

Zaakgericht Werken API

Logius Handreiking

Werkversie 23 september 2025

**Laatste werkversie:**

<https://logius-standaarden.github.io/zaakgericht-werken-api/>

Redacteur:

Logius Standaarden ([Logius](#))

Auteur:

Logius Standaarden ([Logius](#))

Doe mee:

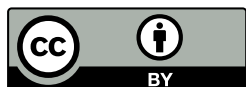
[GitHub logius-standaarden/zaakgericht-werken-api](#)

[Dien een melding in](#)

[Revisiehistorie](#)

[Pull requests](#)

Dit document is ook beschikbaar in dit niet-normatieve formaat: [pdf](#)



Dit document valt onder de volgende licentie:

[Creative Commons Attribution 4.0 International Public License](#)

Samenvatting

De standaard "API's voor Zaakgericht Werken" bestaat uit een aantal API's. Per API is er een OAS3-specificatie en een beschrijving van het vereiste "run-time"-gedrag in zoverre dat niet kon worden vastgelegd in de API-specificatie. De OAS3-specificaties met beschrijvingen zijn normatief. De overige documentatie is ondersteunend en ter informatie.

Deze standaardisatie zorgt vervolgens voor gegarandeerde interoperabiliteit tussen registraties (providers) en consumers die van de API's gebruik maken.

Status van dit document

Dit is een werkversie die op elk moment kan worden gewijzigd, verwijderd of vervangen door andere documenten. Het is geen door het TO goedgekeurde consultatieversie.

Inhoudsopgave

Samenvatting

Status van dit document

1. Conformiteit

2. Zaken API

2.1 Runtime gedrag

2.1.1 Algemeen

2.1.2 Archiveren

A. Referenties

A.1 Normatieve referenties

§ 1. Conformiteit

Naast onderdelen die als niet normatief gemarkeerd zijn, zijn ook alle diagrammen, voorbeelden, en noten in dit document niet normatief. Verder is alles in dit document normatief.

De trefwoorden *MAG*, *MOET* en *MOETEN* in dit document moeten worden geïnterpreteerd als in [BCP 14](#) [[RFC2119](#)] [[RFC8174](#)] als, en alleen als deze in hoofdletters zijn weergegeven, zoals hier getoond.

§ 2. Zaken API

API voor het opslaan en ontsluiten van zaakgegevens.

De API ondersteunt het opslaan en het naar andere applicaties ontsluiten van gegevens over alle gemeentelijke zaken, van elk type. Opslag vindt plaats conform het RGBZ waarin objecten, gegevens daarvan en onderlinge relaties zijn beschreven. Het bevat echter niet alle gegevens uit het RGBZ: documenten worden bijv. opgeslagen in de Documenten API, besluiten in de Besluiten API, etc. Vanuit zaken worden er dan ook relaties gelegd naar andere resources.

§ 2.1 Runtime gedrag

Bepaalde gedragingen kunnen niet in een OAS spec uitgedrukt worden omdat ze businesslogica bevatten. Deze gedragingen zijn hieronder beschreven en *MOETEN* zoals beschreven geïmplementeerd worden.

§ 2.1.1 Algemeen

§ 2.1.1.1 Valideren zaaktype op de Zaak-resource ([zrc-001](#))

aangepast in versie 1.5.0

Bij het aanmaken (`zaak_create`) of bijwerken (`zaak_update`, `zaak_partial_update`) van een zaak *MOET* de URL-referentie naar het zaaktype gevalideerd worden op het bestaan. Indien het ophalen van het zaaktype niet (uiteindelijk) resulteert in een HTTP 200 status code, *MOET* het ZRC antwoorden met een HTTP 400 foutbericht.

De provider *MOET* tevens valideren dat het opgehaalde zaaktype een zaaktype is conform de vigerende Catalogi API specificatie.

§ 2.1.1.2 Garanderen uniciteit bronorganisatie en identificatie op de Zaak-resource ([zrc-002](#))

Bij het aanmaken (`zaak_create`) en bijwerken (`zaak_update` en `zaak_partial_update`) *MOET* gevalideerd worden dat de combinatie identificatie en bronorganisatie uniek is, indien de identificatie door de consumer meegestuurd wordt.

Indien de identificatie niet door de consumer gestuurd wordt, dan *MOET* het ZRC de identificatie genereren op een manier die garandeert dat de identificatie uniek is binnen de bronorganisatie.

§ 2.1.1.3 Valideren informatieobject op de ZaakInformatieObject-resource ([zrc-003](#))

Bij het aanmaken (zaakinformatieobject_create) *MOET* de URL-referentie naar het informatieobject gevalideerd worden op het bestaan. Indien het ophalen van het object niet (uiteindelijk) resulteert in een HTTP 200 status code, *MOET* het ZRC antwoorden met een HTTP 400 foutbericht.

§ 2.1.1.4 Valideren relatieinformatie op ZaakInformatieObject-resource ([zrc-004](#))

Op basis van het objectType *MOET* de aardRelatie gezet worden conform het RGBZ. Omdat het objectType zaak is, moet aardRelatie gelijk zijn aan "hoort_bij".

De registratiedatum *MOET* door het systeem gezet worden op het moment van aanmaken.

Bij het updaten (zaakinformatieobject_update en zaakinformatieobject_partial_update) is het NIET TOEGESTAAN om de relatie te wijzingen. Bij andere waarden voor de attributen zaak, en informatieobject *MOET* het ZRC antwoorden met een HTTP 400 foutbericht.

§ 2.1.1.5 Synchroniseren relaties met informatieobjecten ([zrc-005](#))

Wanneer een relatie tussen een INFORMATIEOBJECT en een ZAAK gemaakt of bijgewerkt wordt, dan *MOET* het ZRC in het DRC ook deze relatie aanmaken/bijwerken.

Een voorbeeld:

1. Een informatieobject wordt gerelateerd aan een zaak door een consumer:

```
POST https://zrc.nl/api/v1/zaakinformatieobjecten HTTP/1.0

{
  "informatieobject": "https://drc.nl/api/v1/enkelvoudigeinformatie",
  "zaak": "https://zrc.nl/api/v1/zaken/456789",
  "titel": "",
  "beschrijving": ""
}
```

2. Het ZRC *MOET* de relatie spiegelen in het DRC:

```
POST https://drc.nl/api/v1/objectinformatieobjecten HTTP/1.0

{
  "informatieobject": "https://drc.nl/api/v1/enkelvoudigeinformatie
  "object": "https://zrc.nl/api/v1/zaken/456789",
  "objectType": "zaak"
}
```

Merk op dat het aanmaken van de relatie niet gelimiteerd is tot het aanmaken via de API. Indien elders (bijvoorbeeld via een admininterface) een relatie tot stand kan komen, dan *MOET* deze ook gesynchroniseerd worden.

§ 2.1.1.6 Data filteren bij de bron op basis van zaaktypes ([zrc-006](#))

Het AC legt op het niveau van **zaaktype** vast welke operaties mogelijk zijn en wat de maximale vertrouwelijkheidsaanduiding is voor een consumer.

Het ZRC **MAG ENKEL** zaken ontsluiten waarvan:

- het zaaktype voorkomt in de autorisaties van de consumer.
- de vertrouwelijkheidsaanduiding lager of gelijk aan de maximale vertrouwelijkheidsaanduiding is voor het betreffende zaaktype.

De API-specificatie legt vast welke scopes nodig zijn voor welke operaties. Een provider *MOET* operaties blokkeren op zaken waarvan de nodige scopes niet toegekend zijn voor het zaaktype van de betreffende zaak.

Indien een operatie niet toegelaten is, dan *MOET* de provider met een HTTP 403 foutbericht antwoorden.

Een consumer is verbonden aan het concept **Applicatie**, waarop autorisaties gedefinieerd worden. Het is mogelijk om op het niveau van **Applicatie** de vlag heeftAlleAutorisaties te zetten. Indien deze gezet is, dan *MOET* de provider alle operaties voor deze consumer toelaten, op alle zaken.

§ 2.1.1.7 Afsluiten zaak ([zrc-007](#))

Een zaak wordt afgesloten door een eindstatus toe te kennen aan een Zaak. Elk `ZaakType` *MOET* minimaal één `StatusType` kennen. De eindstatus binnen een `ZaakType` is het `StatusType` met het hoogste volgnummer.

Een `Zaak` *MOET* een `Resultaat` hebben voor deze afgesloten kan worden.

De `Zaak.einddatum` *MOET* logisch afgeleid worden uit het toekennen van de eindstatus via de `Status.datumStatusGezet`.

Als een `Zaak` een eindstatus heeft dan is de zaak afgesloten en mogen gegevens van de zaak niet meer aangepast worden (behalve om redenen van correctie). Dit *MOET* beveiligd worden met de scope `zds.scopes.zaken.geforceerd-bijwerken`. "Gegevens van de zaak" omvat hier ook de gerelateerde gegevens zoals klantcontacten, resultaat, rollen, statussen, zaakinformatieobjecten, zaakobjecten, zaakbesluiten en zaakeigenschappen.

Bij het afsluiten van een `Zaak` *MOET* het ZRC het `Informatieobject.indicatieGebruiksrecht` controleren van alle gerelateerde informatieobjecten. Deze *MAG NIET* null zijn, maar *MOET* `true` of `false` zijn. Indien dit niet het geval is, dan dient het ZRC een validatiefout te genereren met code `indicatiegebruiksrecht-unset`.

§ 2.1.1.8 Heropenen zaak ([zrc-008](#))

Bij het heropenen van een `Zaak` *MOET* de client een andere status toevoegen aan de zaak dan een eindstatus. Hiervoor *MOET* de client de scope `zds.scopes.zaken.heropenen` hebben.

Tevens *MOET* de provider de volgende velden op null zetten zodra een eindstatus wordt gewijzigd in een andere status:

- `Zaak.einddatum`
- `Zaak.archiefactiedatum`
- `Zaak.archiefnominatie`

§ 2.1.1.9 Vertrouwelijkheidsaanduiding van een zaak ([zrc-009](#))

Indien de client een vertrouwelijkheidsaanduiding meegeeft bij het aanmaken of bewerken van een zaak, dan *MOET* de provider deze waarde toekennen. Indien de client deze niet expliciet toekent, dan *MOET* deze afgeleid worden uit `Zaak.ZaakType.vertrouwelijkheidsaanduiding`.

Een Zaak response van de provider *MOET* altijd een geldige waarde voor vertrouwelijkheidsaanduiding bevatten. Een client *MAG* een waarde voor vertrouwelijkheidsaanduiding meesturen.

§ 2.1.1.10 Valideren communicatiekanaal op de Zaak resource ([zrc-010](#))

Bij het aanmaken (`zaak_create`) en bijwerken (`zaak_update` en `zaak_partial_update`) *MOET* de URL-referentie naar het communicatiekanaal gevalideerd worden op het bestaan. Indien het ophalen van het zaaktype niet (uiteindelijk) resulteert in een HTTP 200 status code, *MOET* het ZRC antwoorden met een HTTP 400 foutbericht.

Het ophalen van deze resource moet een JSON-document zijn met de vorm van een communicatiekanaal zoals gedocumenteerd op de [referentielijsten-api](#):

```
{
  "url": "http://example.com",
  "naam": "string",
  "omschrijving": "string"
}
```

§ 2.1.1.11 Valideren relevanteAndereZaken op de Zaak-resource ([zrc-011](#))

De lijst `relevanteAndereZaken` bevat URL-referenties naar andere zaken. Elke URL-referentie *MOET* gevalideerd worden op het bestaan. Indien het ophalen van de url niet (uiteindelijk) resulteert in een HTTP 200 status code, *MOET* het ZRC antwoorden met een HTTP 400 foutbericht.

In het foutbericht *MOET* de naam binnen `invalid-params` dan `relevanteAndereZaken`. `<index>` zijn, waarbij index start bij 0.

§ 2.1.1.12 Gegevensgroepen ([zrc-012](#))

De client *MAG* gegevensgroepen zoals *Zaak.verlenging* en *Zaak.opschorting* meesturen met een waarde *null* om aan te geven dat er geen waarde gezet is. Dit is equivalent aan het niet-meesturen van de key in de request body. Als de client een (genest) object meestuurt, dan *MOET* de provider hierop de validatie van de gegevensgroep toepassen.

De provider *MOET* altijd de geneste structuur van de gegevensgroep antwoorden.

§ 2.1.1.13 Valideren hoofdzaak op de Zaak-resource ([zrc-013](#))

Bij het aanmaken of bewerken van een *Zaak* kan de hoofdzaak aangegeven worden. Dit *MOET* een geldige URL-referentie naar een *Zaak* zijn, indien opgegeven.

Indien de client een hoofdzaak opgeeft die zelf een deelzaak is (i.e. *Zaak.hoofdzaak* != *null*), dan moet het ZRC antwoorden met een HTTP 400 foutbericht (deelzaken van deelzaken zijn NIET toegestaan).

Indien de client een zaak bewerkt en diezelfde zaak als URL-referentie meegeeft als hoofdzaak, dan moet het ZRC antwoorden met een HTTP 400 foutbericht (een zaak *MAG* GEEN deelzaak van zichzelf zijn).

§ 2.1.1.14 *Zaak.betalingsindicatie* en *Zaak.laatsteBetaaldatum* ([zrc-014](#))

Indien de betalingsindicatie de waarde "nvt" heeft en een waarde gegeven is voor *laatsteBetaaldatum*, dan *MOET* het ZRC antwoorden met een HTTP 400 foutbericht. Bij alle andere waarden van *betalingsindicatie* *MAG* een waarde opgegeven worden voor *laatsteBetaaldatum*.

Indien een waarde ingevuld is voor *laatsteBetaaldatum* en de *betalingsindicatie* wordt gewijzigd naar "nvt", dan *MOET* de *laatsteBetaaldatum* op *null* gezet worden.

§ 2.1.1.15 Valideren van producten en/of diensten bij een Zaak ([zrc-015](#))

Bij het aanmaken (`zaak_create`) en bijwerken (`zaak_update` en `zaak_partial_update`) *MOET* de collectie `productenOfDiensten` worden getoetst tegen het `Zaaktype.productenOfDiensten` van het betreffende `zaaktype`. De producten en/of diensten van de zaak *MOETEN* een subset van de producten en/of diensten op het `zaaktype` zijn.

§ 2.1.1.16 Valideren tegen de catalogus en bijhorende typen

Het `zaaktype` van een zaak legt vast wat de mogelijke waarden zijn voor gerelateerde resources aan zaken. Dit *MOET* gevalideerd worden door een provider.

§ 2.1.1.17 Valideren dat het *statustype* van een *Status* bij het `Zaak.zaaktype` hoort ([zrc-016](#))

Documentatie toegevoegd in patch 1.0.1

Wanneer een *Status* bij een *Zaak* toegevoegd wordt, dan *MOET* het `Status.statustype` voorkomen in de `Status.zaak.zaaktype.statustypen`.

§ 2.1.1.18 Valideren dat het *informatieobjecttype* van een *ZaakInformatieObject.informatieobject* bij het `Zaak.zaaktype` hoort ([zrc-017](#))

Documentatie toegevoegd in patch 1.0.1

Wanneer een *ZaakInformatieObject* bij een *Zaak* toegevoegd wordt, dan *MOET* het `ZaakInformatieObject.informatieobject.informatieobjecttype` voorkomen in de `ZaakInformatieObject.zaak.zaaktype.informatieobjecttypen`.

§ 2.1.1.19 Valideren dat de *eigenschap* van een *ZaakEigenschap* bij het `Zaak.zaaktype` hoort ([zrc-018](#))

Documentatie toegevoegd in patch 1.0.1

Wanneer een `ZaakEigenschap` bij een `Zaak` toegevoegd wordt, dan *MOET* de `ZaakEigenschap.eigenschap` voorkomen in de `ZaakEigenschap.zaak.zaaktype.eigenschappen`.

TODO: [fix in ZRC](#)

§ 2.1.1.20 Valideren dat het `roltype` van een `Rol` bij het `Zaak.zaaktype` hoort ([zrc-019](#))

Documentatie toegevoegd in patch 1.0.1

Wanneer een `Rol` bij een `Zaak` toegevoegd wordt, dan *MOET* het `Rol.roltype` voorkomen in de `Rol.zaak.zaaktype.roltypen`.

§ 2.1.1.21 Valideren dat het `resultaattype` van een `Resultaat` bij het `Zaak.zaaktype` hoort ([zrc-020](#))

Documentatie toegevoegd in patch 1.0.1

Wanneer een `Resultaat` bij een `Zaak` toegevoegd wordt, dan *MOET* het `Resultaat.resultaattype` voorkomen in de `Resultaat.zaak.zaaktype.resultaattypen`.

§ 2.1.2 Archiveren

§ 2.1.2.1 Afleiden van archiveringsparameters ([zrc-021](#))

Het resultaat van een zaak is bepalend voor het archiefregime. Bij het afsluiten van een zaak *MOETEN* de attributen `Zaak.archiefnominatie` en `Zaak.archiefactiedatum` bepaald worden uit het `Zaak.Resultaat` als volgt:

1. Indien de zaak geen `archiefnominatie` heeft, dan *MOET* deze overgenomen worden uit `Resultaat.Resultaattype.archiefnominatie`
2. Indien `Resultaat.Resultaattype.archiefactietermijn` gevuld is:
 1. Bepaal de brondatum van de archiefprocedure
 1. Consulteer het groeppattribuut `Resultaat.Resultaattype.brondatumArchiefprocedure`

2. Afhankelijk van de waarde van afleidingswijze:

- afgehandeld -> gebruik `Zaak.einddatum`
- hoofdzaak -> gebruik `Zaak.hoofdzaak.einddatum`
- eigenschap -> gebruik de waarde van de eigenschap met als naam de waarde van
`Resultaat.Resultaattype.brondatumArchiefprocedure.datumkenmerk`
- ander_datumkenmerk -> `Zaak.archiefactiedatum` *MOET* handmatig afgeleid en gezet worden
- zaakobject -> zoek de gerelateerde objecten van type
`Resultaat.Resultaattype.brondatumArchiefprocedure.objecttype`. Lees van elk object het attribuut met de naam
`Resultaat.Resultaattype.brondatumArchiefprocedure.datumkenmerk` en gebruik de maximale waarde.
- termijn -> `Zaak.einddatum` +
`Resultaat.Resultaattype.brondatumArchiefprocedure.procestermijn`
- gerelateerde_zak -> maximale `Zaak.einddatum` van alle gerelateerde zaken
- ingangsdatum_besluit -> maximale `Besluit.ingangsdatum` van alle gerelateerde besluiten
- vervaldatum_besluit -> maximale `Besluit.vervaldatum` van alle gerelateerde besluiten

2. Zet `Zaak.archiefactiedatum` als `brondatum` + `Resultaat.Resultaattype.archiefactietermijn`

Indien de archiefactiedatum niet bepaald kan worden, dan *MAG* er geen datum gezet worden. Dit kan voorkomen als de brondatum niet bepaald kan worden of de archiefactietermijn niet beschikbaar is.

§ 2.1.2.2 Zetten `Zaak.archiefstatus` ([zrc-022](#))

De standaardwaarde voor archiefstatus is `nog_te_archiveren`. Indien een andere waarde gezet wordt, dan *MOETEN* alle gerelateerde informatieobjecten de status gearchiveerd hebben.

De attributen `Zaak.archiefnominatie` en `Zaak.archiefactiedatum` *MOETEN* een waarde krijgen als de de archiefstatus een waarde krijgt anders dan `nog_te_archiveren`.

Indien deze voorwaarden niet voldaan zijn, dan *MOET* het ZRC met een HTTP 400 foutbericht antwoorden.

§ 2.1.2.3 Vernietigen van zaken ([zrc-023](#))

Bij het verwijderen van een Zaak *MOETEN* de zaak en gerelateerde objecten daadwerkelijk uit de opslag verwijderd worden. Zogenaamde "soft-deletes" zijn NIET TOEGESTAAN. Onder gerelateerde objecten wordt begrepen:

- zaak - de deelzaken van de verwijderde hoofdzaak
- status - alle statussen van de verwijderde zaak
- resultaat - het resultaat van de verwijderde zaak
- rol - alle rollen bij de zaak
- zaakobject - alle zaakobjecten bij de zaak
- zaakeigenschap - alle eigenschappen van de zaak
- zaakkenmerk - alle kenmerken van de zaak
- zaakinformatieobject - relatie naar enkelvoudige informatieobjecten *
- zaakverzoek - relatie naar een zaak verzoek (**nieuw in 1.1.0**)
- zaakcontactmoment - relatie naar een zaak contactmoment (**nieuw in 1.1.0**)
- klantcontact - alle klantcontacten bij een zaak
- audittrail - de geschiedenis van het object

Een deelzaak KAN vernietigd worden zonder dat de hoofdzaak vernietigd wordt.

* Het verwijderen van een zaakinformatieobject in het ZRC leidt er toe dat het objectinformatieobject in het DRC ook verwijderd wordt indien dit kan.

§ 2.1.2.4 Bewaren van zaken ([zrc-026](#))

Nieuw in 1.3.0

De 'Startdatum bewaartermijn' markeert het einde van de Selectielijst-procestermijn en het begin van de Selectielijst-bewaartermijn. De periode waarover een zaakdossier na afronding van de zaak gearcheveerd blijft, bestaat in de Selectielijst uit twee gedeelten: achtereenvolgens de Procestermijn en de Bewaartermijn. De procestermijn eindigt bij het vervallen van het procesobject waarop de

zaak betrekking heeft (zie attribuutsoort Procesobjectaard). Dit is het startmoment van de bewaartermijn d.w.z. van de periode waarover het zaakdossier vervolgens bewaard dient te blijven.

De attribuutsoort wordt alleen van een waarde voorzien voor te vernietigen zaakdossiers. Voor altijd te bewaren zaakdossiers start de bewaartermijn op de datum van afronding van de zaak. De waarde van de attribuutsoort wordt zoveel als mogelijk bepaald gedurende de behandeling van de zaak, teneinde de archiefactiedatum (cq. datum vernietiging) te kunnen bepalen bij afronding van de zaak. In sommige gevallen is evenwel van het vervallen van het procesobject pas sprake nadat de zaak afgerond is. Een dergelijk procesobject moet gevolgd worden (m.b.v. de waarden van de groeppattribuutsoort 'Procesobject') teneinde het vervallen daarvan te constateren en alsnog de waarde van 'Startdatum bewaartermijn' te kunnen bepalen.

§ 2.1.2.5 HTTP-Caching

Nieuw in versie 1.1.0

De Zaken API moet HTTP-Caching ondersteunen op basis van de ETag header. In de API spec staat beschreven voor welke resources dit van toepassing is.

De ETag *MOET* worden berekend op de JSON-weergave van de resource. Verschillende, maar equivalente weergaves (bijvoorbeeld dezelfde API ontsloten wel/niet via NLX) *MOETEN* verschillende waarden voor de ETag hebben.

Indien de consumer een HEAD verzoek uitvoert op deze resources, dan *MOET* de provider antwoorden met dezelfde headers als bij een normale GET, dus inclusief de ETag header. Er *MAG* GEEN response body voorkomen.

Indien de consumer gebruik maakt van de If-None-Match header, met één of meerdere waarden voor de ETag, dan *MOET* de provider antwoorden met een HTTP 304 bericht indien de huidige ETag waarde van de resource hierin voorkomt. Als de huidige ETag waarde hier niet in voorkomt, dan *MOET* de provider een normale HTTP 200 response sturen.

§ 2.1.2.6 TODO: synchroniseren zaakcontactmomenten

§ 2.1.2.7 Synchroniseren relaties met verzoeken ([zrc-024](#))

Nieuw in versie 1.1.0

Wanneer een relatie tussen een VERZOEK en een ZAAK gemaakt of bijgewerkt wordt, dan *MOET* het ZRC in de Verzoeken API ook deze relatie aanmaken/bijwerken.

Een voorbeeld:

1. Een verzoek wordt gerelateerd aan een zaak door een consumer:

```
POST https://zrc.nl/api/v1/zaakverzoeken HTTP/1.0

{
  "verzoek": "https://vrc.nl/api/v1/verzoeken/1234",
  "zaak": "https://zrc.nl/api/v1/zaken/456789",
}
```

2. Het ZRC *MOET* de relatie spiegelen in de Verzoeken API:

```
POST https://vrc.nl/api/v1/objectverzoeken HTTP/1.0

{
  "verzoek": "https://vrc.nl/api/v1/verzoeken/1234",
  "object": "https://zrc.nl/api/v1/zaken/456789",
  "objectType": "zaak"
}
```

Merk op dat het aanmaken van de relatie niet gelimiteerd is tot het aanmaken via de API. Indien elders (bijvoorbeeld via een admininterface) een relatie tot stand kan komen, dan *MOET* deze ook gesynchroniseerd worden.

§ 2.1.2.8 Synchroniseren relaties met contactmomenten ([zrc-025](#))

Nieuw in versie 1.1.0

Wanneer een relatie tussen een CONTACTMOMENT en een ZAAK gemaakt of bijgewerkt wordt, dan *MOET* het ZRC in het CRC ook deze relatie aanmaken/bijwerken.

Nieuw in versie 1.5.0

§ 2.1.2.9 Reikwijdte expand parameters ([zrc-025](#))

Indien een verzoek één of meer expand parameters bevat *MOET* deze parameter alleen informatie uit de Zaken API of gerelateerde informatie uit de Catalogi API bevatten. Indien een expand

parameter om informatie uit andere bronnen vraagt moet een foutmelding (http 406?) worden teruggegeven.

§ 2.1.2.10 Diepte uitvoeren expand parameters ([zrc-026](#))

Indien een verzoek één of meer expand parameters bevat *MOET* de expand niet dieper gaan dan maximaal 3 niveaus diep. Wanneer een consumer een diepere expand opgeeft *MOET* het antwoord maximaal de 3 niveaus diep gaan. Hiermee kan de volgende informatie in een response opgenomen worden:

```
Zaak
  zaaktype
  status
    statustype
  gerelateerdeZaak
    zaaktype
    status
      statustype
```

§ 2.1.2.11 Expand parameter onderdeel van opgevraagde resource ([zrc-026](#))

Indien een verzoek één of meer expand parameters bevat *MOET* het attribuut onderdeel zijn van de opgevraagde resource. Indien een expand parameter geen geldig attribuut is van de opgevraagde resource moet een foutmelding (http 404) worden teruggegeven.

§ 2.1.2.12 Gedrag bij fouten in expand parameters ([zrc-027](#))

Op een verzoek *MOET* een geldige response zoals deze opgevraagd is opleveren. Indien een verzoek één of meer expand parameters bevat *MOET* ook de te expanderen informatie opgehaald en teruggegeven kunnen worden. Indien geen geldige response kan worden teruggegeven moet een foutmelding (http 404) worden teruggegeven.

§ A. Referenties

§ A.1 Normatieve referenties

[RFC2119]

Key words for use in RFCs to Indicate Requirement Levels. S. Bradner. IETF. March 1997.

Best Current Practice. URL: <https://www.rfc-editor.org/rfc/rfc2119>

[RFC8174]

Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words. B. Leiba. IETF. May 2017.

Best Current Practice. URL: <https://www.rfc-editor.org/rfc/rfc8174>