

Handreiking "Hoe bouw ik een register" 0.0.1

Samenvatting

In dit deel van het document kan een samenvatting of een leeswijzwer worden opgenomen

- Je kan bijvoorbeeld hoofdstuk wat toelichten.
- En dan hier over Hoofdstuk 2 iets toelichten
- Uiteraard zou je hier ook een toelichting op kunnen nemen over hoe dit document gebruikt kan worden. Bijvoorbeeld dat het geen "menu" of "kookboek" voor een capabel register is, maar dat het een aantal onderwerpen beschrijft en daarbij overwegingen meegeeft om uiteindelijk tot weloverwogen keuzes te kunnen komen.

Inhoudsopgave

Samenvatting

1. **Command**
 - 1.1 Twijfel en dry-run
 - 1.2 Transacties
2. **De omgeving en grenzen van een register**
 - 2.1 Autonomie en gemeenschappelijkheid
 - 2.2 Registergrenzen: domein
 - 2.3 Bounded context
 - 2.4 'Wellbounded' registers
 - 2.5 Bepalen van de 'bounds'
 - 2.6 Uitvoeringsproces op hoofdlijnen
 - 2.6.1 Het domeinmodel als kern
 - 2.6.2 Signaal ontvangen en interpreteren
 - 2.6.3 Taken definiëren en uitvoeren
 - 2.6.4 Gevolgen vastleggen
 - 2.6.5 Informatie beschikbaar stellen
 - 2.7 Applicatieve ondersteuning met het register
 - 2.7.1 Van proces naar applicatie
 - 2.7.2 Notificatie → Command (Signaalverwerking)
 - 2.7.3 Command → Effecten (Taakuitvoering)
 - 2.7.4 Effecten → Projecties (Presentatieopmaaking)
 - 2.7.5 Register = Commands + Effecten + Projecties
3. **Samenstellen Respec documentatie in GitHub (under construction)**
 - 3.1 Door administrator uit te voeren acties
 - 3.2 Door repository eigenaar uit te voeren acties
 - 3.2.1 Invertor resultaat plaatsen
 - 3.2.2 De content van het Respec document aanpassen
 - 3.2.2.1 Content methode
 - 3.2.2.2 Sectie methode
 - 3.2.2.3 Secties met 'id' attribuutwaarde 'sotd'
 - 3.2.2.4 Secties met 'id' attribuutwaarde 'abstract'
 - 3.2.2.5 Secties met 'id' attribuutwaarde 'conformance'
 - 3.2.2.6 Secties met 'id' attribuutwaarde 'tof'
 - 3.2.2.7 Secties met 'id' attribuutwaarde 'index'
 - 3.2.2.8 Secties met een andere 'id' attribuutwaarde
 - 3.2.2.9 Secties met `data-include-format="html"`
 - 3.2.2.10 Andersoortige secties
 - 3.2.3 Bijlage N Referenties
 - 3.2.4 Images in de documentatie
 - 3.3 Lokale Respec configuratie properties

3.4	Functie Respec configuratie properties
4. Conformiteit	
A.	Index
A.1	Begrippen gedefinieerd door deze specificatie
A.2	Begrippen gedefinieerd door verwijzing
B. Referenties	
B.1	Normatieve referenties

1. Command

- Werkdefinitie: een command is een opdracht aan het register, aangeboden met de intentie daarin een verandering aan te brengen.
- Aangepast vanwege een 5 testen

1.1 Twijfel en dry-run

- Aan gegevens in een capabel register kan getwijfeld worden. Dit heeft gevolgen voor het geautomatiseerd verwerken van commando's. Systemen kunnen immers niet (altijd) (geautomatiseerd) commando's verwerken als bij het beoordelen daarvan betwijfelde gegevens betrokken zijn. Dit is niet per se een registervraagstuk. Wel een vraagstuk dat zich op dit moment veelal buiten registers - en wel binnen procesondersteunende systemen - afspeelt.
- Willen we registers en degenen die ze bedienen (beter) laten werken met betwijfelde gegevens, dan is één oplossing het aanbieden van interfaces die het mogelijk maken daarover met het register te converseren. Wanneer sprake is van twijfel over de juistheid van relevante gegevens, kan in zo'n conversatie van verwerkingsregels worden afgeweken. Mits beargumenteerd en voorzien van instructies over hoe en waar specifieke regels wel of niet moeten worden toegepast, wordt een waarschuwing of error zo dus 'overrulebaar'.
- Hieraan gerelateerd is het idee van 'dry run-functionaliteit'. Dit is een query die antwoord geeft op de vraag of een command verwerkt kan worden zonder een poging te doen dat command daadwerkelijk te verwerken. Om verschillende uitkomsten tussen 'dry run' en daadwerkelijke registratie te voorkomen, ligt het - ondanks scheiding tussen command en query (CQRS) - voor de hand deze query te laten beantwoorden aan de commandkant van het register, waar immers ook de daadwerkelijke verwerking van het command moet plaatsvinden.

1.2 Transacties

Conventie (en aanbeveling?):

- Command is een opdracht die één transactie verwerkt moet worden.

...maar: business bepaalt in eerste instantie transactiegrenzen, commands sluiten daarbij aan.

...maar: digitaal werken kan innovatie in business mogelijk maken. Sneller (en sequentieel) verwerken van wijzigingen in 'echte' wereld kan bijvoorbeeld consistentievaborgen van aggregaatniveau naar register niveau (dat daardoor in feite zelf aggregaat wordt?) brengen.

2. De omgeving en grenzen van een register

Een register heeft betekenis door het proces dat het ondersteunt. Om te begrijpen waar de grenzen liggen, beginnen we bij het vijflaagsmodel van Common Ground:

Laag	Naam	Wat
5	Interactie	Gebruikersinterfaces, kanalen en toegang
4	Procesinrichting	Bedrijfsprocessen, bedrijfsregels en informatiestructuren
3	Connectiviteit	Veilige verbindingen
2	Diensten	Datadiensten, APIs en services
1	Databronnen	Registers, bijhouding en vastlegging
(0)	Infrastructuur	Hardware en infrastructuur

Een register bevindt zich in laag 1 (databronnen) en laag 2 (diensten). Een register sluit aan bij laag 4 (procesinrichting) door de **informatiestructuren** van het proces: de conceptuele modellen die beschrijven welke informatie het proces nodig heeft en produceert.

2.1 Autonomie en gemeenschappelijkheid

Binnen [Common Ground](#) ligt de **autonomie van gemeenten** bij de interactielag (hoe burgers worden bediend) en de procesbesturing (hoe het werk wordt georganiseerd). Elke gemeente kan eigen keuzes maken in werkwijze en gebruikersinteractie.

De **common ground** - wat gemeenschappelijk is - ligt bij het domeinmodel: welke acties mogelijk zijn binnen een domein en welke gevolgen deze hebben. Dit zorgt voor interoperabiliteit terwijl gemeenten hun autonomie behouden.

Gemeenten zijn hier gebruikt als voorbeeld, want dit geldt voor alle organisaties waar autonomie en gemeenschappelijkheid een rol spelen, oftewel een gedeeld domein betreffen.

2.2 Registergrenzen: domein

Een register is een verzameling geordende informatie. De grenzen van de verzameling worden net als de betekenis en samenhang van de in het register opgeslagen informatie bepaald vanuit het **domein** dat voor het register verantwoordelijk is. Anders dan in de [wiskunde](#) kunnen we voor registers niet precies aangeven waar het ene domein ophoudt en het volgende begint. Analoog aan de definitie van [Eric Evans in de context van Domain Driven Design](#) beschouwen we een domein daarom als "een sfeer van kennis, invloed of activiteit". Deze definitie veronderstelt een zekere mate van samenhang. Tegelijkertijd kan het binnen één domein voorkomen dat:

- dezelfde concepten verschillend geïnterpreteerd worden,
- regels en gedrag niet consistent zijn, en
- bedrijfsprocessen niet op elkaar zijn afgestemd.

Voor wie zoekt naar aanknopingspunten over welke informatie binnen 'hoort' binnen een te ontwerpen register en welke buiten de registergrenzen zou moeten vallen, biedt het denken in 'sferen' onvoldoende aanknopingspunten. Daarom introduceren we **bounded context** als begrenzend begrip. Ook dit concept is ontleend aan [\[domain-driven-design\]](#).

2.3 Bounded context

Binnen een domein kunnen meerdere subdomeinen of taakgebieden bestaan. Daarbij horen verantwoordelijkheden, die zijn toegekend aan specifieke teams, afdelingen of bedrijfseenheden. Zij worden ondersteund door eigen semantiek, processen en regels. Deze zaken kunnen worden beschreven in een **model**: een systeem van abstracties dat beschrijft hoe men vanuit een taakgebied naar de wereld kijkt en reageert op veranderingen in de buitenwereld. Zo'n model is beschreven in **gemeenschappelijke taal** die binnen het hele taakgebied begrepen wordt. Model en taal beschrijven dus een voor betrokkenen herkenbare 'blauwdruk' van het taakgebied, die (onder andere) de basis kan vormen voor het ontwerp van een register. Een in gemeenschappelijke taal ondubbelzinnig en samenhangend gemodelleerd taakgebied noemen we een '**bounded context**'.

Het belang van het erkennen van bounded contexten wordt door Eric Evans in '[the Blue Book](#)' als volgt beschreven:

"A bounded context delimits the applicability of a particular model so that team members have a clear and shared understanding of what has to be consistent and how it relates to other contexts. Within that context, work to keep the model logically unified, but do not worry about applicability outside those bounds. In other contexts, other models apply, with differences in terminology, in concepts and rules, and in dialects of the ubiquitous language [*red, gemeenschappelijke taal*]. By drawing an explicit boundary, you can keep the model pure, and therefore potent, where it is applicable. At the same time, you avoid confusion when shifting your attention to other contexts. Integration across the boundaries necessarily will involve some translation, which you can analyze explicitly."

Uit het bovenstaande blijkt dat de ambiguïteiten die we op domeinniveau nog konden tegenkomen binnen een bounded context (idealiter) verdwijnen. Hier geldt (zoveel mogelijk) dat:

- dezelfde concepten eenduidig geïnterpreteerd worden (op basis van gemeenschappelijke taal),
- regels en gedrag consistent zijn, en
- bedrijfsprocessen op elkaar aansluiten.

2.4 'Wellbounded' registers

Bestaande registers bestrijken vaak meerdere bounded contexten. Zo zou je bijvoorbeeld binnen de Basisregistratie Personen (BRP) bijgehouden informatie over verblijfplaats, verblijfsrecht, kiesrecht en reisdocumenten ieder als 'eigen' bounded context kunnen beschouwen. Als je dat onderschrijft en op basis daarvan nieuwe registers zou gaan ontwerpen en ontwikkelen, loop je tegen een aantal (nieuwe) problemen aan:

- **Integratiecomplexiteit** (met name ten opzichte van registers met een 'breder' bereik). Registers moeten worden verbonden middels API's, events of gegevenssynchronisatie.
- **Hoge initiële leercurve.** Ontwerp en ontwikkeling van registers vereist diepgaande domeinkennis.
- **Lagere ontwikkelsnelheid.** Opdoen van domeinkennis en omzetten daarvan naar modellen en code vraagt tijd.

Tegelijkertijd worden aan overheidsregisters bijzondere eisen gesteld, onder meer als het gaat om kwaliteit en betrouwbaarheid. Onderstaande voordelen van het koppelen van registers aan een (of één) bounded context ondersteunen deze eisen, en wegen wat ons betreft zwaarder dan de nadelen:

- **Duidelijkheid.** Registers omvatten ondubbelzinnige terminologie en logica.
- **Consistentie.** Registers omvatten logisch consistente set regels en modellen.
- **Autonomie.** Registers in verschillende bounded contexten kunnen onafhankelijk van elkaar werken en (door)ontwikkeld worden.

Daarom doen we de volgende aanbevelingen voor het begrenzen van registers:

1. Een register valt samen met een (en één) bounded context, en dus
2. bepalen de bijhoudingsverantwoordelijke(n) de registergrenzen

2.5 Bepalen van de 'bounds'

In de praktijk laten de 'bounds' van een context zich niet altijd gemakkelijk herkennen en afbakenen. Bovendien kunnen bounded contexten onder invloed van impliciete of uitgesproken belangen worden 'opgerekt' of juist verkleind. In algemene zin is moeilijk te zeggen wanneer voor een register de 'ideale' bounded context is gevonden. Wel zijn enkele aanwijzingen voor een te ruime of te krappe afbakening te geven. Deze zijn hieronder beschreven.

Indicatoren en symptomen (van sterk naar zwakker) voor een te ruime bounded context:

1. **Gebrek aan gemeenschappelijke taal.** Er is discussie over concepten en betekenis, betrokkenen hanteren verschillend jargon.
2. **Regels en gedrag niet in samenhang te brengen.** Het lukt niet tot een samenhangend model te komen.
3. **Afwijkende werkwijze(n).** Bedrijfsprocessen zijn niet te verenigen.
4. **Ontwerp-/ontwikkeltraject heeft onwenselijke overhead.** Het ontwerp-/ontwikkelteam is te groot (geworden).
5. **(Vermijdbare) overschrijding van team- en/of organisatiegrenzen.** Conflict tussen eigenaren en/of verantwoordelijken.
6. **Registeronderdelen vereisen verschillende opslag- en/of integratietechniek.** Niet-verenigbare technische vereisten.

Indicatoren en symptomen (van sterk naar zwakker) voor een te krappe bounded context:

1. **Registeroverstijgende consistentieaborgen vereist.** Saga's zijn nodig om transacties gedistribueerd af te handelen.
2. **Model beschrijft domein zonder zelfstandig bestaansrecht.** Gegevens in een register hebben 'los' geen enkel bestaansrecht en zijn niet te generaliseren voor gebruik in andere domeinen.

2.6 Uitvoeringsproces op hoofdlijnen

Het uitvoeringsproces binnen een domein volgt een herkenbaar patroon. Door dit patroon te begrijpen, wordt duidelijk waar het register precies bijdraagt en waar de grenzen liggen.

// **TODO** Vereenvoudigd plaatje van een administratief uitvoeringsproces (process flow plaatje?): Signaal -> Taak -> Gevolg -> Presentatie

Stap 1: Signaal ontvangen Een proces start altijd met een signaal - een melding dat er iets is gebeurd wat aandacht vraagt. Dit signaal kan van buiten het domein komen (bijvoorbeeld een melding van een burger) of van binnen (bijvoorbeeld een

(automatische) controle die iets detecteert).

Stap 2: Taken creëren Het proces beoordeelt het signaal en bepaalt welke acties nodig zijn. Deze acties worden vastgelegd als taken. Een signaal kan leiden tot één taak, meerdere taken, of soms geen taken als geen actie nodig is.

Stap 3: Taken uitvoeren Elke taak wordt uitgevoerd volgens de regels van het domein. Dit kan handmatige verificatie inhouden, automatische berekeningen, of andere domeinspecifieke activiteiten.

Stap 4: Gevolgen vastleggen Het uitvoeren van taken levert gevolgen op - dit zijn de definitieve veranderingen die het domein accepteert als geldig. Deze gevolgen vormen de waarheid voor dit domein. Ze beschrijven wat er is gebeurd: een persoon is verhuisd, een huwelijk is voltrokken, een kind is geboren, een bedrijf is ingeschreven, een eigendom is overgedragen.

Stap 5: Presentatie opmaken De gevolgen worden niet rechtstreeks gedeeld, maar gebruikt om projecties op te maken die aansluiten bij specifieke informatiebehoeften. Een verhuizing kan bijvoorbeeld leiden tot verschillende presentaties: een uittreksel voor de gemeente, een adreswijziging voor de belastingdienst, of een notificatie voor de zorgverzekeraar. Elke presentatie^[^1] toont alleen de informatie die relevant is voor de ontvangende partij.

Deze cyclus vormt de kern van elk register-ondersteund proces. Het register bewaart de gevolgen en stelt ze beschikbaar voor raadpleging.

 Gedetailleerd uitvoeringsproces dat het register ondersteunt *Figuur 2: Gedetailleerd uitvoeringsproces dat het register ondersteunt*

2.6.1 Het domeinmodel als kern

Het **domeinmodel** definieert wat binnen het domein mogelijk is. Het beschrijft:

- **Taakdefinities:** Welke acties mogelijk zijn binnen het domein
- **Taakverwerkingsregels:** Hoe taken worden uitgevoerd en welke gevolgen ze hebben
- **Presentatiedefinities:** Welke informatie in welke vorm beschikbaar gesteld wordt
- **Presentatieverwerkingsregels:** Hoe gevolgen worden omgezet naar specifieke informatiebehoeften

Signaalverwerking staat buiten het domeinmodel. Allerlei signalen kunnen op het proces afkomen. De signaalverwerking interpreteert deze naar het domeinmodel toe: welke van de beschikbare taken moet worden uitgevoerd?

Het domeinmodel vormt de kern van wat het register ondersteunt - de mogelijke acties en hun gevolgen.

2.6.2 Signaal ontvangen en interpreteren

- Een gebeurtenis binnen of buiten het domein leidt tot een *Signaal*
- *Signaalverwerking* beoordeelt of dit signaal relevant is voor het domein
- Relevante signalen worden vertaald naar acties die het domein kan uitvoeren

2.6.3 Taken definiëren en uitvoeren

- Uit het signaal ontstaan één of meerdere *Taken* die binnen het domein uitgevoerd kunnen worden
- *Taakuitvoering* voert deze taken uit volgens de regels van het *Domeinmodel*
- Taken volgen *Taakdefinities* en *Taakverwerkingsregels* uit het domeinmodel

2.6.4 Gevolgen vastleggen

- Uitgevoerde taken leiden tot *Gevolgen* - de daadwerkelijke veranderingen in het domein
- Deze gevolgen vormen de nieuwe waarheid voor dit domein

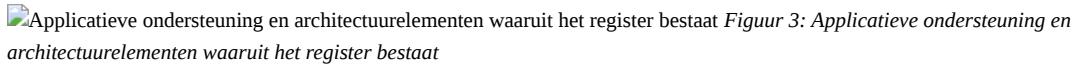
2.6.5 Informatie beschikbaar stellen

- *Presentatieopmaking* creëert verschillende *Presentaties* op basis van de gevolgen

- Elke presentatie is afgestemd op specifieke informatiebehoeften volgens *Presentatiedefinities* en *Effectverwerkingsregels* uit het domeinmodel
- Presentaties kunnen als nieuwe signalen dienen voor vervolgprocessen in eigen of andere domeinen

2.7 Applicatieve ondersteuning met het register

Het uitvoeringsproces krijgt concrete vorm in een applicatiearchitectuur. Het register ondersteunt dit proces door de juiste componenten en datastromen in te richten. Ultiem is een (business) domeinmodel gelijk aan een register, één *bounded context* met eigen regels en begrippen, zoals bedoeld in [[domain-driven-design](#)]. Veel vaker is het zo dat er meerdere bounded contexten te onderscheiden zijn in één business domeinmodel[^2]. Dit heeft te maken met verantwoordelijkheden en technologie *stacks* en hebben vaak ook historische redenen; de verzameling van applicaties die door de tijd heen ontwikkeld zijn. Toch tekenen we hier de eenvoudig en ultieme situatie als uitgangspunt.

Applicatieve ondersteuning en architectuurelementen waaruit het register bestaat *Figuur 3: Applicatieve ondersteuning en architectuurelementen waaruit het register bestaat*

2.7.1 Van proces naar applicatie

Het abstracte proces wordt concreet door deze architectuurcomponenten:

2.7.2 Notificatie → Command (Signaalverwerking)

- **Notificatieprocessor** ontvangt *notificaties* van binnen en buiten het domein
- Beoordeelt relevantie voor het eigen domein door raadpleging van projecties
- Vertaalt relevante notificaties naar *commands* volgens domeinconventies

2.7.3 Command → Effecten (Takuitvoering)

- **Commandprocessor** ontvangt commands en bepaalt welke *effecten* ontstaan
- Voert domeinregels uit en valideert tegen bestaande gegevens via projecties
- Effecten beschrijven de daadwerkelijke veranderingen binnen het domein

2.7.4 Effecten → Projecties (Presentatieopmaking)

- **Effectprocessor** maakt *projecties* op basis van effecten
- Projecties bevatten alleen informatie uit het eigen domein ('data bij de bron')
- Elke projectie is afgestemd op specifieke (groepen van)[^1] informatiebehoeften

2.7.5 Register = Commands + Effecten + Projecties

Het register bestaat uit:

- **Commands:** De opdrachten aan het register, aangeboden met de intentie daarin een verandering aan te brengen
- **Effecten:** De waarheid van het domein - wat er werkelijk is gebeurd
- **Projecties:** Verschillende views op deze waarheid voor verschillende gebruikers

Syntheses[^3] (views die meerdere domeinen combineren) vallen buiten het register en verdienen eigen architectuur.

[^1]: Dit is conceptueel juist, maar zeker geen dagelijkse praktijk. In de informatiebehoeften van verschillende afnemers zijn vaak overeenkomstigheden te vinden, zodat groepering voor de hand. Richtlijnen en overwegingen gaan nog (elders) beschreven worden.

[^2]: [No, Your Domains and Bounded Contexts Don't Map 1 on 1](#) - Mathias Verraes.

[^3]: Naar [Chronolexografie](#), een conceptueel model en specifieke architectuur voor het digitaal vastleggen en bijhouden van de rechtstoestand. De *projecties* in bovenstaand artikel zijn te relateren aan de *reductie* uit Chronolexografie. Met

synthese wordt in beide artikelen geduid op analyse en *view* op meer dan alleen projecties en reducties uit registers. Het betreft vaak uitgebreidere verwerkingen, transformaties en combinatie met meerdere projecties uit verschillende registers.

3. Samenstellen Respec documentatie in GitHub (under construction)

De acties die in het voorgaande hoofdstuk staan beschreven leveren een html bestand voor de Respec documentatie op waarin een informatiemodel wordt beschreven. Respec documentatie hoeft echter niet persé over informatiemodellen te gaan, voor de Respec documentatie die je nu leest is dat immers ook niet het geval. Het resultaat van het voorgaande hoofdstuk kan samen met andere html of markdown bestanden worden gebundeld tot de Respec documentatie. Daarnaast wordt een deel van de content van de Respec documentatie door het Respec framework in GitHub gegenereerd a.d.h.v. een aantal variabelen. Dat framework verzorgt daarnaast ook de vormgeving dat essentieel is voor de Respec documentatie.

Binnen VNG-R maken we gebruik van een door Logius vervaardigde extensie op het W3C Respec framework. We volgen daarbij andere organisaties in Nederland die hetzelfde doen zoals Geonovum. Van het door Logius beschikbaar gestelde template is een VNG-R versie beschikbaar binnen de VNG-Realisatie GitHub organisatie. Dat geeft de mogelijkheid om te verwijzen naar een VNG-R Respec configuratie waardoor we specifiek voor VNG-Realisatie geldende configuraties, zoals bijv. het VNG-Realisatie logo, kunnen aanbrengen. Deze vind je in de repository 'Respec-Organization-configurations'. Het template zelf kan echter door eenieder worden gebruikt om de eigen Respec documentatie te vervaardigen en daarbinnen bestaan nog mogelijkheden om jouw Respec documentatie een invulling tintje te geven.

Hieronder wordt de werkwijze beschreven waarbij de 8 in de volgende paragraaf beschreven stappen moeten worden uitgevoerd door een GitHub organisatie administrator. Voorzie hem daarvoor van de gewenste repository naam.

3.1 Door administrator uit te voeren acties

1. Open het [VNG-R Respec template](#) en klik in de README op die pagina op de link 'Use this template';
2. Je komt nu in het menu om een nieuwe repository aan te maken waarbij al een aantal velden zijn ingevuld. De te maken repository mag niet private zijn want dat maakt het gebruik van GitHub Pages onmogelijk. Geef de van de aanvrager verkregen repository naam in en klik op 'Create repository';
3. Voer de acties, zoals beschreven in [de handleiding voor het initieel inrichten van GitHub repositories](#), uit;
4. Verwijder in de root van de repository het 'README.md' bestand en hernoem 'Alt-README.md' naar 'README.md'

Dat bestand moet nog gecreëerd worden in het template;

6. Activeer GitHub Pages voor de nieuwe repository. Selecteer daarvoor het tabblad 'Settings' en kies daar 'Pages';
7. Kies daar waar bij Branch 'None' staat voor 'main' en klik op 'Save';
8. Nadat de build en deployment is uitgevoerd ga je naar het 'Code' tabblad, klikt daar op het tandwieltje bij 'About' en klikt op de checkbox naast 'Use your GitHub Pages website'. Klikken op de resulterende link onder 'About' brengt je naar de standaard gegenereerde Respec documentatie die nu kan worden aangepast door de eigenaar van de repository;

3.2 Door repository eigenaar uit te voeren acties

Je beschikt nu over een repository die je kunt gaan vullen en waarin je je persoonlijke configuratie properties van een waarde kunt voorzien. Indien je een met Imvertor gegenereerd Respec html bestand wil gebruiken dan volg je de beschrijving van de volgende paragraaf, zo niet dan ga je direct naar de daarop volgende paragraaf.

3.2.1 Imvertor resultaat plaatsen

Plaats het met Imvertor gegenereerde bestand in de root van de repository. Van dat bestand gebruiken we alleen de inhoud van het 'section' element met het id 'cat'. Het section element zelf gebruiken we dus niet. Verwijder alle andere content behalve de processing instruction 'DOCTYPE HTML' aan het begin van dit bestand en commit het bestand. Open vervolgens het bestand 'index.html' en plaats daarin op de gewenste locatie het volgende html fragment:

```
<section id="XXXX" data-include-format="html" data-include="XXXX.html"></section>
```

Waarbij je 'XXXX.html' vervangt door de naam van het zojuist aangepaste bestand en 'XXXX' door een id dat de sectie duidelijk en uniek identificeert.

3.2.2 De content van het Respec document aanpassen

Een Respec document kan op 2 verschillende manier van content worden voorzien:

- door de 'sectie' elementen aan het 'index.html' bestand toe te voegen.
- m.b.v. de 'content' configuratie property;

Beide methodes kunnen naast elkaar worden gebruikt. Advies is echter om de eerste methode te gebruiken. Deze is transparanter omdat met 1 blik op het index.html bestand te zien is wat er in wordt opgenomen.

Het Respec document zoals dat van het VNG-R Respec template is overgenomen moet nog aangepast worden. Deels kan dat door in de 'index.html' secties aan te passen danwel te vervangen en deels door de configuration property 'content' aan te passen.

3.2.2.1 Content methode

M.b.v. de 'content' configuratie property kunnen alleen secties waarvan de content in markdown bestanden staan worden toegevoegd. In deze property kan per bestand worden aangegeven of die sectie informatief is. Is dat het geval dan wordt automatisch de tekst **Dit onderdeel is niet normatief.** aan het hoofdstuk toegevoegd. Het toevoegen van bestanden aan de 'content' configuratie property doe je door de naam van het bestand (zonder de extensie) en een eventueel relevante CSS class in de property te plaatsen. De volgorde van bestanden binnen content bepaalt de volgorde in het resulterende document.

De code content: {"ch01": "informative", "mermaid": ""}, voegt 2 markdown bestanden toe, te weten:

- ch01.md met de CSS class **informative**;
- mermaid.md zonder CSS class.

Voor een volledige lijst van CSS classes zie de [ReSpec Documentation](#). Deze classes zijn ook binnen de markdown files te gebruiken op de volgende manier:

```
<div class="example">voorbeeld</div>
```

Het gebruik van de 'content' properties is niet verplicht, er mag voor worden gekozen nieuwe content alleen toe te voegen door het 'index.html' bestand aan te passen. De 'content' property moet dan wel uit het lokale 'js/config.js' bestand worden verwijderd of worden uitbemerkteerd. Ook kan de plaats waar de in 'content' gedefinieerde hoofdstukken moeten worden toegevoegd worden aangepast. Zorg er dan voor dat het 'section' element waarna je die chapters wil toevoegen een 'id' attribuut met een waarde heeft en wijzig in het script in 'index.html' de regel

```
document.getElementById("id-van-sectie").insertAdjacentHTML('afterend', content);
```

zodanig dat de waarde 'id-van-sectie' de waarde van het id heeft.

3.2.2.2 Sectie methode

In tegenstelling tot de methode met de 'content' configuratie property kunnen aan het 'index.html' bestand zowel 'sectie' elementen worden toegevoegd waarvan de content uit markdown bestaat als 'sectie' elementen waarvan de content uit html bestaat. Aangezien het gegenereerde Respec bestand een html bestand is kunnen we het alleen toevoegen aan het Respec document door een 'sectie' element toe te voegen aan het index.html bestand.

Bij de methode met de 'section' elementen maken we nog verschil tussen 'sectie' elementen met specifieke waarden voor het 'id' attribuut en 'sectie' elementen die andere waarden voor dat 'id' attribuut hebben of die zelfs helemaal geen 'id' attribuut hebben.

In de onderstaande paragrafen volgt per sectie een toelichting.

3.2.2.3 Secties met 'id' attribuutwaarde 'sotd'

Toe te voegen m.b.v. <section id="sotd"></section>. Leidt ertoe dat het hoofdstuk met de titel 'Status van het document' wordt toegevoegd met als inhoud de, van de waarde van de configuration property 'specStatus' afhankelijke, content van de configuration property 'sotdText'.

Tevens wordt een TOC gegenereerd waarin de titels (incl. evt. hoofdstuk en paragraafnummers) van alle, in het document opgenomen, hoofdstukken en paragrafen worden opgenomen afhankelijk van de configuratie property 'maxTocLevel'. Ook

de titels van 'sectie' elementen zonder 'id' attribuut worden daar opgenomen.

Indien de configuration property 'content' bestaat dan worden de daarin gedefinieerde markdown bestanden na de 'sotd' sectie opgenomen. Zo niet dan worden de in de 'content' configuratie property gedefinieerde secties ook niet toegevoegd en wordt er ook geen TOC gegenereerd.

3.2.2.4 Secties met 'id' attribuutwaarde 'abstract'

Indien de sectie wordt toegevoegd met `<sectie id="abstract" data-include-format="markdown" data-include="filenaam.md">` dan krijgt het hoofdstuk de titel Samenvatting zonder hoofdstuknr. als inhoud wordt de inhoud van het bestand 'filenaam.md' toegevoegd.

3.2.2.5 Secties met 'id' attribuutwaarde 'conformance'

Door `<section id='conformance'></section>` wordt een hoofdstuk met als titel 'Conformiteit' toegevoegd.

De inhoud komt waarschijnlijk uit <https://github.com/Logius-standaarden/respec>. Het is nog niet duidelijk hoe dit hoofdstuk zijn inhoud krijgt.

3.2.2.6 Secties met 'id' attribuutwaarde 't0f'

`<section id='t0f'></section>` genereert een hoofdstuk met als titel 'Lijst met Figuren' als er in minimaal een van de opgenomen bestanden minimaal een html 'figure' element met een 'figcaption' element is opgenomen of een markdown equivalent daarvan ('!-[Tekstueel alternatief voor toegankelijkheid](pad naar illustratie bestand "Onderschrift")'). In de markdown variant mag het onderschrift ontbreken.

De titel komt waarschijnlijk uit <https://github.com/Logius-standaarden/respec>. Het is nog niet duidelijk hoe die titel wordt toegekend.

3.2.2.7 Secties met 'id' attribuutwaarde 'index'

`<section id="index"></section>` genereert een hoofdstuk met als titel 'Bijlage N Index' als er in minimaal 1 van de in het document opgenomen bestanden (zowel markdown als html) minimaal 1 'dfn' element is opgenomen. Vanuit de tekst kan naar dat element verwezen worden door een 'a' element op te nemen zonder attributen maar met als inhoud de naam van een 'dfn' element.

3.2.2.8 Secties met een andere 'id' attribuutwaarde

- Indien de sectie wordt toegevoegd met `<sectie id="nnnnnn" data-include-format="markdown" data-include="filenaam.md">` dan wordt het hoofdstuk gevuld met de inhoud van 'filenaam.md'. Als 'filenaam.md' met een markdown titel start (ongeacht het level en het aantal blanco regels er voor) dan wordt een hoofdstuknummer voor die titel gegenereerd anders wordt de content zonder titel toegevoegd aan het document. Een evt. titel wordt ook opgenomen in de TOC.
- Indien de sectie wordt toegevoegd met `<sectie data-include-format="markdown" data-include="filenaam.md">` dan wijkt het resultaat niet af van die van hierboven. Alleen wordt bij deze variant het 'id' van de sectie en de gerelateerde 'href' in de TOC gegenereerd op basis van de titel van deze sectie.

In alle gevallen is `data-include-format="markdown"` verplicht.

3.2.2.9 Secties met `data-include-format="html"`

Dit soort secties wordt direct opgenomen op de plaats waar `<section id="nnnn" data-include-format="html" data-include="filenaam.html"></section>` is geplaatst.

Het html fragment in het bestand hoeft niet te bestaan uit 1 root element. Sterker nog als dat wel het geval is en het fragment heeft de root 'div' of 'sectie' dan wordt het fragment niet vertaalt naar een separaat hoofdstuk.

Om een separaat hoofdstuk te kunnen starten dient het document wel met een 'hx' element te starten (h1, h2, h3, etc..).

De titel wordt dan ook opgenomen in de TOC.

Dit soort secties mag ook zonder 'id' attribuut worden opgenomen. Die variant geeft geen ander resultaat dan die hiervoor geschatst. Alleen wordt bij deze variant het id van de sectie en de gerelateerde href in de TOC gegenereerd op basis van de titel van deze sectie.

`data-include-format="html"` mag worden weggelaten.

3.2.2.10 Andersoortige secties

Indien een sectie element leeg is en het 'id' komt niet overeen met een van de, in de voorgaande paragrafen beschreven, bekende id's dan wordt de sectie genegeerd.

3.2.3 Bijlage N Referenties

Wordt alleen opgenomen als er in een van de andere documenten (zowel markdown als html) een referentie is opgenomen in de vorm '[Ref]' en die referentie in config.js of organisation-config.js is gedefinieerd.

3.2.4 Images in de documentatie

Plaats eventuele images die je in de Respec documentatie wil opnemen in de 'media' folder. Daarbinnen mag je elke door jou gewenste folderstructuur creëren.

3.3 Lokale Respec configuratie properties

Zoals aangegeven maken we in het Respec framework gebruik van een aantal VNG-R properties. Properties die er voor zorgen dat alle Respec documentatie van VNG-R eenzelfde look en feel heeft. Er zijn echter ook een aantal lokale configuratie properties waarmee voor ieder Respec document eigen keuzes kunnen worden gemaakt. Denk daarbij aan de status die het document heeft, de publicatie datum, de editors, etc...

Alle lokale configuratie properties kun je vinden in 'js/config.js' en mag je naar eigen inzicht aanpassen.

Er moet nog bepaald worden welke properties lokaal moeten zijn en welke globaal (dus welke behoren te staan in de repository 'Respec-Organization-configurations').

3.4 Functie Respec configuratie properties

Hieronder vind je de totale lijst van Configuratie properties. De vierde kolom geeft aan of het om een globale of lokale property gaat. Voor enkele properties is dat heel logisch, zo zijn 'localizationStrings' en 'previousPublishVersion' logischerwijs globaal, 'github' en 'title' zijn juist weer lokaal. De meeste globaal gedefinieerd properties kunnen lokaal overruled worden zoals 'useLogo'. Doe dat echter alleen als daar een hele goede reden voor is.

Property	Type	Afspraak gebruik binnen VNG-R (Globaal/Lokaal)	Vaste globale waarde of default waarde	Beschrijving
addSectionLinks	boolean	Globaal en lokaal	true	Bepaald of er een paragraafteksten (§), me link naar de paragraaf teken vóór komt te sta gegenereerd of niet. Biedt anderen de gelegenheid links naar specifieke paragrafen in je Respec document te kopiëren en te gebruiken. Er is een gekozen standaard altijd mee te genereren.

Property	Type	Afspraak gebruik binnen VNG-R (Globaal/Lokaal)	Vaste globale waarde of default waarde	Beschrijving
<u>alternateFormats</u>		Array met per formaat de properties 'label' en 'url'.	Lokaal	Hiermee kun je aangeven de Respec documentatie een ander formaat dan aanbiedt, op dit moment alleen pdf mogelijk. Deze configuratie prop zorgt er voor dat er eer bestand wordt gegenereerd dat er in de Respec documentatie een zin gekozen wordt aan het pdf formaat, daarin de link naar het bestand.
<u>authors</u>		Array met per naam de properties 'name', 'company' en 'companyURL'.	Lokaal	Bevat 1 of meerdere beschrijvingen van personen die hebben bijgedragen tot stand koming van het Respec document.
<u>content</u>		Array (zie een beschrijving onder deze tabel).	Lokaal	Het heeft de voorkeur om te gebruiken boven authoren. Indien deze configuratie property niet aanwezig is, 'Auteurs' niet getoond. Te gebruiken voor het toevoegen van content in het Respec document. Het voorkeur [de 'Sectie' in (./#sectie-methode) te gebruiken.
<u>editors</u>		Array met per naam de properties 'name', 'company' en 'companyURL'.	Lokaal	Één of meerdere beschrijvingen van personen die hebben bijgedragen aan de tot stand koming van het Respec document.
<u>formerEditors</u>		Array met per naam de properties 'name', 'company' en 'companyURL'.	Lokaal	Het heeft de voorkeur om te gebruiken boven authoren. Indien deze configuratie property niet aanwezig is, 'Redacteurs' getoond zonder vulling.
<u>github</u>		URI of een array van de properties 'repoURL' en 'branch'.	Lokaal	Bevat 1 of meerdere beschrijvingen van personen die in het verleden hebben bijgedragen aan de totstandkoming van het Respec document.
				Gebruikt voor het genereren van de links in de 'Doe tabel' bovenin de Respec documentatie. Kan gevuld worden met: <ul style="list-style-type: none"> • een url naar een GitHub repository • het deel van de url van een GitHub repository

Property	Type	Afspraak gebruik binnen VNG-R (Globaal/Lokaal)	Vaste globale waarde of default waarde	Beschrijving
<u>labelColor</u>	Hexadecimale colorcode.	Globaal		komt na 'https://github.com/ • een set van property bestaande uit <ul style="list-style-type: none"> ◦ repoURL: Een bovenstaand URL. ◦ branch: de branch waarin het R document moet issues staan opgeslagen. Verwijst naar de GitHub repository waarin het Informatiemodel wordt beheerd.
<u>latestVersion</u>	Combinatie van strings en configuration propertynamen.	Globaal en lokaal	Definieert de url van de laatst gepubliceerde versie. Samenvoeging van achtereenvolgens nl_organisationPublishURL, pubDomain, "/", en shortName.	Indien niet gedefinieerd wordt de 'Doe mee' tab gegenererd.
				Definieert de bij de in 'LocalizationStrings' gedefinieerde statusser horende kleuren.
				Wordt opgebouwd m.b andere gedefinieerde configuration propertie tekens. Daarin voorkomt hoofdletters worden om naar kleine letters.

Property	Type	Afspraak gebruik binnen VNG-R (Globaal/Lokaal)	Vaste globale waarde of default waarde	Beschrijving
<u>license</u>	enumeration	Globaal en lokaal	eupl	Definieert het licentiet van toepassing is op het document. VNG-R har 'EUPL' licentie maar z gewenst kan ook geko worden voor 'CC0', 'C 'CC-BY-ND'. Toegesta waardes 'eupl', 'cc0', 'c 'cc-by-nd'. Wordt gebru licentie-logo en bijbeh link in het document te genereren.
<u>licenses</u>	properties	Array met per licentiecode de properties	Globaal en lokaal 'name', 'short', 'url' en 'image'.	Definieert middels een van configuratie prope ('name', 'short', 'url' en de te gebruiken soort licenties waarnaar mid code kan worden verw de configuratie-optie '1 Hiermee kan een lijst r referenties in het hoofd 'Referenties' worden gegenereerd. Die refer bevatten metainformat 'auteur', 'publicatiедatum' status') en links naar d betreffende externe ref De referenties worden alleen opgenomen in d hoofdstuk als er in het document naar verwez middels een link in de syntax [[Referentienaam]] syntax geldt voor zowals markdown docume
<u>localBiblio</u>	array	Array van één of meerdere objecten met de properties 'href', 'title', 'publisher', 'date' en 'rawDate'.	Globaal en lokaal	Indien een link wordt opgenomen in een nor documentdeel zal de re terecht komen in de subparagraaf 'Normatireferenties'. Is deze op in een informatief documentdeel dan kon de subparagraaf 'Infor referenties' terecht.
				Gerefereerd kan word specrefs die beschikba de SpecRef database (https://github.com/tobi) of aan zelf in deze proj gedefinieerde referenti syntax voor de inhoud localBiblio property is

Property	Type	Afspraak gebruik binnen VNG-R (Globaal/Lokaal)	Vaste globale waarde of default waarde	Beschrijving
				beschreven.
				Neem waar van toepas verwijzingen op naar gerelateerde wetgeving gerelateerde standaard 'Pas toe leg uit'-lijst va Forum Standaardisatie Gemeentelijke standaa met verbindendheid 'pas-of-leg-uit' en 'verplicht
<u>localizationStrings</u>		Array van properties per taalcode	Globaal en lokaal	Bevat voor een aantal ('document statussen' en 'document types') / taalcombinaties de te gebruiken codes en de daarbij horende tekst.
<u>logos</u>		Array per logo van de properties 'src', 'alt', 'id', 'height' en 'url'.	Globaal en lokaal VNG Realisatie logo	Definieert de src, alternatieve tekst, url en grootte van linksboven in het Response document te plaatsen logo.
<u>maxTocLevel</u>	Integer	Globaal en lokaal		<p>Is van invloed op twee property's:</p> <ul style="list-style-type: none"> • het aantal niveau's dat maximaal wordt opgenomen in de inhoudsopgave van Respec document • het aantal paragraafniveau's in de inhoud van het document. Wordt benummerd altijd 1 niveau meer dan de waarde van de property.
<u>nl_organisationName</u>	String	Globaal en lokaal	VNG Realisatie	Wordt gebruikt om de en het vertikale label linksboven te genereren.

Property	Type	Afspraak gebruik binnen VNG-R (Globaal/Lokaal)	Vaste globale waarde of default waarde	Beschrijving
nl_organisationPublishURL	URL	Globaal en lokaal	https://johanboer.github.io/Handreiking	Wordt gebruikt voor het genereren van de link naar GitHub pages van de huidige en de laatste gepubliceerde versie. Het leidt naar een documentatie GitHub Pages interface 'publicatie' GitHub repository en zo gewerkt de 'publicatie' repository gedefinieerde custom cname.
nl_organisationStylesURL	URL	Globaal en lokaal	https://gitdocumentatie.logius.nl/publicatie/respec/style/	Kan worden gebruikt in properties 'lastVersion' en 'thisVersion' en 'prevVersion'. Definieert de locatie waar te gebruiken CSS bestanden exclusief dat bestand zelf. Bepaald of er links van inhoud een frame met de inhoudsopgave gegeneerd wordt.
noTOC	boolean	Lokaal	false	Indicateert of er een inhoudsopgave in de inhoudsopgave gegeneerd wordt.
otherLinks	Array van properties	Lokaal		Genereert een of meer secties (afhankelijk van aantal 'key' 'data' voorkeuren) in de header van het Respec document met als titel de waarde van de property als inhoud een of meer links.
postProcess	Functie aanroep.	Globaal	?	Bevat een of meer Java functies die achtereen opgestart worden nadat de documentatie klaar is met generatie van het Respec document.
previousMaturity	enumeration	Lokaal		Status van de voorgaande 'publicatie' repository gepubliceerde versie.
previousPublishDate	Datum in het formaat YYYY-MM-DD	Lokaal		Publicatiedatum van de voorgaande versie.
previousPublishVersion	SemVer notatie	Lokaal		Versienummer van de voorgaande versie in SemVer notatie (https://semver.org/lang)
prevVersion	Combinatie van strings en	Globaal en lokaal	Samenvoeging van achtereenvolgens nl_organisationPublishURL, pubDomain, "/"	Wordt gebruikt in de property 'prevVersion'. Wordt opgebouwd m.b.v. andere gedefinieerde

Property	Type	Afspraak	Beschrijving
		gebruik binnen VNG-R (Globaal/Lokaal)	Vaste globale waarde of default waarde
	configuration propertynamen.	shortName, "/" en previousPublishVersion.	configuration propertie tekens. Daarin voorkomt hoofdletters worden over naar kleine letters.
<u>pubDomain</u>	enumeration	Globaal en lokaal	Definieert het publicatie domein van het Respecut document en heeft op een moment de waarde 'cir'
			Wordt nu gebruikt in de properties 'lastVersion' 'thisVersion' en 'prevVer'

Property	Type	Afspraak gebruik binnen VNG-R (Globaal/Lokaal)	Vaste globale waarde of default waarde	Beschrijving
<u>publishDate</u>	Datum in het formaat YYYY-MM-DD	Lokaal		Publicatiедatum van de versie.
<u>publishVersion</u>	SemVer notatie	Lokaal		Kan evt. worden gebruikt als property 'thisVersion'. Versienummer van de] versie in SemVer notatie (https://semver.org/lang)
<u>shortName</u>	String	Lokaal		Wordt gebruikt in de property 'thisVersion'. Korte naam (bijv. een mnemonic) van het Re document.
<u>sotdText</u>	enumeration	Array van properties per taalcode.	Globaal en lokaal	Bevat voor een aantal 'specStatus'sen en taler gebruiken codes en de horende volledige teks
<u>specStatus</u>	enumeration	Lokaal		Definieert de status van Respec document. De gebruikte statussen zijn gedefinieerd in de gloeilicht configuratie property 'localizationStrings'. Onderstaand moment zijn dat:
				<ul style="list-style-type: none"> • cv: Consultatieve versie • vv: Versie ter vaststelling • ig: In Gebruik ver • io: In Ontwikkeling ver
				Wordt gebruikt om de verticale en het vertikale label linksboven te genereren. Bepaald ook de kleur van het label. Dit dient in de gloeilicht configuratie gedefinieerd te worden.
				De kleuren voor de VNV statussen kunnen worden gedefinieerd in de gloeilicht.

Property	Type	Afspraak		Beschrijving
		gebruik binnen VNG-R	Vaste globale waarde of default waarde (Globaal/Lokaal)	
				'labelColor'.
				Kan ook worden gebruikt om de properties 'latestVersion', 'thisVersion' en 'prevVersion' te vervangen.
				Definieert het type van Respec document. De mogelijke waarden zijn 'spec', 'im', 'hl' en 'pr'. De mogelijke types zijn 'globaal' en 'lokaal'. De configuratie property 'localizationStrings' kan worden gebruikt om de verticale labels te configureren.
<u>specType</u>	enumeration	Lokaal		<ul style="list-style-type: none"> • im: Informatiemodel • hl: Handleiding • pr: Praktijkrichtlijn
				Wordt gebruikt om de verticale labels linksboven te genereren. De template heeft dit de waarde 'IM' aangezien we bijvoorbeeld een Respec veelal zullen gebruiken voor Informatiemodelleerpubliceren.
<u>subtitle</u>	String	Lokaal	n.v.t.	Kan evt. ook worden gebruikt in de properties 'latestVersion', 'thisVersion' en 'prevVersion'. Bevat een string die als subtitle van de titel van het document wordt geplaatst boven de gegenererde 2e subtitle. De organisatienaam, documenttype, specStatus en versiedatum worden gebruikt.
<u>testSuiteURI</u>	URL	Lokaal	n.v.t.	Dit is een optionele configuratie property. Genereert een sectie in header van het Respec document met als titel 'testsuite' en als inhoud een URL naar een testsuite. Welke URL moet worden gebruikt voor het API Testplatform maar alle andere Respec ook gaan gebruiken voor de API's.
<u>thisVersion</u>	Combinatie van strings en configuration propertynamen.	Globaal en lokaal	Samenvoeging van achtereenvolgens nl_organisationPublishURL, pubDomain, "/", shortName, "/" en publishVersion.	Wordt opgebouwd m.b.v. andere gedefinieerde configuration propertynamen. Daarin voorkomen hoofdletters worden overgeschreven door kleine letters.

Property	Type	Afspraak	gebruik binnen VNG-R (Globaal/Lokaal)	Beschrijving
title	Lokaal	De titel van het Respec document.		
<u>useLabel</u>	boolean	Globaal en lokaal	true	Bepaald of het vertical aan de linker bovenzijc inhoudsopgave gegene moet worden.
<u>useLogo</u>	boolean	Globaal en lokaal	true	Deze property kan lokaal overruled worden. Bepaald of het VNG-R logo in de rechter bovenzijc van het document geplaatst moet worden.
<u>edDraftURI</u>	URL	Globaal en lokaal		Beschrijft de url waarop van het Respec document worden bekeken (de laagste werkversie).

4. Conformiteit

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

A. Index

A.1 Begrippen gedefinieerd door deze specificatie

A.2 Begrippen gedefinieerd door verwijzing

B. Referenties

B.1 Normatieve referenties

[domain-driven-design]

Reference not found.

[domain-driven-design#the-blue-book]

Reference not found.

↑