Business rules van STOP, BHKV en LVBB

Laatste update: 14 Jul 2020

Inleiding

In de ontwikkeling van STOP1.0 is de uitwerking van business rules die niet in XSDs kunnen worden uitgedrukt "organisch gegroeid". De beslissing om IMTP te schrappen nam niet de noodzaak weg van business rules-validatie. Business rules worden nu handmatig bijgehouden op de volgende plaatsen

- STOP1.0 kent eén schematron: imop-tekst.sch die enkele van de mogelijke business rules van de tekst-modules van STOP bevat.
- Daarnaast heeft PR30 in het BHKV een serie noodzakelijke validaties geïmplementeerd in schematron en XQuery om de LVBB correct te laten functioneren.

De huidige validatiematrix (zie bijlages als subpagina van Validaties) is tot stand gekomen dmv reverse engineering van deze bestaande validaties die in de loop van de tijd aan het project zijn toegevoegd. De regels in deze matrix zijn meestal vanuit de technische eisen beschreven.

Er is een noodzaak om de volgende punten op te pakken in de doorontwikkeling van STOP:

1. correcte identificatie en scherpe definitie van de business rules aanvullend op de imop-XSDs,
2. een keuze voor implementatie van de business rule
3. consistentie van business rules met de modulaire opzet van STOP en de STOP-verwerkende APIs (bronhouderkoppelvlak)
4. toepassing van business rules op STOP-modules en LVBB-aanleveringen
5. correcte documentatie van de business rules die onderdeel zijn van een uitlevering van STOP en API-schemas
6. testen van de specificatie van de business rules in STOP- en API-uitleveringen
7. consistente afhandeling van foutmeldingen uit validaties

Context vooraf

- Voor nu is validatie alleen relevant voor aangeleverde content.
- gebruikers van schematrons: PR34, PR33, LVBB, (softwareleveranciers van) BGs

Business rules

Zowel STOP als het bronhouderkoppelvlak onderkennen business rules waaraan uitgewisselde informatie moet voldoen. Als aan een business rule niet voldaan wordt, dan moet het resultaat een melding zijn waaruit af te leiden is wat er mis is met de uitgewisselde informatie. Omdat een deel van de meldingen door STOP gedefinieerd wordt, is het voorstel om de definitie van een melding (informatiemodel van een melding) ook in STOP op te nemen. Deze reeds lang bestaande wens is in het verleden geïmplementeerd middels een semi-STOP namespace voor meldingen in de lvbb-transport.xsd

Business rules zijn onafhankelijk van implementatie

De reverse engineering die toegepast moest worden om de validatiematrix op te stellen kan beter omgedraaid worden:

1. Stel eerst sluitende business rules (BRs) op. Maak goede afspraken over consistent taalgebruik
2. Specificeer deze op zo'n manier dat implementatie makkelijk is, met de daarvoor geschikte technologie op de daarvoor beste plek (STOP-schematron, BHKV-schematron, LVBB-xquery bijv).
3. Gebruik de specificatie van de BRs tevens voor documentatie
    a. index van alle business rules
    b. koppeling met bestaande documentatie
    c. koppeling met de validatiematrix
4. Gebruik BRs ook voor afstemming binnen het project (de "validatatiematrix" in Excel): Een lijst met alle meldingscodes en een verwijzing naar de documentatie van de business rule

XSD-validiteit is één "business rule" (en niet meer)

Het informatiemodel zelf definieert ook "rules" maar dat zijn geen "business rules". STOP kent een informatiemodel (in UML) dat in XML-schema is geïmplementeerd. De structuur van het informatiemodel impliceert technische business rules, zoals cardinaliteit, het al dan niet voorkomen van bepaalde elementen als child van een ander element of uniciteit, Deze regels zijn zoveel mogelijk in een schema opgenomen, en worden pas een business rule als het niet anders kan.

Het feit dat content niet aan het informatiemodel voldoet (lees: niet valide is tov de STOP-schema's), kan zeer uiteenlopende oorzaken hebben. Toch wordt deze "niet-validiteit" gemeld met één en dezelfde foutmelding met dezelfde code (die staat voor: voldoet niet aan informatiemodel). De beschrijving van de fouten in de schema-validatie kunnen applicatie-specifiek zijn. Deze komen voort uit XML-parsers; op de foutmeldingen die zo'n parser genereert heeft de implementerende ontwikkelaar weinig invloed.

Business rules die niet uit het informatiemodel komen worden op een standaard manier opgeschreven

Een beschrijving van een business rule bevat tenminste:

- Een **code** die identificerend is voor de business rule, waarmee snel in de documentatie een beschrijving opgezocht kan worden van de oorzaak van de melding en aanwijzingen gevonden kunnen worden hoe de informatie aangepast moet worden zodat wel aan de business rule voldaan wordt. Aan de code moet te zien zijn waar de business rule onderdeel van uitmaakt (van STOP, de API, TPOD, etc).
- De **ernst** (fout, waarschuwing of informatief):
    - fout = informatie kan niet geaccepteerd worden;
    - waarschuwing = de informatie kan gebruikt worden door het ontvangende systeem (of achterliggende systemen) voor de kerntaken, maar niet alle functionaliteit is beschikbaar die wel beschikbaar zou zijn als de informatie wel zou voldoen aan de business rule;
    - informatief = wat het ontvangende systeem nog even kwijt wil. [Dit betekent dat STOP alleen fouten als uitkomst kent want STOP weet niets van functionaliteit, maar dat het bronhouderkoppelvlak wel waarschuwingen kent. Misschien zou het STOP-model van meldingen alleen fout en waarschuwing moeten onderkennen en een manier om als API daar extra levels aan toe te kennen, bijv trace en debug.]
- De **business rule** zelf, opgeschreven in een simpele, consistente taal. RuleSpeak (http://www.rulespeak.com/nl/) biedt daarvoor handvatten. Zie ook de bijlagen
    - RuleSpeak Zinsstructuren v2-NL5.pdf
    - Basis RuleSpeak Dos en Donts v2-NL5.pdf
- De **melding** die optreedt als er niet aan de business rule wordt voldaan. Deze beschrijving geeft extra informatie over de reden dat niet aan de business rule voldaan is. De melding bevat ook identificerende variabelen die meer informatie geven over de locatie van de fout. In het ideale geval geeft de melding ook een remedie, een hint hoe de fout op te lossen. Doelgroepen van de melding is de ontwikkelaar/beheerder van de software die de informatie aanlevert.
- Een of meer verwijzingen naar onderliggende **documentatie** en specificaties. Deze documentatie is publiek toegankelijk.
    - Als er aan een business rule géén documentatie gelinkt kan worden, is dit een waarschuwing voor de schrijver van business rule en/of van de documentatie: is de BR wel goed of de documentatie wel compleet?

De beschrijving van de BRs vanuit STOP, BHKV en LVBB moet in de overgangsfase waarin de validatiematrix nog gebruikt wordt, voldoende de informatie bevatten om de de "oude regel" te kunnen herleiden. Dit kan op basis van de huidige regelcodes.

De lvbb-transport.xsd (https://standaarden.overheid.nl/lvbb/1.0.3/lvbb-transport.xsd) importeert een schema (https://standaarden.overheid.nl/lvbb/1.0.3/stop_toevoeging.xsd) dat de STOP-melding modelleert. Deze STOP-melding bevat meer informatie dan bovengenoemd voorstel. Op het moment van instellen werd er als eis werd gesteld dat meldingen ook voor eindgebruikers begrijpbaar moeten zijn. Dat leidde tot een tweedeling in technisch (niet voor eindgebruikers) en functioneel (potentieel wel voor eindgebruikers) en een mechanisme om de beschrijving in een applicatie te kunnen vertalen naar een app-specifieke melding. Deze eis kwam voort uit de manier waarop nu met RP.nl wordt omgegaan: met de hand aanleveren, eindgebruikers die ook echt meldingen zien. Dit kan vereenvoudigd worden als de meldingen alleen voor devops zijn, die dan zelf maar moeten beslissen hoe ze dat aan een eindgebruiker melden - de normale manier van werken bij een koppelvlak.

De stop-uitbreiding van de LVBB-schema's voor meldingen blijft voor nu ongewijzigd gehandhaafd "onderdeel" van de lvbb-transport.xsd

Business rules zijn van toepassing op documenttypes

Business rules worden toegepast op XML-documenten. STOP en LVBB kennen verschillende documenttypes (STOP-modules of LVBB-aanleveringen) waarop de business rules geldig zijn. Deze koppeling van een specifieke BR aan een specifiek type instance moet ook worden vastgelegd.

Deze BR-toekenning kan ten eerste worden gebruikt door STOP-implementatoren (zoals leveranciers) om te kunnen bepalen welke BRs op een specifiek documenttype geïmplementeerd moet worden. Ook wordt hiermee een ontkoppeling bereikt van de BRs enerzijds met de fysieke bestanden waarin deze worden opgeschreven anderzijds. Met andere woorden: de schrijvers van BRs zijn op deze manier vrij om de regels over één of meer bestanden te verdelen; ook de ontwikkelaar van de implementatie van de regels in bijvoorbeeld schematron is vrij om naar eigen inzicht de regels te verdelen over één of meerdere schematron-instances.

Koppelen BRs schematron aan instances beschrijft de methodiek om business regels (die bijvoorbeeld in schematron-bestanden zijn uitgedrukt) toe te passen op XML-instances (die volgens STOP zijn gecodeerd).

Opbouw van BRs

Bovenstaande leidt tot een "registratieformaat voor BRs". Een voorbeeld werkt verhelderend.

Eén business rule

voorbeeld business rule

```
<BusinessRule>
  <!-- de code van de BR, met afgesproken prefix (STOP, BHKV of LVBB) -->
  <code>STOP0001</code>

  <!-- "ernst" heeft 3 waarden: fout, waarschuwing, info -->
  <ernst>fout</ernst>

  <!-- de business rule, opgeschreven met een heldere en consistente taal, zoals RuleSpeak -->
  <regel>Een Lijst van het type 'ongemarkeerd' MAG GEEN lijst-items met nummering of symbolen
      bevatten.</regel>

  <!-- melding bevat de melding met afgesproken variabelen -->
  <melding>De Lijst met eId <var naam="eId"/> van type 'ongemarkeerd' heeft LiNummer-elementen met een
    nummering of symbolen, dit is niet toegestaan. Pas het type van de lijst aan of verwijder de
    LiNummer-elementen.</melding>

  <!-- link naar relevante documentatie -->
  <documentatie>
    <link href="xsd:tekst:Lijst">documentatie van Lijst</link>
    <link href="xsd:tekst:Li"/>
    <link href="xsd:tekst:LiNummer"/>
  </documentatie>
</BusinessRule>
```

Variabelen in de melding

Variabelen worden opgeschreven als `<var naam="naam-van-var">`.

`@naam` voldoet aan de regexp `[a-Z0-9-_]+` . Dat betekent dat er geen spaties, punten of andere karakters zijn toegestaan.

Een eerste inventarisatie welke waarden `naam-van-var` er nodig zijn:

naam	betekenis
bestandsnaam	de bestandsnaam
ID	generieke identiifier. alleen te gebruiken als het voor de BR niet van belang is of welke **soort** identifier beschreven is. Als dat wel bekend of van belang is, verdient het de voorkeur een van de specifieke types te gebruiken
Work-ID	de AKN- of JOIN-identificatie van het Work van de STOP-module
Expression-ID	de AKN- of JOIN-identificatie van het Expression van de STOP-module
soortWork	de waarde van data:soortWork van de module
rootelement	de Qnaam van het root-element van de module
componentnaam	de naam van de AKN-component waarin het element voorkomt
element	QNaam van het context-element
ouder	QNaam van het ouder-element van het context-element
ref	de waarde van het attribuut ref van het context-element
wId	de waarde van het attribuut wId van het context-element
eId	de waarde van het attribuut eId van het context-element
wat	de waarde van het attribuut wat van het context-element

Als er ook een JSON-formaat voor het vastleggen van de BRs wordt voorzien, moet een syntax daarvoor worden afgesproken voor dit element, bijvoorbeeld `{$naam-van-variabele}`. In JSON kan geen mixed content gecodeerd worden.

De toepasbaarheid van een (set) business rule(s)

Een of meer business rules worden gegroepeerd in een BusinessRuleGroep. Per groep wordt de set van documenttypes vastgelegd waarvoor de business rules gelden:

BusinessRuleGroep

```
<BusinessRuleGroep>
  <geldtVoor>
   <!-- De documenttype(s) waarvoor deze BR geldt en waarop de BR moet worden toegepast
        De documenttypes worden gespecificeerd middels de naam van het root-element en de namespace daarvan -->
    <Documenttype>
      <localName>BesluitCompact</localName>
      <namespace>https://standaarden.overheid.nl/stop/imop/tekst/</namespace>
    </Documenttype>
    <Documenttype> ...  </Documenttype>
    ...
  </geldtVoor>

  <geldendeBusinessRules>
    <!-- alle business rules die gelden voor deze documenttypes -->
    <BusinessRule>
      <code>STOP0001</code>
      <ernst>fout</ernst>
      <regel>Een Lijst van het type 'ongemarkeerd' MAG GEEN lijst-items met nummering of symbolen
    hebben</regel>
      <melding>De Lijst met eId <var naam="eId" /> van type 'ongemarkeerd' heeft
        LiNummer-elementen met een nummering of symbolen, dit is niet toegestaan. Pas het type van
        de lijst aan of verwijder de LiNummer-elementen.</melding>
      <documentatie>
        <link href="stop#xsd:tekst:Lijst">tekst:Lijst</link>
        <link href="stop#xsd:tekst:Li">tekst:Li (lijst-item)</link>
        <link href="stop#xsd:tekst:LiNummer">tekst:Linummer (lijstnummering)</link>
      </documentatie>
    </BusinessRule>
    <BusinessRule>...</BusinessRule>
    <BusinessRule>...</BusinessRule>
  </geldendeBusinessRules>
<BusinessRuleGroep>
```

Een business rules-bestand kan meerdere BusinessRuleGroeps bevatten:

volledige XML-instance

```
<BusinessRules schemaversie="1.1"
  xmlns="https://standaarden.overheid.nl/stop/imop/businessrules/">

  <BusinessRuleGroep>..</BusinessRuleGroep>
  <BusinessRuleGroep>..</BusinessRuleGroep>
  <BusinessRuleGroep>..</BusinessRuleGroep>

</BusinessRules>
```

Aandachtspunten

- Het schema voor deze wordt onderdeel van STOP en STOP-uitleveringen
- Schematron kent zelf een standaard voor meldingen: "Simple Validation Reporting Language". Van SVRL zijn niet veel implementaties bekend. Het wordt derhalve niet gebruikt.
- De huidige stop-meldingen-schema kent "soort" en "metadata". Deze zijn niet nodig; deze informatie zegt alleen iets over de toepassing van de regel, niet over de regel zelf. Deze informatie kan gegenereerd in de melding.

Gebruik van business rules-formaat voor Schematron

De hierboven gespecificeerde business rules moeten worden geïmplementeerd. Voor validatie van STOP-modules en aan te leveren XML-instances is Schematron de beste keuze. Hierbij gelden de volgende regels

- De business rules zoals die in het registratieformaat zijn opgeschreven zijn leidend. Deze regels worden beschikbaar gesteld bij STOP.
- Schematrons worden geschreven / ontwikkeld door interpretatie van de business regels. Er mist teveel informatie om de Schematrons automatisch te genereren
- Redundantie van regels en meldingen moet zo veel mogelijk voorkomen worden; de "single source of truth" is vastgelegd in het registratieformaat voor business rules. Meldingen en regels mogen hooguit automatisch worden geïncludeerd in Schematrons, bijvoorbeeld voor documentatiedoeleinden.
- De Schematrons voor STOP-modules en LVBB-aanleveringen worden uitgeleverd aan eindgebruikers. Deze Schematrons kunnen zinvol worden toegepast voordat content aan het BHKV wordt aangeboden
    - LVBB implementeert dezelfde schematrons als worden uitgeleverd
    - STOP levert een specificatie welke schematrons op welke bestanden van toepassing zijn: Koppelen BRs schematron aan instances
- Schematrons zijn geschikt voor de bekende "reference skeleton implementation" van ISO-Schematron (http://schematron.com/front-page/the-schematron-skeleton-implementation/)

Dit betekent dat STOP-Schematrons op een specifieke manier worden opgeschreven.

Alweer, een voorbeeld is het beste startpunt. De <BusinessRule> over lijstitem wordt als volgt in Schematron genoteerd:

```
<sch:pattern id="sch_tekst_001" see="tekst:Lijst tekst:Li tekst:LiNummer">
  <sch:title>Lijst - Nummering lijstitems</sch:title>
  <sch:rule context="tekst:Lijst[@type = 'ongemarkeerd']">
    <sch:assert id="STOP0001" role="error" test="count(tekst:Li/tekst:LiNummer) = 0">
     { "code" :  "STOP0001",
       "eId": "<sch:value-of select="@eId" />"}</sch:assert>
  </sch:rule>
</sch:pattern>
```

Deze schematron-assert geeft bij validatie van een niet-valide XML-instance een JSON-snippet terug:

```
{ "code": "STOP0001",
  "eId": "list_o_1" }
```

In principe is die voldoende informatie om de business rule op te zoeken in het gepubliceerde overzicht van BRs.

Het is wel een stuk gebruikersvriendelijker om de **regel** en de **melding** ook op te nemen in de Schematron. Deze regel en melding kunnen relatief makkelijk in een uit te leveren Schematron worden toegevoegd, op basis van de code van de BR:

```
<sch:pattern id="sch_tekst_001" see="tekst:Lijst tekst:Li tekst:LiNummer">
  <sch:title>Lijst - Nummering lijstitems</sch:title>
  <sch:rule context="tekst:Lijst[@type = 'ongemarkeerd']">
    <sch:assert id="STOP0001" role="error" test="count(tekst:Li/tekst:LiNummer) = 0">
     { "code" :  "STOP0001",
       "eId": "<sch:value-of select="@eId" />",
       "regel" : "Een Lijst van het type 'ongemarkeerd' MAG GEEN lijst-items met nummering of symbolen hebben",
       "melding": "De Lijst met eId <sch:value-of select="@eId" /> van type 'ongemarkeerd' heeft LiNummer-
elementen
        met een nummering of symbolen, dit is niet toegestaan. Pas het type van de lijst aan of verwijder de
        LiNummer-elementen." } </sch:assert>
  </sch:rule>
</sch:pattern>
```

Als deze regel "afvuurt", bevat de output ook de mensleesbare regel en melding:

```
{ "code": "STOP0001",
  "eId": "list_o_1",
  "regel": "Een Lijst van het type 'ongemarkeerd' MAG GEEN lijst-items met nummering of symbolen hebben",
  "melding": "De Lijst met eId list_o_1 van type 'ongemarkeerd' heeft LiNummer-elementen met een nummering
    of symbolen, dit is niet toegestaan. Pas het type van de lijst aan of verwijder de LiNummer-elementen." }
```

De verwachting is dat deze JSON-snippet makkelijk te verwerken is door validators.

JSON ipv XML

Achtergrond: waarom wordt nu ineens JSON als outputformaat voorgesteld, terwijl de rest van STOP in XML is geïmplementeerd?

Belangrijkste reden ligt in de beperkingen van de eerdergenoemde reference implementation (in XSLT) van Schematron. Deze kan binnen custom XML-elementen geen variabelen "resolven", zoals `<sch:value-of select="@eId" />` . Een XML-snippet binnen de assert wordt één-op-één doorgevoerd naar de output. Dit zou voor het voorbeeld hierboven betekenen dat bijvoorbeeld de `sch:value-of` niet wordt "ingevuld met de waarde van @eId" maar als literal xml-fragment wordt overgenomen in de output. Dan wordt de melding

```
De Lijst met eId <sch:value-of select="@eId" /> van type 'ongemarkeerd' heeft LiNummer-elementen met een nummering of symbolen
```

Dat helpt de gebruiker nog niet.

Gedetailleerde specs

Koppelen schematrons aan instances: Koppelen BRs schematron aan instances

Opzet schematrons in meer detail: Opzet schematron en documentatie

Koppelen BRs schematron aan instances

Laatste update: 08 Jul 2020

STOP kent een beperkt aantal module-types. De LVBB kent ook aan te leveren documenttypes. Deze module-types worden geserialiseerd in XML in een beperkt aantal "root-elementen", deze zijn beschreven in de snelstartgids van STOP.

Aangenomen wordt dat alleen zinvol is om de door BGs aangeleverde XML-bestanden te valideren. Door LVBB gegenereerde instances in de uitlevering naar bijvoorbeeld OZON hoeven niet gevalideerd te worden.

Context en scope:

1. STOP-schematron beschrijven business rules binnen de module.
    a. hieronder valt nadrukkelijk ook de validatie van de opbouw van /akn/ en /join/ identificatiestrings zoals vastgelegd in https://koop.gitlab.io/STOP/standaard/naamgevingsconventie.html. De LVBB heeft alleen een AKN-resolver, die valideert niet. En de opbouw van de /akn en /join is vastgelegd in STOP
2. LVBB-schematrons beschrijven business rules binnen één van de aangeleverde XML-instances in de zip, dus met name de onderlinge afhankelijkheid tussen geleverde tekst-, geo-, data- en andere STOP-modules
3. LVBB voert consistentie-checks en andere validaties betreffende
    a. de bestanden van een aangeleverde zip
    b. reeds eerder aangeleverde informatie
4. TPOD-schematrons zijn geen onderdeel van STOP of het bronhouderkoppelvlak
    a. Maar als de BRs uit de TPODs op een machineleesbare manier worden genoteerd en er schematrons uit worden gemaakt, zouden deze ook opgenomen kunnen worden in dit "schemakoppelingsformaat"

Voorstel:

- PR34 maakt en onderhoud de STOP-schematrons
- PR34 en PR30 maken en beheren de LVBB-schematrons en andere validaties (XQuery) in overleg
- PR33 maakt en beheert TPOD-schematrons

Om maximale flexibiliteit in toepassen van schematrons op instances te realiseren

1. Schematrons worden geversioneerd op dezelfde manier als STOP- en LVBBA-XSDs. Bij elke oplevering van een nieuwe STOP of LVBBA wordt net als bij de XSDs elke schematron voorzien van een opgehoogd versienummer, ook al verandert er niets aan de business regels in die betreffende schematron.
2. STOP-schematrons: Een XML-instance van een STOP-module in de namespaces `tekst`:, `data:`, `geo:` en `gio:` en een specifieke waarde voor @schemaversie wordt geassocieerd met 0 of meer Schematrons binnen die STOP-versie.
3. LVBB-schematrons: Elk mogelijk root-element van een uitgewisseld XML-bestand in de LVBBA-namespace (`aanlevering`) en een specifieke waarde voor @schemaversie wordt geassocieerd met 0 of meer Schematrons
4. de STOP-schematrons worden zo opgeschreven dat ze ook toepasbaar zijn op XML-instances die aan een API worden aangeboden, zoals instances in de LVBBA-namespace.
5. De koppeling schematron - rootelement wordt vastgelegd in een of meer configuratiebestanden. Deze hebben een specifiek formaat voor STOP in JSON, XML (of beide). Dit bestand wordt uitgeleverd en moet dus ook beschreven worden

- Er wordt dus expliciet gekozen om géén gebruik te maken van xml-model-processing instructions zoals beschreven in https://www.w3.org/TR/xml-model/#the-xml-model-processing-instruction.
- NVDL is een ISO-standaard voor meta-schema's (zie bijvoorbeeld http://www.xfront.com/nvdl/#) maar er zijn niet veel implementaties en expressiemogelijkheden zijn te beperkt voor deze use case.

Voorstel voor configuratie-bestand

schemakoppeling XML

```
<?xml version="1.0" encoding="UTF-8"?>
<Schemata
 xmlns="https://standaarden.overheid.nl/stop/businessrules/" schemaversie="1.0.3">
  <!--
  deze XML is een onderdeel van de IMOP specificatie.
  bestand waarin alle namespaces worden gekoppeld aan XSDs en schematrons
  -->

  <Documenttype>
    <localName>BesluitCompact</localName>
    <namespace>https://standaarden.overheid.nl/stop/imop/tekst/</namespace>
    <implementatie>
      <Schema>
        <type>XML-schema</type>
        <locatie>https://standaarden.overheid.nl/stop/imop-tekst.xsd</locatie>
```

```
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst.sch</locatie>
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst-besluit.sch</locatie>
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst-regeling.sch</locatie>
    </Schema>
  </implementatie>
</Documenttype>

<Documenttype>
  <localName>RegelingCompact</localName>
  <namespace>https://standaarden.overheid.nl/stop/imop/tekst/</namespace>
  <implementatie>
    <Schema>
      <type>XML-schema</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst.xsd</locatie>
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst.sch</locatie>
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst-regeling.sch</locatie>
    </Schema>
  </implementatie>
</Documenttype>

<Documenttype>
  <localName>BesluitKlassiek</localName>
  <namespace>https://standaarden.overheid.nl/stop/imop/tekst/</namespace>
  <implementatie>
    <Schema>
      <type>XML-schema</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst.xsd</locatie>
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst.sch</locatie>
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-besluit.sch</locatie>
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-tekst-regeling.sch</locatie>
    </Schema>
  </implementatie>
</Documenttype>

<Documenttype>
  <localName>ConsolidatieInformatie</localName>
  <namespace>https://standaarden.overheid.nl/stop/imop/data/</namespace>
  <implementatie>
    <Schema>
      <type>XML-schema</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-data.xsd</locatie>
    </Schema>
    <Schema>
      <type>schematron</type>
      <locatie>https://standaarden.overheid.nl/stop/imop-data.sch</locatie>
    </Schema>
    <Schema>
```

```
        <type>schematron</type>
        <locatie>https://standaarden.overheid.nl/stop/imop-data-consolidatieinfo.sch</locatie>
      </Schema>
    </implementatie>
  </Documenttype>

</Schemata>
```

Opzet schematron en documentatie

VOORSTEL UITWERKING

Documentatie van Schematrons via DITA

Documentatie van de schematrons moet meelopen met de documentatie en uitlevering. De documentatie beschrijft

- Welke validaties zijn er en waarom bestaan ze (verwijzing naar andere delen van documentatie of schema waar de reden van de validatie staat)
- Als iemand een melding krijgt (op basis van meldingcode): waarom treedt die op, wat moet er gebeuren om de fout/waarschuwing op te lossen?

Omdat schematrons niet breed ondersteund worden, leveren STOP en het bronhouderkoppelvlak naast de schematrons ook een XSLT-implementatie van de schematrons als serviceproduct.

Opzet

- Validaties met schematron zijn alleen vooral van toepassing voor aanleveringen (BesluitKlassiek of BesluitCompact)
- De schematron is voor alle partijen (ketenpartners) bruikbaar
- Er is een schematron voor imop-tekst generiek (voor alle typen Besluit en Regeling te gebruiken)
- Er is een of meer schematron voor imop-data (mogelijk specifiek voor combinatie van modules
- Er is een of meer schematron voor imop-data (mogelijk specifiek voor combinatie van modules)
- Schematron moet xslt2 als querybinding declareren in de root van de schematron als `queryBinding="xslt2"`.

Mogelijke beperkingen in notatie

Gebleken is dat sommige omgevingen - zoals MarkLogic - niet alle xpath constructies ondersteunen. *Niet duidelijk of dit probleem zich ook in andere omgevingen voordoet*.

Mocht dit niet te verhelpen zijn moeten de volgende notaties worden vermeden:

1. Gebruik van de **descendant axis**

- `element1/descendant::element2` - *Dit moet dan genoteerd worden als:*
- `element1//element2`

2. **Groepering** van elementen als een OR constructie

- `(element1 | element2 | element3)/element4` - *Dit moet dan genoteerd worden als:*
- `element1/element4 | element2/element4 | element3/element4`

Opbouw validaties

Schematron validaties moeten op een consistente manier worden opgebouwd zodat:

- Voor iedere validatiegroep (`sch:pattern`) duidelijk is waar de validatie betrekking op heeft, kort beschreven in een `sch:title`
- Voor iedere controle duidelijk is beschreven welke business rule wordt gevalideerd in een `sch:p` direct voor een `sch:assert` of `sch:report`
- Voor iedere context van een `sch:rule` duidelijk is elementen of attributen worden geraakt, gedeclareerd in een attribuut `@context`

Deze tekstuele documentatie is van belang voor zowel onderhoud van de Schematron (voor de schrijver is duidelijk waar een instructie over gaat en welk doel het dient) als voor het genereren van documentatie-overzicht en de relatie met de xsd-documentatie.

Conventies schematron

Om dit te realiseren zijn een aantal conventies voor het schrijven van schematron van belang:

- Een `sch:pattern` heeft een uniek identificatie attribuut `@id` binnen de context van de schematron
    - *Dit attribuut wordt gebruikt om voor documentatie de verwijzing van XSD-schema naar de specifieke schematron regel te mogelijk te maken.*
- Een `sch:pattern` heeft een attribuut `@see` met een spatie gescheiden opsomming van alle elementen waarop de validatie betrekking heeft
    - *Dit attribuut wordt gebruikt om voor documentatie de verwijzing van de specifieke schematron regel naar XSD-schema documentatie mogelijk te maken.*
- Een `sch:pattern` heeft een `sch:title` element met daarin een *korte* omschrijving voor onderliggende regels.
    - *beginnend met een duidelijke naam aan de hand waarvan de documentatie kan worden gesorteerd.*
- Een `sch:rule` element heeft altijd een attribuut `@context`
- Een `sch:assert` of `sch:report` heeft altijd een attribuut `@id` aan de hand waarvan een referentie naar bv. de validatie matrix of andere messages kan worden gelegd.
    - *NB - Deze ID wordt als code meegeleverd in de schematron output en is gelijk aan de code vermeld in de 'validatie-matrix' indien van toepassing.*
- Een `sch:assert` of `sch:report` heeft altijd een attribuut `@role` met waarde '`error`' of '`warning`' waarmee de ernst van de melding wordt aangegen.

Voorbeeld schematron opbouw

Lijst - Nummering lijstitems

```
<sch:pattern
  id="sch_tekst_001"
  see="tekst:Lijst tekst:Li tekst:LiNummer">
  <sch:title>Lijst - Nummering lijstitems</sch:title>
  <sch:rule
    context="tekst:Lijst[@type = 'ongemarkeerd']">
    <sch:assert
       id="STOP0001"
       role="error"
       test="count(tekst:Li/tekst:LiNummer) = 0">
       { "code": "STOP0001", "eId": "<sch:value-of select="@eId" />" }
    </sch:assert>
  </sch:rule>
  <sch:rule
    context="tekst:Lijst[@type = 'expliciet']">
    <sch:assert
      id="STOP0002"
      role="error"
      test="count(tekst:Li[descendant::tekst:LiNummer]) = count(tekst:Li)">
      { "code": "STOP0002", "eId": "<sch:value-of select="@eId" />" }
    </sch:assert>
  </sch:rule>
</sch:pattern>
```

Unit-testen

Voor elke sch:pattern worden ten miste 2 XML-voorbeelden gemaakt waarmee de functie van de schematron validatie kan worden getest. Deze test documenten voldoen aan de volgende condities:

Een document voor dit doel:

- is valide volgens de XSD;
- is zo klein mogelijk;
- Bevat niet noodzakelijk een volledig STOP-document, maar een afdoende fragment;
- 1 document Illustreert *alleen de fout* voor de specifieke validatie-regel;
- 1 document moet valide zijn (geen fout melding)
- Als commentaar is een korte omschrijving van de specifieke fout opgenomen
- De verwachte respons wordt in een apart json bestand geplaatst ter controle gebruik juiste gegevens voor de melding

De bestandsnaam van deze testdocumenten geeft altijd een korte omschrijving van de unit-test. Deze bestanden worden in een directory per businessrule met de code zoals in de `@id` van de `sch:assert` of `sch:report` is opgenomen als naam.

Voorbeeld testdocument (fout)

ongemarkeerde-lijst-zonder-linrs-fout.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Lijst eId="list_o_1" type="ongemarkeerd" wId="bgCode_1-0__list_o_1"
   xmlns="https://standaarden.overheid.nl/stop/imop/tekst/">
   <!-- Een Lijst van het type 'ongemarkeerd' MOET lijst-items hebben zonder nummering of symbolen -->
   <Lijstaanhef>Aanhef voor de lijst</Lijstaanhef>
   <Li eId="list_o_1__item_1" wId="bgCode_1-0__list_o_1__item_1">
      <LiNummer>1.</LiNummer>
      <Al>Tekst</Al>
   </Li>
   <Li eId="list_o_1__item_2" wId="bgCode_1-0__list_o_1__item_2">
      <LiNummer>2.</LiNummer>
      <Al>Tekst</Al>
   </Li>
   <Li eId="list_o_1__item_3" wId="bgCode_1-0__list_o_1__item_3">
      <LiNummer>3.</LiNummer>
      <Al>Tekst</Al>
   </Li>
</Lijst>
```

Verwacht resultaat

ongemarkeerde-lijst-zonder-linrs-fout.json

```
{
  "code": "STOP0001",
  "eId": "list_o_1"
}
```

Genereren documentatie

Er is een xslt script beschikbaar voor het genereren van documentatie in dita-formaat.

- Voor iedere sch:pattern wordt een `dita:topic` documentatie bestand gegenereerd.
- Voor ieder schematron bestand wordt een `dita:map` gegenereerd als inhoudsopgave, verwijzend naar de afzonderlijke topics
- In de `dita:map` worden de topics alfabetisch gerangschikt op basis van de `sch:title` in `sch:pattern`
- Bij iedere sch:pattern wordt voor de elementen genoemd in @see een referentie opgenomen die verwijst naar de documentatie voor dat element (of attribuut)
    - Aanname is dat de referentie naar een element ook resulteert in de verwijzing naar de schematron regels bij de schema-documentatie.
    - ***NB*** *- Bij voorkeur zou ook de sch:title van een sch:pattern in de documentatie van het element getoond moeten worden bij de terug-verwijzing, evt. handmatig toevoegen aan de schema-documentatie?*.
- Voor iedere `sch:assert` of `sch:report` wordt een documentatieblok opgenomen, voorafgegaan door de businessrule die wordt betrokken uit de XML-registratiebestand voor businessrules.
- De waarden van de attributen @see, @context en de @id's die gebruikt worden zijn opgenomen in de documentatie.
- De naamgeving wordt nederlandstalig:
    - assert - Handhaven
    - report - Rapporteer
    - error - Fout
    - warning - Waarschuwing
    - context - Context
    - Melding

Voorbeeld documentatie - (weergave zonder stijlen)

Lijst - Nummering lijstitems

	Schematron id "sch_tekst_001"	
Handhaven	Een Lijst van het type 'ongemarkeerd' MAG GEEN lijst-items met nummering of symbolen hebben	
	Melding id	[STOP0001]
	Melding	[STOP0001] De Lijst met @eId: (...) van type 'ongemarkeerd' heeft LiNummer elementen met een nummering of symbolen, dit is niet toegestaan. Pas het type van de lijst aan of verwijder de LiNummer elementen.
	Context	tekst:Lijst[@type = 'ongemarkeerd']
	Ernst	Fout - blokkerend
Handhaven	Een Lijst van het type 'expliciet' MOET lijst-items hebben met nummering of symbolen	
	Melding id	[STOP0002]
	Melding	[STOP0002] De Lijst met @eId (...) van type 'expliciet' heeft geen LiNummer elementen met nummering of symbolen, het gebruik van LiNummer is verplicht. Pas het type van de lijst aan of voeg LiNummer's met nummering of symbolen toe aan de lijst-items.
	Context	tekst:Lijst[@type = 'expliciet']
	Ernst	Fout - blokkerend
XML elementen	tekst_xsd_Element_tekst_Lijst.html#Lijst, tekst_xsd_Element_tekst_Li.html#Li, tekst_xsd_Element_tekst_LiNummer.html#LiNummer	

Registratie Bedrijfsregels

Voor registratieve doeleinden is er een schema en model om het bestaan van een bedrijfsregel vast te leggen; Business rules van STOP, BHKV en LVBB

Er is een xslt beschikbaar om dit uit een schematron bestanbd dat voldoet aan de converties te genereren - `sch2bedrijfsregels.xsl`