

Techniek sprint 5

#344: Insert catch phrase

Prio: Laag



Inhoud

- Opbouw integratietests
- Overzicht userstories
- Ontwerpkeuzes
- Autorisatie
- Historie

Opbouw integratietests

Hoe implementatie business logica valideren?



Opbouw integratietests - infrastructuur

We willen bewijzen dat onze referentie-implementaties kunnen samenwerken (E2E)

- Docker container images (virtualisatie)
 - ZRC, DRC, ZTC, BRC, ORC
 - Plug-and-play → 1 component makkelijk met andere implementatie vervangen
- API calls gaan over echt (docker) netwerk
- Echte, persistente databases (testdata inladen!)

```
Test session starts (platform: linux, Python 3.6.6, pytest 3.6.1, pytest-sugar 0.9.1)
rootdir: /testenv, inifile:
plugins: sugar-0.9.1

tests/test_registreer_besluiten.py XXXXXXXX 44%
tests/test_registreer_zaakdocument.py ✓✓✓✓ 69%
tests/test_userstory_171.py ✓ 75%
tests/test_userstory_39.py ✓ 81%
tests/test_userstory_42.py ✓ 88%
tests/test_userstory_45.py ✓ 94%
tests/test_userstory_52.py ✓ 100%
----- generated xml file: /testenv/reports/junit.xml -----

Results (15.83s):
9 passed
2 xpassed
1 xfailed
```

Opbouw integratietests - tests

Sequentieel realistische flow nabootsen

- success-story
- error-cases

```
-->
13 @pytest.mark.incremental
14 class TestZaakInformatieObjecten:
15
16     def test_creeer_zaak_en_informatieobject(self, state, zrc_client, drc_client, ztc_client, text_file): ...
50
51     def test_relateer_zaak_en_informatieobject(self, state, zrc_client, drc_client): ...
76
77     def test_relateer_informatieobject_dubbel_zrc(self, state, zrc_client): ...
92
93     def test_relatie_eerst_in_drc_dan_zrc(self, state, zrc_client, drc_client, text_file): ...
```

Opbouw integratietests

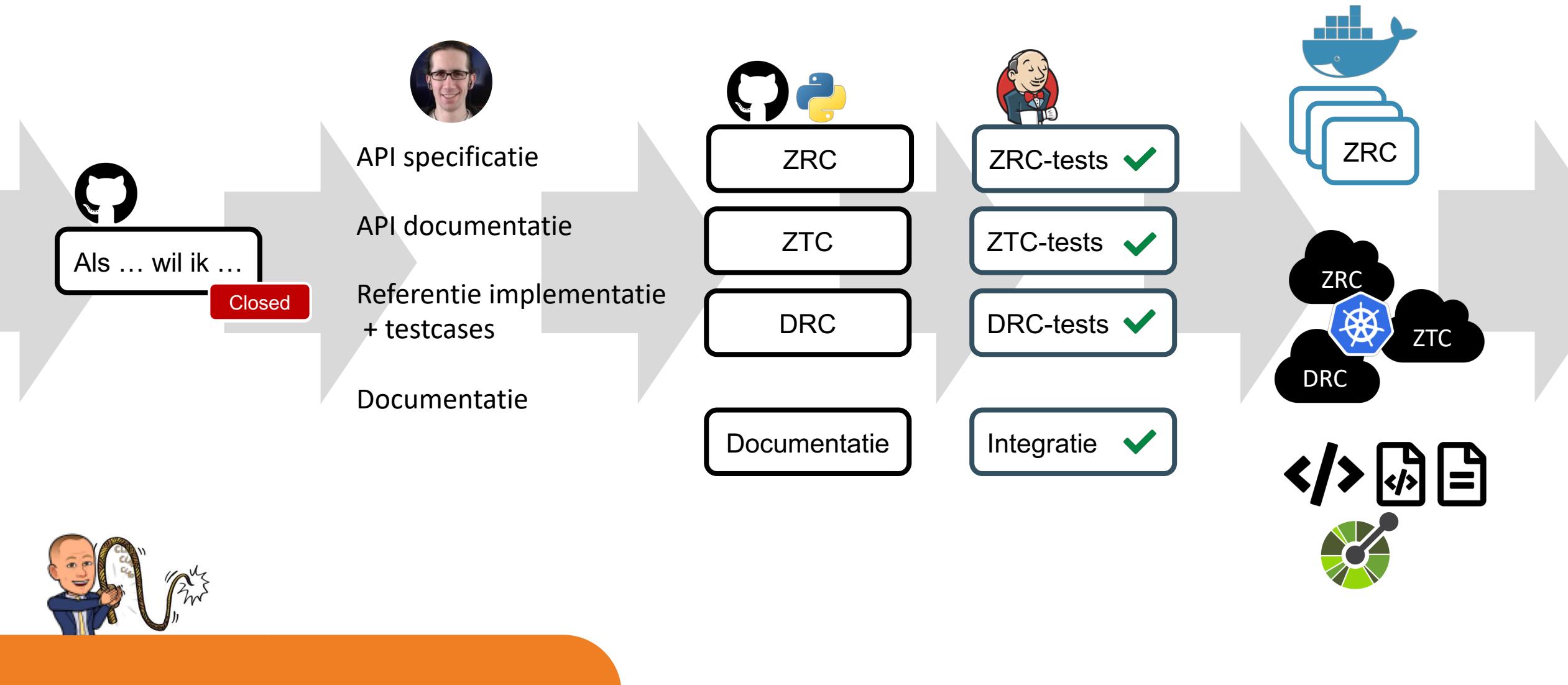
- input
- roep aan wat je wil testen
- verifieer het resultaat

```
13 @pytest.mark.incremental
14 class TestZaakInformatieObjecten:
15
16     def test_creeer_zaak_en_informatieobject(self, state, zrc_client, drc_client):
17
18         def test_relateer_zaak_en_informatieobject(self, state, zrc_client, drc_client):
19
20             def test_relateer_informatieobject_dubbel_zrc(self, state, zrc_client, drc_client):
21
22                 def test_relatie_eerst_in_drc_dan_zrc(self, state, zrc_client, drc_client):
23                     """
24                         Test dat de relatie zaak-informatieobject moet bestaan in het DRC voordat je de symmetrische relatie in het ZRC mag leggen.
25                     """
26
27                     document2 = drc_client.create('enkelvoudiginformatieobject', {
28                         'creatiedatum': '2018-09-12',
29                         'titel': 'zaak_samenvatting.txt',
30                         'auteur': 'Jos den Homeros',
31                         'taal': 'dut',
32                         'informatieobjecttype': state.informatieobjecttype['url'],
33                         'inhoud': encode_file(text_file),
34                     })
35
36                     zaak_uuid = get_uuid(state.zaak)
37
38                     with pytest.raises(ClientError) as exc_context:
39                         zrc_client.create('zaakinformatieobject', {
40                             'informatieobject': document2['url'],
41                         }, zaak_uuid=zaak_uuid)
42
43                         assert exc_context.value.args[0]['status'] == 400
```

Overzicht userstories

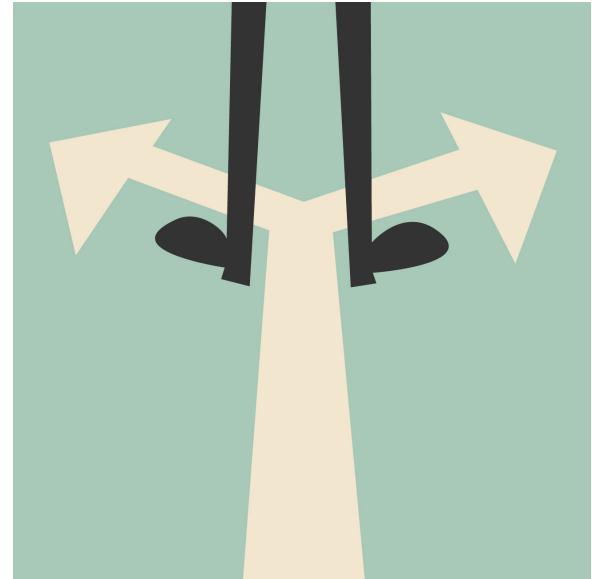
Kijkje op het scrumbord

Van User Story naar ...



Ontwerpkeuzes

Gemaakt in sprint 5



Created by Dooder - Freepik.com

Totstandkoming

- Praktische problemen...
 - Bij interpretatie van de informatiemodellen
 - Bij realisatie van de referentie implementatie
 - Bij opzetten (integratie-) tests op basis van User Stories
 - ...

joeribekker commented on 17 Sep

Member + ...

Besloten tijdens overleg d.d. 17 september 2018

➤..leiden tot ontwerp keuzes

 joeribekker added **Verhef tot ontwerp keuze**

27 Done - Sprint 5 + ...

4 results

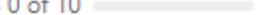
 Zaak afsluiten design keuze toegevoegd ...

✓ #351 opened by joeribekker

Prio H Verhef tot ontwerp k... 

 Changes approved

 Als ontwikkelaar wil ik duidelijkheid over wanneer een API resource "genest" moet worden...

0 of 10 

#230 opened by joeribekker

DSO Prio H Verhef tot ontwerp k... 

 Als ontwikkelaar wil ik dat URLs zowel met als zonder "/" werken voor collections (list)

#419 opened by joeribekker

Prio H Verhef tot ontwerp k... 

 Design keuze omtrent query parameters toevoegen voor relaties en/of groepsattributen

#422 opened by joeribekker

Prio H Verhef tot ontwerp k... 

Zaak afsluiten: Hoe?

- Oorsprong
 - Blijkbaar niet ondubbelzinnig sinds het ontstaan van RGBZ
 - Issue voor het eerst (zichtbaar) op de radar in 2016
 - Laatste vraag hierover begin dit jaar
- Case in het kort: Hoe wordt een Zaak afgesloten?
 - Middels een `actualiseerZaakstatus`, waarbij een `eindstatus` wordt gezet;
 - Middels `updateZaak` waarbij een `einddatum` wordt gezet;
 - Of beide?

Zaak afsluiten: Ontwerp keuze

Een Zaak wordt afgesloten door een eindstatus* toe te kennen

* een eindstatus is de status binnen het Zaaktype met het hoogste volgnummer

4. Consumer haalt de ZAAK opnieuw op:

```
GET /zrc/api/v1/zaken/12345

HTTP 200
{
    "einddatum": "2018-10-08",
    // ...
}

,1
"uuid": 67890,
"volgnummer": 2, # Het laatste STATUSTYPE binnen dit ZAAKTYPE
"isEindstatus": true,
// ...
```

Afwijking op DSO: Nesten van resources

API-09

Relaties van geneste resources worden binnen het eindpunt gecreëerd

Als een relatie alleen kan bestaan binnen een andere resource (geneste resource), wordt de relatie binnen het eindpunt gecreëerd. De afhankelijke resource heeft geen eigen eindpunt.

Voor bijvoorbeeld het ZTC betekent dit:

/api/v1/catalogus/12345/zaaktypes/67890/statustypes

En dus niet:

/api/v1/statustypes -of-

/api/v1/catalogus/12345/statustypes

Afwijking op DSO: Nesten van resources

Toon een lijst van zaken met hun status...

```
zaken = client.get('/zrc/api/v1/zaken?expand=status')
for zaak in zaken:
    # 1 request per zaak... of best case per zaaktype
    statustype = client.get(zaak['status']['statustype'])
    print(zaak['identificatie'], statustype['naam'])
```

```
zaken = client.get('/zrc/api/v1/zaken?expand=status')
statustypen = dict([entry['url'] for entry in client.get('/ztc/api/v1/statustypen')])
for zaak in zaken:
    # geen request per zaak nodig, alles is al opgehaald
    statustype = statustypen[zaak['status']['statustype']]
    print(zaak.identificatie, statustype.naam)
```

Overige design keuzes

- URLs voor het opvragen van een resource eindigen nooit met een slas

~~http://example.com/api/v1/zaken/~~ vs http://example.com/api/v1/zaken

- Query parameters voor bijv. filteren van geneste objecten middels “__”

/api/v1/zaken?kenmerken__bron=foobar

```
{  
  "url": "https://zrc/api/v1/zaken/12345",  
  "identificatie": 12345,  
  // ...  
  "kenmerken": [  
    {"kenmerk": "test",  
     "bron": "foobar"  
  ]  
}
```

Autorisatie

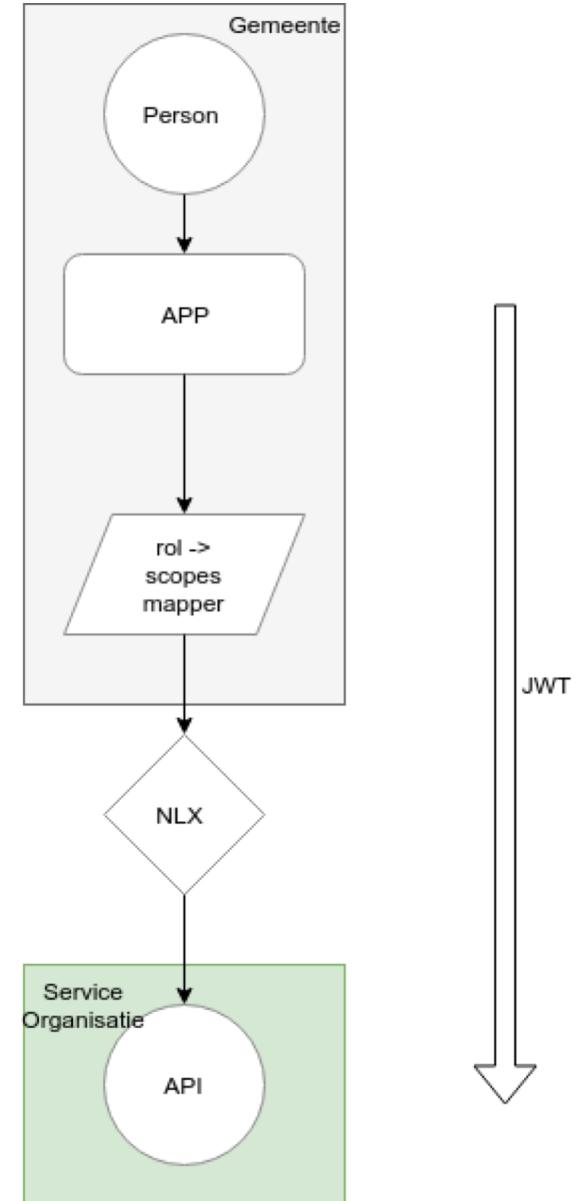
Verkenning



Autorisatie - probleemstelling

- Elke organisatie kent eigen ‘rollen’
- Medewerker heeft beperkte rechten
- ZRC, DRC, ZTC... zijn generiek

- Hoe limiteren wat een eindgebruiker kan zien/doen?
- Hoe voorkomen we beheerslast?
- Hoe laten we toe dat elke organisatie zelf interne autorisaties regelt?
- Context bij API call blijkt belangrijk bij limiteren data in response



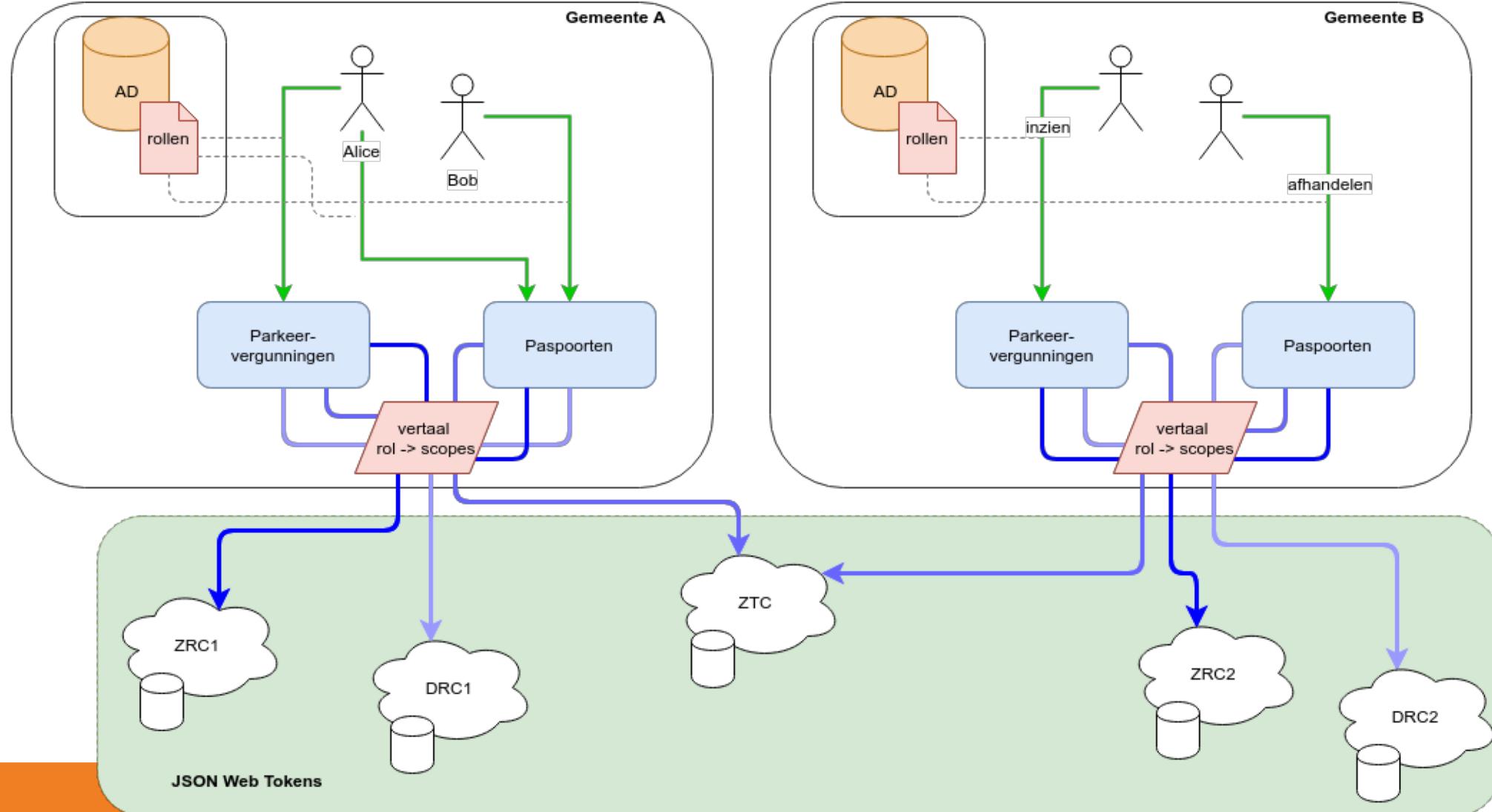
Autorisatie - oplossingsrichting

- API kent scopes (= sets van permissies, vb. 'zaak.lezen')
- Scopes kunnen mappen op generieke rollen/vertrouwelijkheidsaanduiding
- Taakapplicatie stuurt JWT naar API met scopes
- Organisatie maapt interne rollen naar scopes
- NLX logt requests met JWT header → gebruikte scopes kunnen ge-audit worden

- Authenticatie: taak gemeente, in taakapplicatie → leidt tot JWT
- Autorisatie: enkel taakapplicaties kunnen valide JWT maken met nodige scopes

- Context bij API call blijkt belangrijk bij limiteren data in response

Autorisatie - oplossingsrichting



Historie

Verkenning



Historie

- Formele & materiele historie
- Materieel: hoe was werkelijke situatie op moment X
- Formeel: hoe was situatie vastgelegd op moment Y

- Vragen die moeten gesteld kunnen worden:
 - Was wijziging X bekend op moment Y?
 - Op welke ‘versie’ van een document slaat een verwijzing?
 - Geef de versie zoals op moment Y bekend stond
 - Geef de versie zoals de werkelijkheid op moment X was

- Wat kunnen we recyclen van StUF?

Van User Story naar ...

