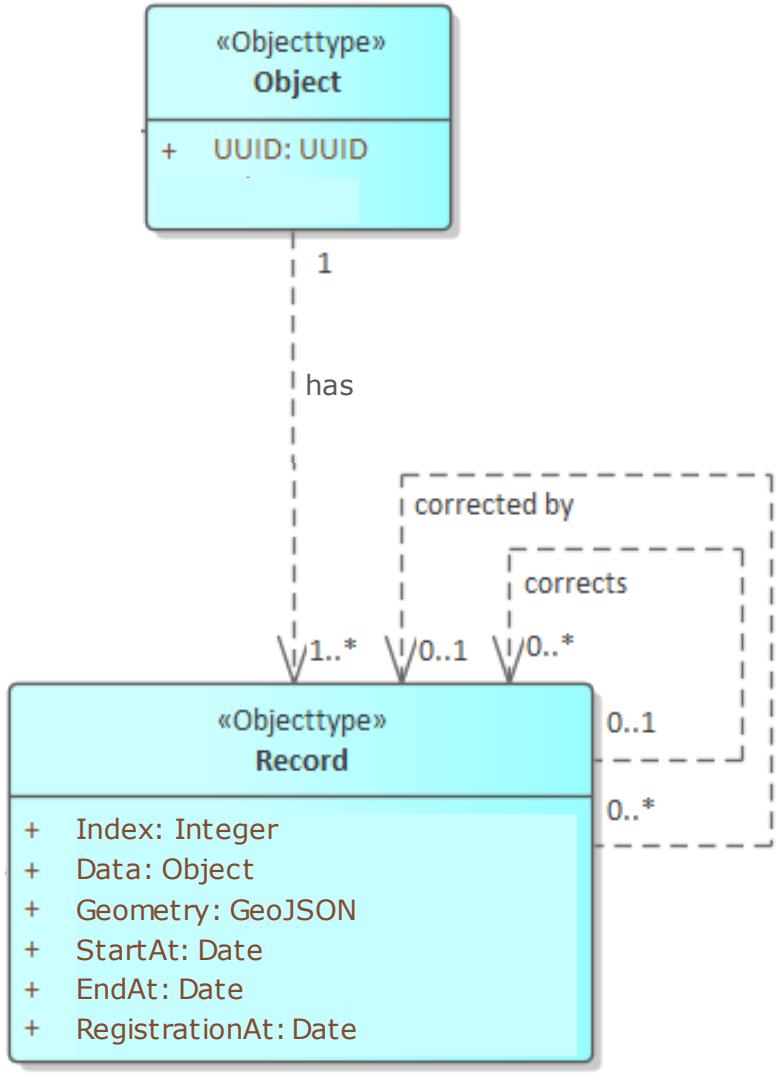
Objecten API features

- Lezen, aanmaken, bewerken en verwijderen
- Validatie tegen het Objecttype
- Materiele en formele historie standaard voor elk object
  - Correcties zijn ook mogelijk
- Zoeken en filteren
  - Op Objecttype en elk willekeurig data-attribuut
  - Op geo-coordinaten of polygonen
- Autorisatie middels API token
  - lees- en/of schrijfrechten op specifieke objecttypen.

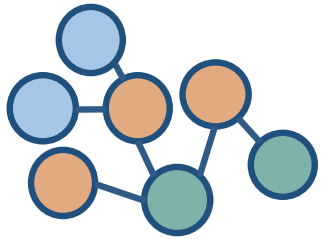
IM: Objecten


«Objecttype»
Object
+ UUID: UUID
1
has
corrected by
corrects
1..*
0..1
0..*
«Objecttype»
Record
0..1
0..*
+ Index: Integer
+ Data: Object
+ Geometry: GeoJSON
+ StartAt: Date
+ EndAt: Date
+ RegistrationAt: Date

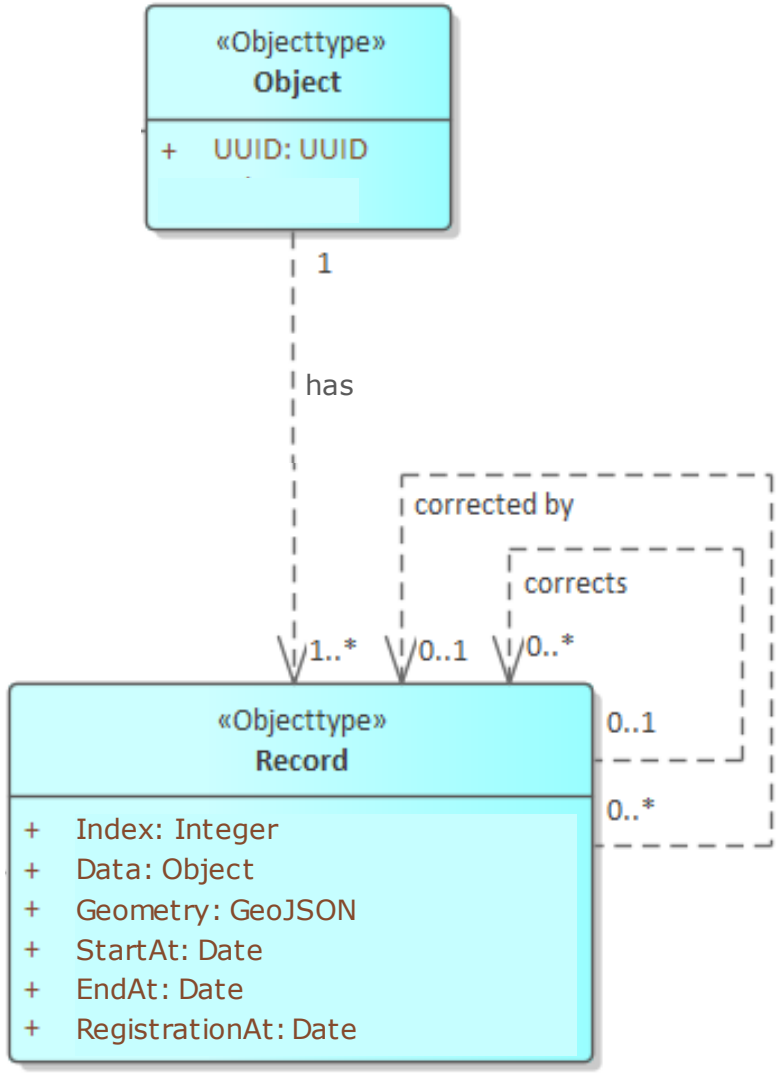
IM: Objecten



Gegevens

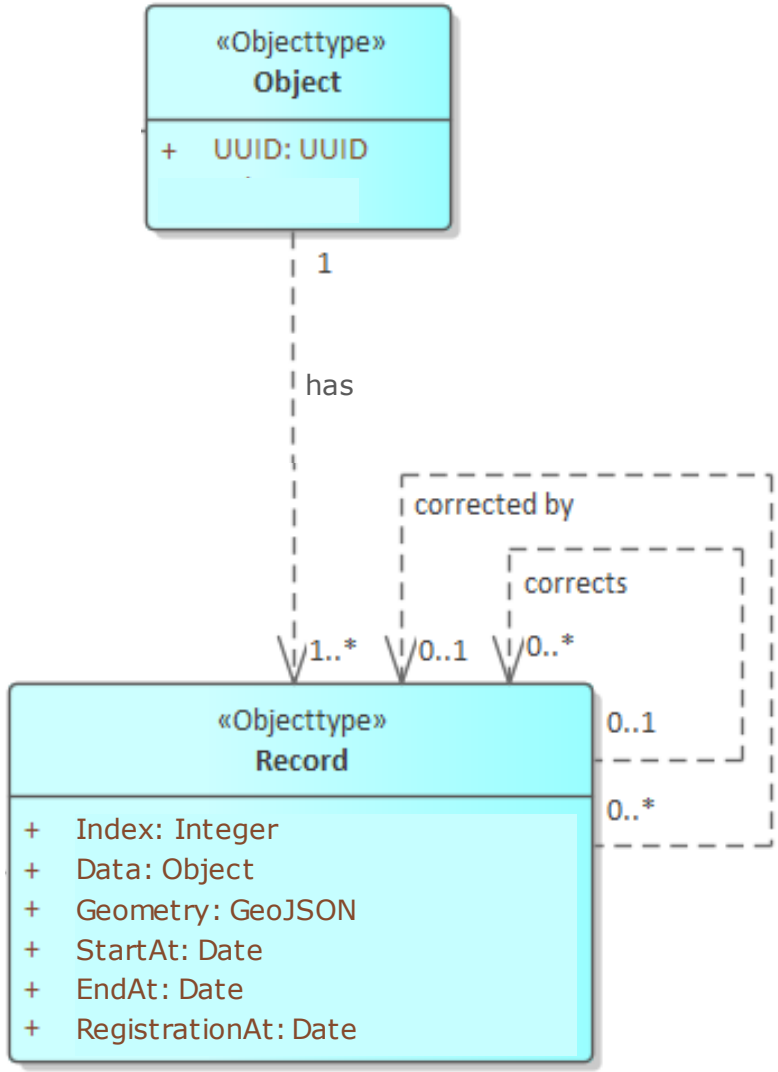


IM: Objecten



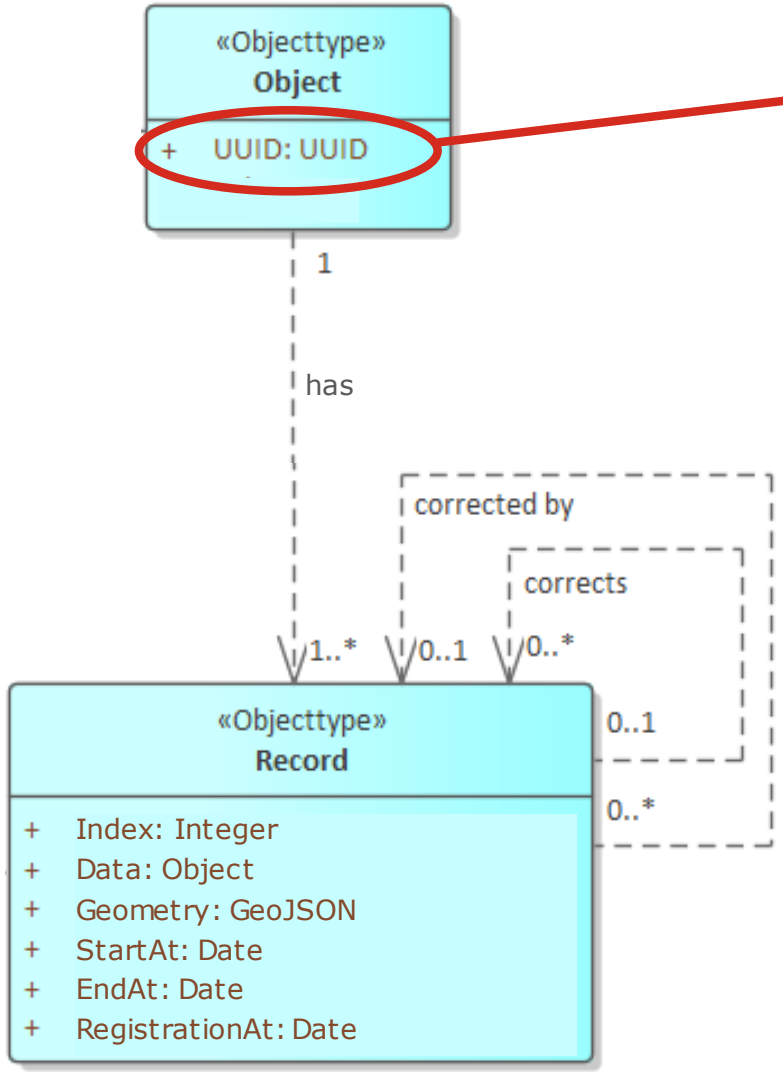

```
{
    "url": "http://example/api/v1/objects/47454f",
    "type": "http://example/api/v1/objecttypes/a63f81",
    "record": {
        "index": 6,
        "typeVersion": 1,
        "data": {
            "diameter": 26,
            "datumGeplant": "2000-11-01"
        },
        "geometry": {
            "type": "Point",
            "coordinates": [
                4.896787,
                52.37359492959213
            ]
        },
        "startAt": "2016-10-24",
        "endAt": null,
        "registeredAt": "2016-10-29",
        "correctionFor": 5,
        "correctedBy": null
    }
}
```

IM: Objecten



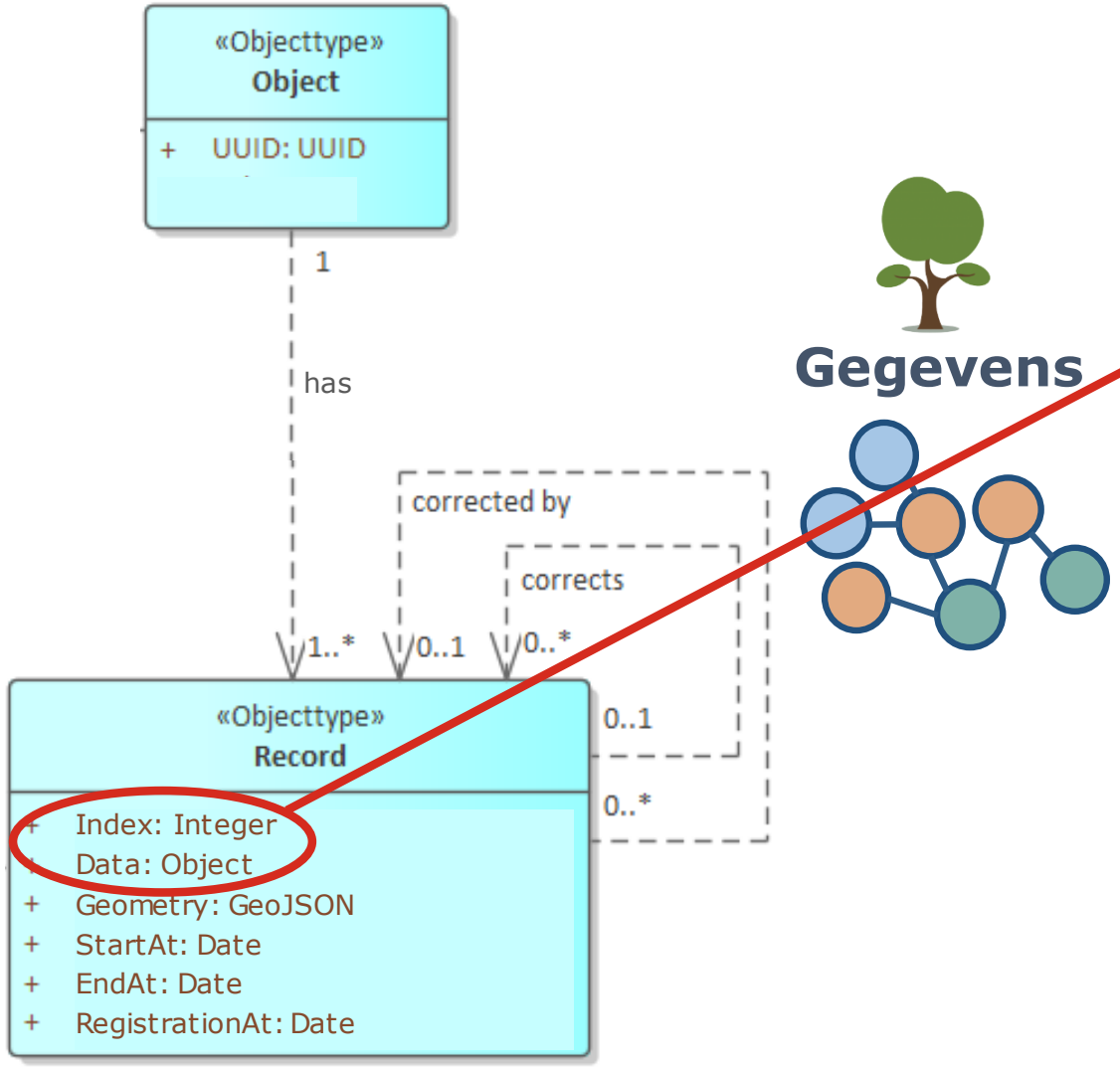

```
{
    "url": "http://example/api/v1/objects/47454f",
    "type": "http://example/api/v1/objecttypes/a63f81",
    "record": {
        "index": 6,
        "typeVersion": 1,
        "data": {
            "diameter": 26,
            "datumGeplant": "2000-11-01"
        },
        "geometry": {
            "type": "Point",
            "coordinates": [
                4.896787,
                52.37359492959213
            ]
        },
        "startAt": "2016-10-24",
        "endAt": null,
        "registeredAt": "2016-10-29",
        "correctionFor": 5,
        "correctedBy": null
    }
}
```

IM: Objecten



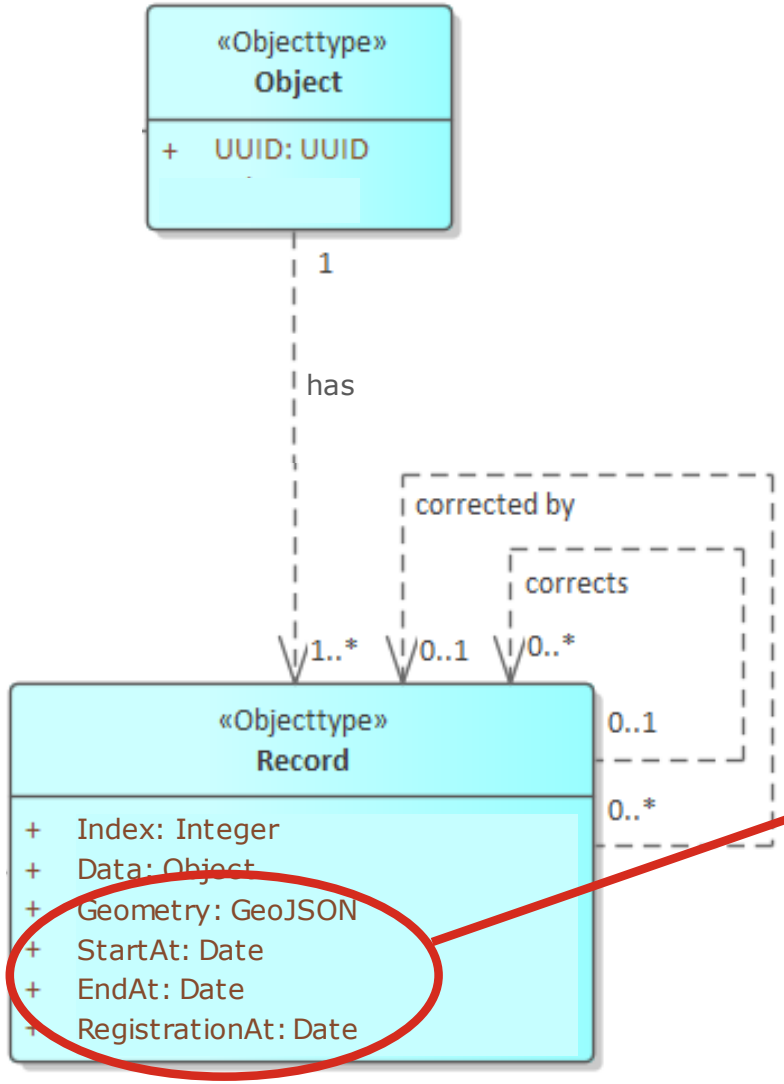

```
{
"url": "http://example/api/v1/objects/47454f",
"type": "http://example/api/v1/objecttypes/a63f81",
"record": {
    "index": 6,
    "typeVersion": 1,
    "data": {
        "diameter": 26,
        "datumGeplant": "2000-11-01"
    },
    "geometry": {
        "type": "Point",
        "coordinates": [
            4.896787,
            52.37359492959213
        ]
    },
    "startAt": "2016-10-24",
    "endAt": null,
    "registeredAt": "2016-10-29",
    "correctionFor": 5,
    "correctedBy": null
}
}
```

IM: Objecten




```
{
    "url": "http://example/api/v1/objects/47454f",
    "type": "http://example/api/v1/objecttypes/a63f81",
    "record": {
        "index": 6,
        "typeVersion": 1,
        "data": {
            "diameter": 26,
            "datumGeplant": "2000-11-01"
        },
        "geometry": {
            "type": "Point",
            "coordinates": [
                4.896787,
                52.37359492959213
            ]
        },
        "startAt": "2016-10-24",
        "endAt": null,
        "registeredAt": "2016-10-29",
        "correctionFor": 5,
        "correctedBy": null
    }
}
```

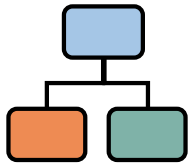
IM: Objecten



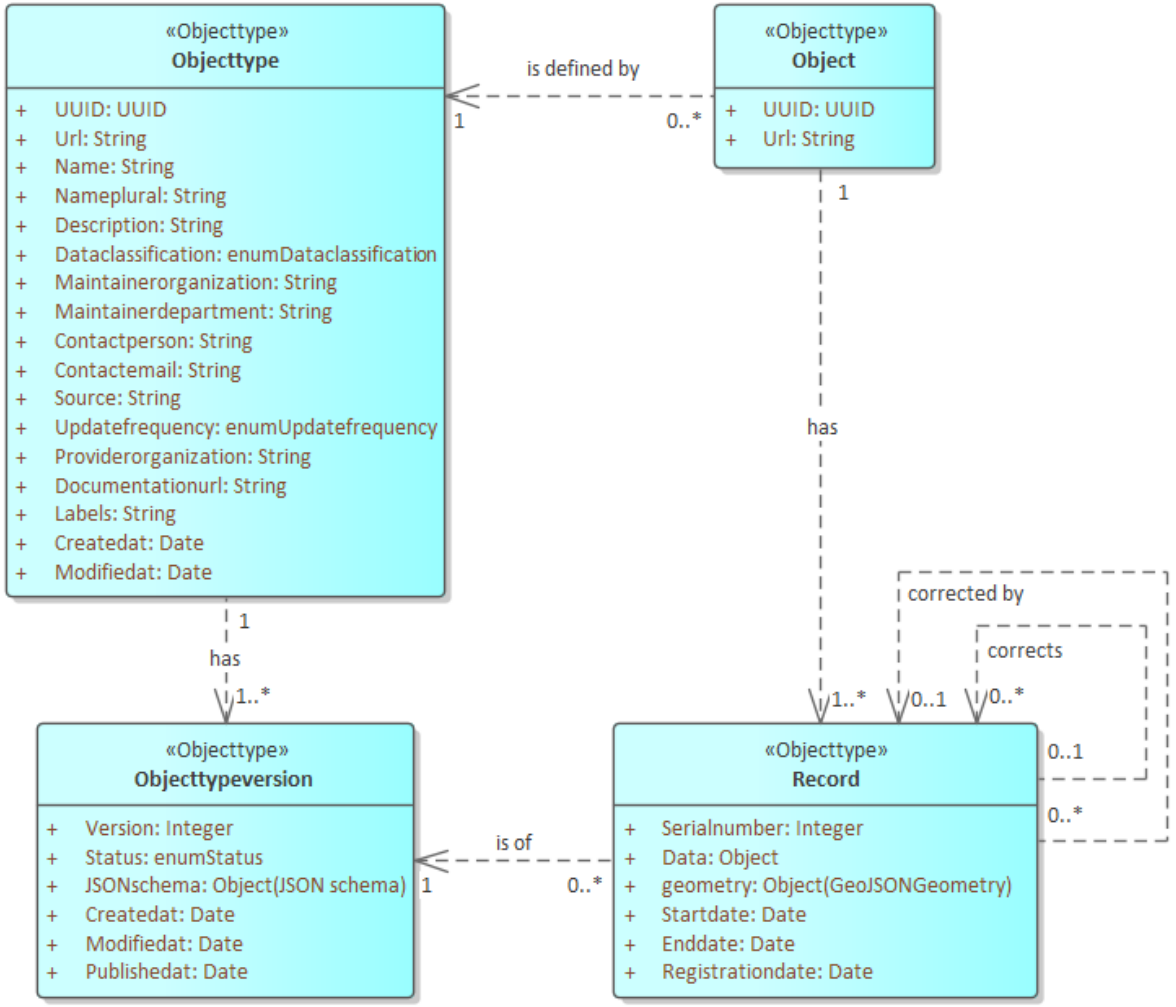

```
{
    "url": "http://example/api/v1/objects/47454f",
    "type": "http://example/api/v1/objecttypes/a63f81",
    "record": {
        "index": 6,
        "typeVersion": 1,
        "data": {
            "diameter": 26,
            "datumGeplant": "2000-11-01"
        },
        "geometry": {
            "type": "Point",
            "coordinates": [
                4.896787,
                52.37359492959213
            ]
        },
        "startAt": "2016-10-24",
        "endAt": null,
        "registeredAt": "2016-10-29",
        "correctionFor": 5,
        "correctedBy": null
    }
}
```

Volledig informatiemodel

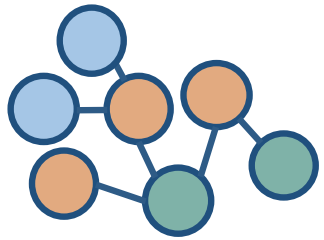
Spelregels



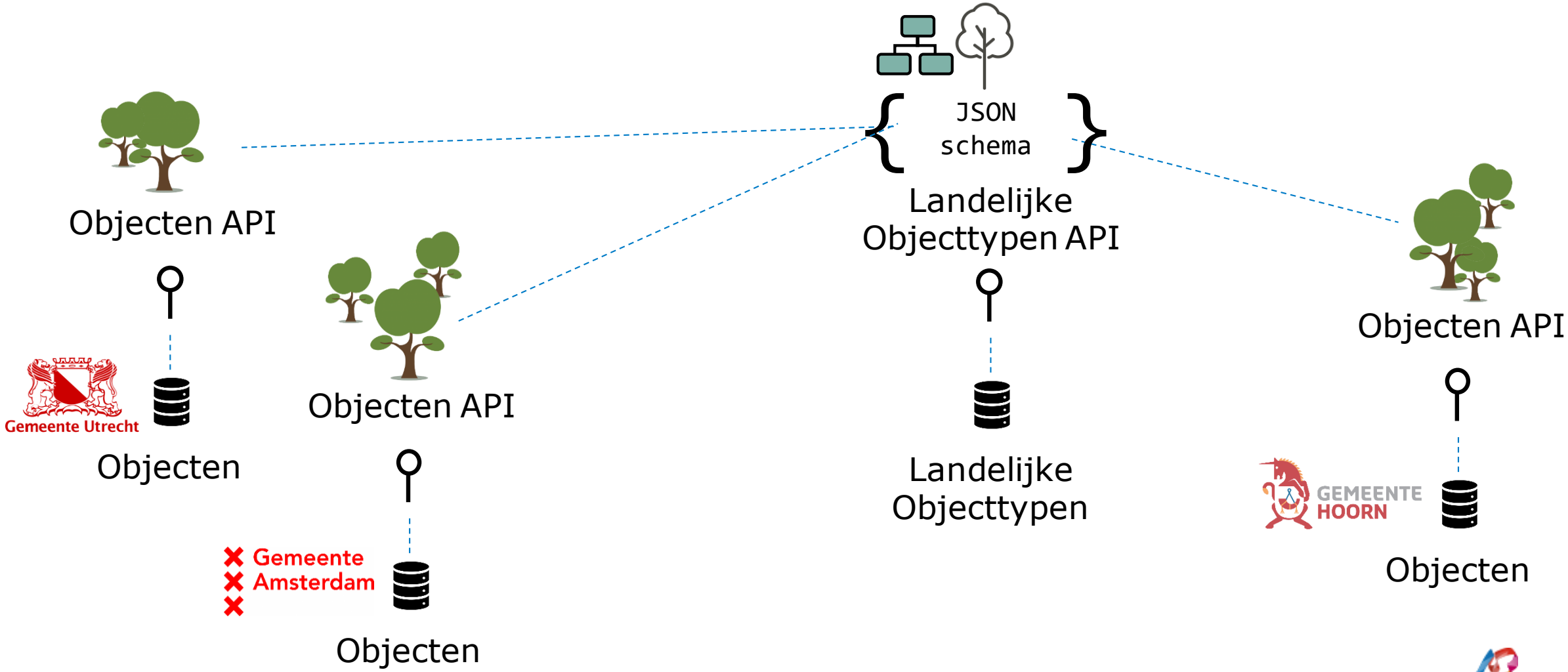
Sectorgegevens



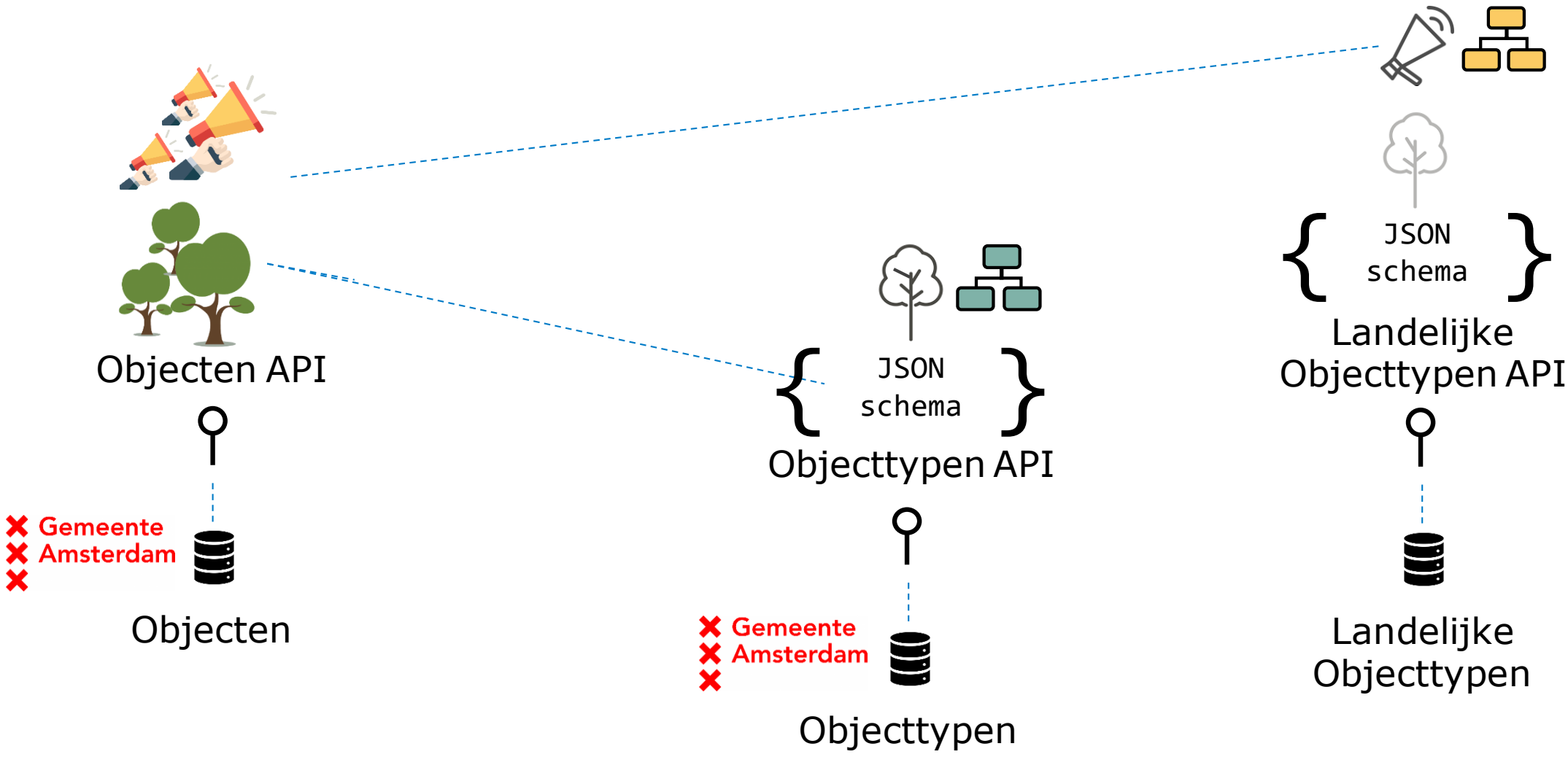
Gegevens



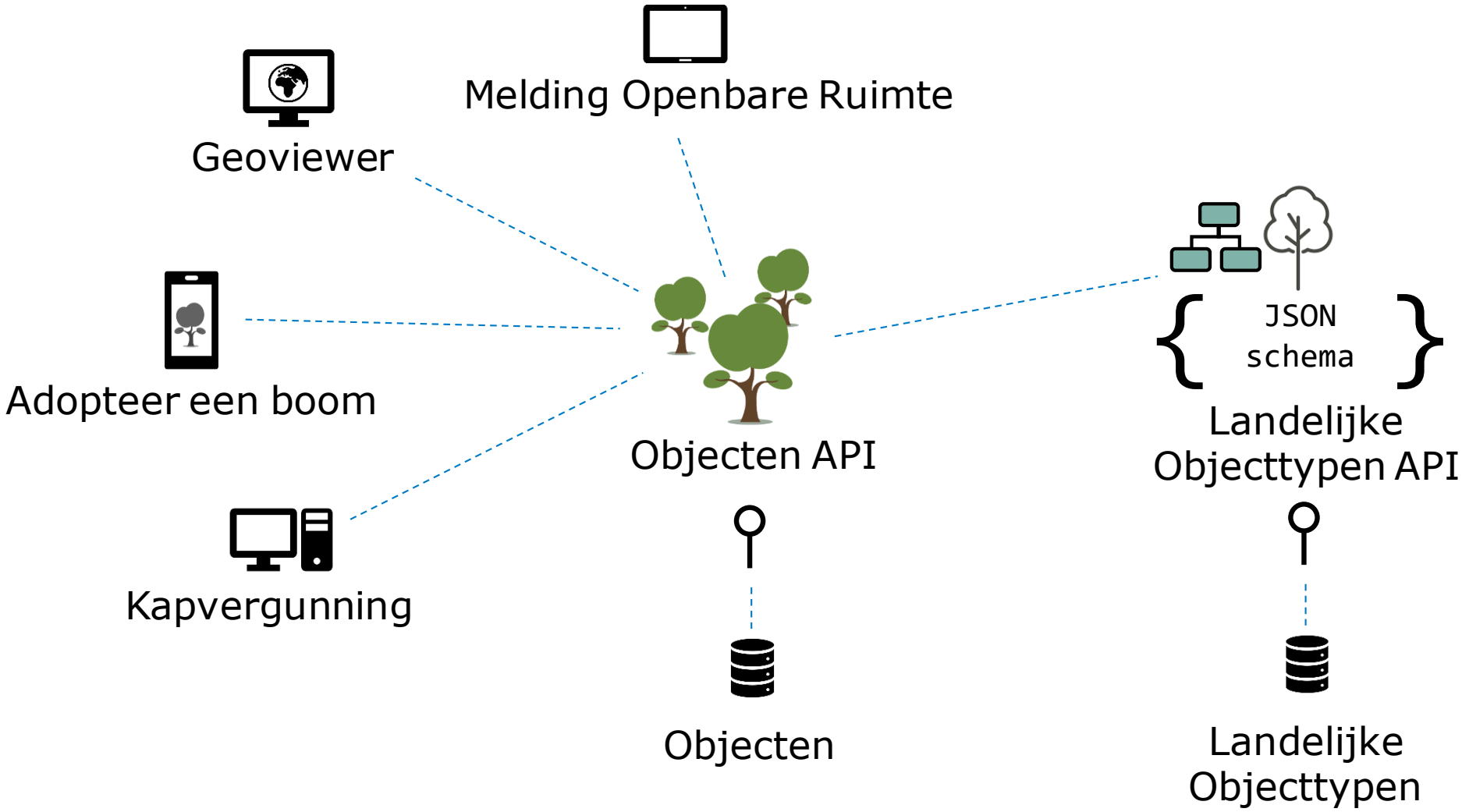
Hoe werkt het?



Hoe werkt het?



Hoe werkt het?



Waar staan we?

- In januari 2021 is versie 1.0 van de API specificaties gereleased
  - En tevens direct gebruiksklare componenten.

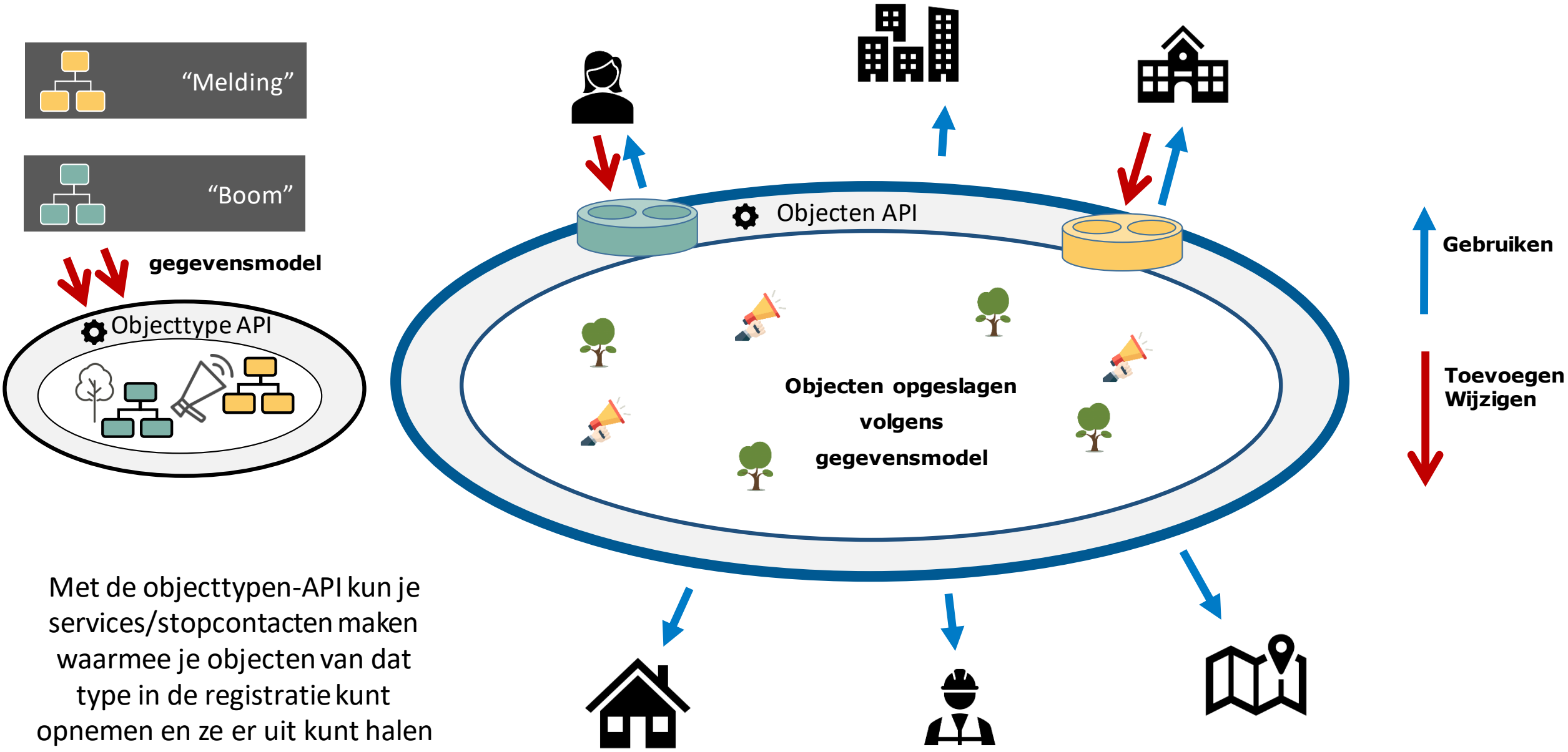
- 3 verschillende leveranciers zijn bij 3 verschillende gemeentes aan het “proeftuinen” met 4 verschillende Objecttypen:
  - Amsterdam: Erfpacht
  - Hoorn: Parkeervakken
  - Utrecht: Laadpalen en meldingen

Waar staan we?

- Ondertussen…
  - Standaardisatie van de Objecten en Objecttypen API’s bij VNG
  - Standaardisatieproces van JSON schema’s optuigen van diverse Objecttypen

- Doorontwikkeling
  - Op basis van business-, gebruikers- en ontwikkelaarsvragen

“Objecten als stroom uit het stopcontact”



Met de objecttypen-API kun je services/stopcontacten maken waarmee je objecten van dat type in de registratie kunt opnemen en ze er uit kunt halen

Zelf aan de slag gaan?

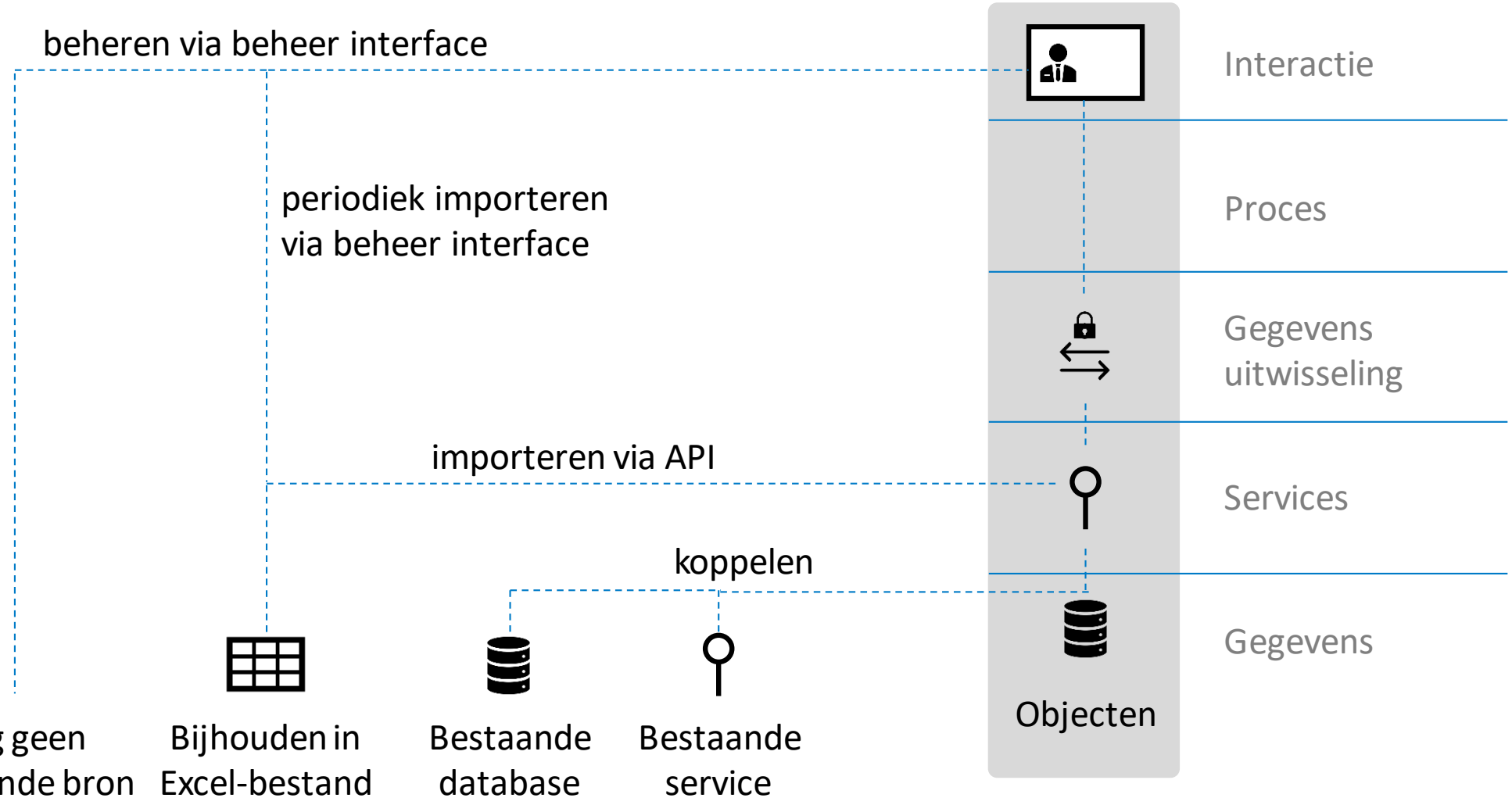
https://github.com/maykinmedia/**objects-api**

https://hub.docker.com/r/maykinmedia/**objects-api**

https://**objects-and-objecttypes-api**.readthedocs.io/

https://commonground.nl/groups/view/54477963/**objecten-en-objecttypen-api**

Discussie: Object registraties



Vraag

- Ga naar www.menti.com
- Voer de code in: 7275970

We zijn benieuwd naar jouw input/mening!

- **Welke domeingegevens ontsluit jij via de Objecten API?**
- **Welke functionaliteiten leveren nog meer waarde?**
- **Wanneer ga jij aan de slag?**

Vraag

- Ga naar www.menti.com
- Voer de code in: 7275970

We zijn benieuwd naar jouw input/mening!

- **Welke domeingegevens ontsluit jij via Objecten API?**
- **Welke functionaliteiten leveren nog meer waarde?**
- **Wanneer ga jij aan de slag?**

Gesprek

- **Administratieve wereld en geo wereld bij elkaar**

- Jij ☺
- En jij ☺
- Bart-Jan de Leuw
- Lazo Bozarov
- Joeri Bekker

In gesprek over jullie vragen en voorbeelden ☺

Gesprek

- Hoe bereiken we landelijke uitwisselbaarheid en gebruik in samenhang van objecten die we verschillend kunnen definiëren?
- Als je a priori een definitie “oplegt” levert dat weerstand op.. Nederlanders hebben hun eigen wijze ☺
- Als er nog geen landelijke standaard is, dan kunnen overheden nu met deze API een eigen definitie maken en deze publiceren.
- Wanneer er voldoende behoefte aan ontstaat, komen we vervolgens best tot een landelijke standaard. Ook, of juist, als er al ervaringen bestaan met lokaal ontstane standaarden.

Gesprek

Wat meer technische vragen:

- Waarom: StartAt in plaats van GeldigOp?
- Waarom: Record gebruiken?
- Waarom: data opnemen in een record met Geo data? In hoeverre is dat wenselijk? (kijk ook internationaal)

En hoe willen we:

- … historiemodel afstemmen op landelijke ontwikkelingen
- … implementaties in zekere mate techniek-onafhankelijk maken (spelregels niet alleen in een JSON schema) ..

Gesprek

- Waarom: StartAt in plaats van GeldigOp?
- Foundation for public code adviseert Engelse taal, NL API strategie adviseert Nederlands … welke adviezen volgen we?
- Er is gekeken naar VNG conventies met betrekking tot StUF
- Voorstel van de bouwers is om te standaardiseren op REST API, tot er in de markt aanleiding ontstaat een ander keuze te maken
- Gesprek wordt vervolgd

Gesprek

Wat is het bereik van een objectdefinitie?

- Stembureau heeft adres en heeft leden.. Hoe definiëren we het stembureau in een JSON schema ofwel spelregel?
- De *leden van het stembureau* kunnen voorkomen in de BRP en dus kun je het ID daarvan opnemen in je stembureau-definitie
- Van het *adres* van het stembureau kun je het BAG ID opnemen

Ander domein: GBI inkomen, vordering, schuldhulpverzoek, etc.

Informatie-architecten hebben hier een rol. Zij kunnen een translatie van gegevensmodel naar een JSON schema laten plaatsvinden (dit is vaak geautomatiseerd mogelijk).

Gesprek

- Landelijk standaardiseren of lokaal initiëren?
  - Welke argumenten zijn er **voor** en **tegen** om lokaal vast te beginnen als er nog geen landelijke afspraak is of komt?
  - Wat zou een checklist kunnen zijn om lokaal te hanteren om te besluiten of je een gegevensverzameling gaat ontsluiten met een “stopcontact”?
  - Wat kun je doen om latere landelijke standaardisering niet moeilijker te maken dan nodig?
- Versiebeheer op de API is een heel belangrijk middel om hiervoor in te zetten.

Gesprek

- Henri: Dit ziet er mooi uit!
- Ook bij de STUF standaarden hebben we de behoefte gezien om gegevens later toe te voegen aan definities (ExtraElementen constructie). Dat kan nu technisch alweer makkelijker gelukkig.
- Object-types lijken erg op zaak-types… daar hebben we ook geprobeerd deze dynamisch te standaardiseren, maar zonder veel succes. Waarom kan de objecten API wel slagen?
- Objecten kun je vastpakken. Dat maakt bereiken van consensus makkelijker. Bij zaaktypes is een grotere afhankelijkheid van processen en werkwijzes (en ze zijn meer virtueel van aard)

Gesprek

- Gert Jan van der Kooij stelt voor om samen met VNG te kijken naar het historiemodel vanuit de StUF periode en naar de ontwikkeling van het historiemodel op dit moment (in het kader van de samenhangende objectenregistratie)
- Lazo wil het bij de basis houden: eenvoud maakt/houdt het bruikbaar

Gesprek

- Replicatie en translatie naar andere standaarden is mogelijk
- Omzetten naar REST is mogelijk: eerst een PoC en dan met stakeholders samen implementeren als het nuttig (b)lijkt

- Partijen hebben we object API zelfstandig in een dag werkend gekregen (er was enkel wat hulp nodig bij het JSON schema)

Save the date

- 30 maart
- Nieuwe DiS Online over dit onderwerp
- Dan kijken we met elkaar naar een toepassing voor gebruikers om “Best of Both Worlds” invulling te geven

Onderzoekspunten op te pakken

- Onderzoekspunt: objecten in basisregistraties hebben al identificaties, en die voldoen niet aan UUID. Hoe gaan we daar mee om?
- Onderzoekspunt: registeredAt is in NEN3610 maar 1 deel van de registratie tijdslijn, StUF kent alleen 1 tijdstip hier (vanuit proces), maar in NEN3610 zijn er dus 2 (vanuit informatieperspectief). Waarom is hier de StUF lijn gevolgd?
- Onderzoekspunt: Record valt mijns inziens onder “implementation bleed"
- Onderzoekspunt: een API mag mijns inziens niet voorschrijvend zijn voor het onderliggende historiemodel. Dat lijkt hier wel zo te zijn, vanwege records.
- Objecten en features in API’s kennen dit extra level "data" niet, alsmede is dit niet gewenst omdat het niet goed aansluit bij OGC standaarden
- startAt en endAt zijn Engelse termen, in NEN3610 heten deze anders, in NL API heten deze parameters beschikbaarOp en geldigOp. Kunnen we “alignen”?

Website

www.geobasisregistraties.nl

E-mail

DiSGeo@minbzk.nl

Contactpersonen

Lazo Bozarov, 14 030

Joeri Bekker, 020 753 0523

Bart-Jan de Leuw, 06 513 631 88