De opbouw van sectormodel schema's

Auteur: M. van den Broek
Versie: 1.1~~0~~

Inhoudsopgave

1 Inleiding

De afgelopen jaren is het aantal StUF-sectormodellen groter geworden. Vrijwel alle nieuwe sectormodellen gebruiken onderdelen van het sectormodel bg0310. De voor bg0310 gehanteerde opbouw van de schema's blijkt om twee redenen in de praktijk niet optimaal:

1. De schema's zijn erg groot en dit leidt tot traag laden in XML editors en tot problemen bij het genereren van code op basis van de schema's.
2. Bij het importeren van bg0310 in andere sectormodellen wordt er veel te veel geïmporteerd.

Daarnaast heeft de expertgroep in het najaar van 2010 vastgesteld dat nieuwe koppelvlakken binnen een sectormodel mogen worden toegevoegd zonder dat een nieuwe versie van het sectormodel wordt uitgebracht. Er is slechts een nieuwe versie van een sectormodel nodig, indien wijzigingen worden doorgevoerd in de complexTypes voor een entiteittype. Nieuwe berichtelementen en services mogen als een koppelvlak definitie aan een bestaand sectormodel worden toegevoegd. Het sectormodel wordt dan uitgebreid met een nieuw koppelvlak. Zie het beheermodel voor een nadere uitleg van dit mechanisme. Het is wenselijk om de onderkende koppelvlakken binnen een sectormodel zichtbaar te maken in de structuur van de schema's.

Dit document doet een voorstel voor een andere opbouw van de schema's en wsdl's voor een StUF-sectormodel dat met het bovenstaande rekening houdt. Het voorstel wordt geïllustreerd aan de hand van de sectormodellen bg0310 en zkn0310. Het koppelvlak BAG-WOZ binnen het sectormodel bg0310 wordt gebruikt om te illustreren hoe wordt omgegaan met nieuwe koppelvlakken.

2 Het zo scherp mogelijk definiëren van de berichten

Servicedefinities worden gezien als een contract tussen de aanbieder van een service en zijn afnemers. Het is best practice om zo scherp mogelijk te definiëren wat een service als verzoek verwacht en wat een service als respons op dat verzoek zal leveren. Bij het definiëren van StUF-services wordt waar mogelijk getracht om de services zo scherp mogelijk te definiëren. Dit leidt tot grote en complexe schema's. Het restriction-mechanisme van XML-Schema biedt goede mogelijkheden om deze complexiteit in de praktijk te beheersen.

Nu volgt een voorbeeld om het bovenstaande concreter te maken. In de stuurgegevens van een Lk01-bericht voor een natuurlijk persoon zijn bijvoorbeeld het berichttype “Lk01” en het entiteittype “NPS” verplicht. In de StUF-namespace is een StuurgegevensLk01-complexType gedefinieerd voor Lk01-kennisgevingen met “Lk01” voor het berichttype, maar zonder een voorschrift voor het entiteittype. De vulling voor het entiteittype wordt gedefinieerd in een complexType StuurgegevensLk01-NPS, dat “NPS” als waarde voor entiteittype voorschrijft. StuurgegevensLk01-NPS is een restriction op StuurgegevensLk01. De restrictions op complexTypes uit de StUF-namespace worden gedefinieerd in schema's voor de StUF-namespace die onderdeel zijn van het te definiëren sectormodel in dit geval van het sectormodel bg0310.

Dezelfde strategie wordt gevolgd als in een sectormodel slechts een deel van de gegevens nodig is van een complexType gedefinieerd in een ander sectormodel bg0310. In een schema in de namespace van bg0310 wordt in een ander sectormodel bijvoorbeeld een restriction gedefinieerd op het NPS-complexType uit bg0310 met daarin alleen de gegevens die binnen dat sectormodel worden gebruikt.

Deze werkwijze heeft een belangrijke consequentie. In sectormodellen worden schema's gedefinieerd voor de namespace stuf0301 of bg0310 met restrictions op complexTypes gedefinieerd in die namespace. Helaas bevat de specificatie van XML Schema een dubbelzinnigheid bij het omgaan met imports van een andere namespace. Een implementatie heeft namelijk de vrijheid om alleen de laatste import te behouden of om alle imports voor een bepaalde namespace “bij elkaar op te tellen”. De eerste strategie leidt tot het niet altijd valideren van een set schema's waarbij een namespace door middel van verschillende schema's wordt geïmporteerd, omdat voor het ene schema in de set een ander schema met restrictions gebruikt dan een ander schema uit de set.

Praktische consequenties
De hierboven geformuleerde werkwijze wordt niet door alle tooling ondersteund. XMLSpy heeft er geen probleem mee. Oxygen kan er een probleem mee hebben, maar daar kan het verholpen worden door het zet-

ten van de optie “honour-all-schemalocations” voor de schema parser. De SaxonEE parser kent een dergelijke optie niet en ook de LibXML library, die gebruikt wordt voor webservices in de php-wereld, kent deze optie niet, waardoor de webservices tooling voor php niet werkt met de standaard geleverde schema's .

Het probleem met het importeren kan opgelost worden door één overkoepelend schema voor de desbetreffende namespace te definiëren dat alle restrictions nodig voor dat sectormodel bevat en dit schema overal te gebruiken waar de betreffende namespace wordt geïmporteerd. Het is niet mogelijk dergelijke “optel”-schema's centraal te definiëren, omdat ook import-statements in schema's voor andere sectormodellen dan het eigen sectormodel dienen te worden aangepast. Bij het definiëren van de schema's zal ervan uitgegaan worden dat tooling in staat is geïmporteerde schema's “op te tellen”. Partijen met tooling die niet “optelt”, dienen eigen versies te maken van schema's van StUF en van geïmporteerde sectormodellen en deze ook te onderhouden bij bugfixes.

3 Van domeinmodel naar berichtmodel

Een domeinmodel geeft een semantische beschrijving van het domein waarover in de vorm van berichten informatie zal worden uitgewisseld. Het berichtmodel definieert met welke berichten de informatie precies uitgewisseld gaat worden. Het is lang niet altijd zo dat een domeinmodel één op één vertaald kan worden naar een berichtmodel. Daarnaast is sowieso een vertaling noodzakelijk van de modelleringstaal voor het domeinmodel (bij voorkeur UML) naar XML Schema, de taal waarin berichten worden gedefinieerd. Dit hoofdstuk gaat dieper in op de vertaling van de objecttypen binnen het domeinmodel naar de StUF-entiteittypen (in XML Schema gedefinieerd als complexTypes) gebruikt binnen de berichten.

3.1 Van domeinen naar simpleTypes

Het informatiemodel dat ten grondslag ligt aan een sectormodel definieert de domeinen voor de attribuutsoorten. In XML-Schema worden domeinen gedefinieerd als simpleTypes. Voor elk nog niet elders binnen de StUF-familie gedefinieerd domein uit een informatiemodel wordt in een sectormodel een simpleType gedefinieerd. Als voor een domein al ergens binnen de StUF-familie een simpleType is gedefinieerd, bijvoorbeeld in de StUF-standaard zelf of in een horizontaal sectormodel, dan wordt dit simpleType gebruikt.

Normaal gesproken wordt voor elk simpleType voor een domein ook als extension een complexType met simpleContent gedefinieerd dat aan het simpleType de attributes StUF:noValue en StUF:exact toevoegt. Het attribute StUF:noValue is nodig voor de door StUF gedefinieerde bijzondere waarden geenWaarde, waardeOnbekend, waardeVastgesteldOnbekend, nietOndersteund en nietGeautoriseerd. Het attribute StUF:exact is nodig binnen vraagberichten. Deze complexTypes met simpleContent krijgen als naam de naam van het simpleType met daaraan toegevoegd “-e”. Om het gebruik van StUF:noValue mogelijk te maken dient aan een element met een “-e”-complexType altijd het attribute “xsi:nil=”true” te worden toegevoegd.

De StUF-standaard heeft simpleTypes gedefinieerd voor de volgende domeinen:

- Sleutel
  Ten behoeve van identificerende sleutels heeft een StUF het simpleType Sleutel gedefinieerd als string met een maximale lengte van 40. Dit simpleType wordt voor de door StUF gedefinieerde sleutelOntvangend, sleutelGegevensbeheer en sleutelVerzendend. Naast Sleutel is ook Sleutel-e gedefinieerd.
- Datum en tijdstip
  StUF0301 definieert in het simpleType Datum een datum als JJJJMMDD en in het simpleType Tijdstip een tijdstip als JJJJMMDDhhmmssSSS. Dit is niet in overeenstemming met de ISO-normen voor datum en tijdstip. StUF definieert ook Datum-e en Tijdstip-e. StUF kent daarnaast zogenaamde onvolledige datums voor het geval in een datum de dag, de maand of het jaar onbekend is. DatumMetIndicator en TijdstipMetIndicator zijn een extension op Datum-e en Tijdstip-e waarbij het attribute StUF:indOnvolledigeDatum wordt toegevoegd.
  In een volgende StUF-versie zullen de voorschriften van het stelsel van basisregistraties voor datum en tijd worden overgenomen. Hierop vooruitlopend is het verstandig om in sectormodellen alvast de door het stelsel voorgeschreven datum en tijdstip formaten te gebruiken.

- Groepsnaam
  Ten behoeve van de attributes elementnaam en groepsnaam binnen de metagegevens inOnderzoek, brondocument en gebeurtenis definieert StUF het simpleType groepsnaam.
- StatusMetagegeven
  Ten behoeve van het metagegeven inOnderzoek definieert StUF het simpleType StatusMetagegeven met als enige toegestane waarde 'J'. In een entiteittype wordt niet dit simpleType gebruikt maar een complexType met simpleContent waarbij aan StUF:StatusMetagegeven in elk geval zijn toegevoegd de attributes StUF:metagegeven en StUF:noValue en mogelijk de attributes elementnaam en groepsnaam. In een inOnderzoek element dient altijd xsi:nil=”true” opgenomen te worden. In een vraagbericht hoort een inOnderzoek element zonder attributes te worden opgenomen.
- Gebeurtenis
  Ten behoeve van het metagegeven gebeurtenis definieert StUF het simpleType Gebeurtenis als een string met een maximale lengte van 200. In een entiteittype wordt niet dit simpleType gebruikt maar een complexType met simpleContent waarbij aan StUF:Gebeurtenis in elk geval zijn toegevoegd het attribute StUF:metagegeven en tijdstip en mogelijk de attributes elementnaam en groepsnaam. In een vraagbericht hoort een gebeurtenis-element zonder attributes te worden opgenomen.

De voor een sectormodel te definiëren simpleTypes en “-e” complexTypes worden opgenomen in een apart schema met als naam de sectormodelcode in kleine letters gevolgd door de sectormodelversie gevolgd door “_simpleTypes.xsd”. Voor het sectormodel bg0310 is de naam dus bg0310_simpleTypes.xsd.

Voor een sectormodel worden in een schema met als target namespace StUF zonodig restrictions gedefinieerd op de door StUF gedefinieerde complexTypes met simpleContent voor StatusMetagegeven en Gebeurtenis. In deze restrictions kan gedefinieerd worden welke waarden voor de attributes groepsnaam en elementnaam zijn toegestaan. Dit schema krijgt als naam de sectormodelcode in kleine letters gevolgd door de sectormodelversie en gevolgd door “_simpleTypes_stuf.xsd”. Voor het sectormodel bg0310 is de naam dus bg0310_simpleTypes_stuf.xsd.

Indien een sectormodel restrictions nodig heeft op simpleTypes uit een ander sectormodel dan worden deze gedefinieerd in een schema met als naam de sectormodelcode in kleine letters gevolgd door de sectormodelversie gevolgd door “_”, gevolgd door de sectormodelcode en -versie van het te restricten sectormodel in kleine letters gevolgd door “_simpleTypes.xsd”. Voor restrictions op simpleTypes uit het sectormodel bg0310 binnen het sectormodel zkn0310 is de naam dus zkn0310_simpleTypes_bg0310.xsd.

3.2 Van objecttypen naar StUF-entiteittypen en complexTypes

In het informatiemodel wordt gewoonlijk gesproken over objecttypen. Bij het ontwerpen van een sectormodel dienen deze objecttypen vertaald te worden naar de StUF-entiteiten binnen de berichten. In XML-Schema worden deze entiteitttypen voor StUF-entiteiten gedefinieerd met behulp van complexTypes. Bij de vertaling van objecttypen uit het informatiemodel naar de entiteittypen worden allerhande keuzen gemaakt. Deze keuzen dienen te worden vastgelegd in een zogenaamd keuzenVerStUFfing-document.

Een kennisgevingbericht stelt andere eisen aan de inhoud van een StUF-entiteit dan een vraagbericht of een antwoordbericht. Een StUF-entiteit voor de actuele gegevens dient anders gevuld te worden dan een StUF-entiteit voor de materiële historie, de formele historie of de formele historie van een relatie. Een StUF-entiteit voor een gerelateerde in een kennisgeving- of een vraag- of antwoordbericht wordt ook weer anders gevuld. Gegeven het uitgangspunt om de berichten zo scherp te mogelijk te definiëren zullen voor al deze functies aparte complexTypes worden gedefinieerd. Daarnaast wordt er nog een complexType gedefinieerd met de kerngegevens om in het schema ook de kerngegevens te specificeren. Al deze complexTypes worden afgeleid van een zogenaamd basis-complexType met als naam de mnemonic voor het entiteittype en als suffix “-basis”. Voor een natuurlijk persoon is er dus een complexType “NPS-basis” en voor de relatie van een natuurlijk persoon naar zijn verblijfsobject (met als mnemonic TGO) een complexType “NPSTGO-basis”.

De hierboven genoemde complexTypes krijgen allemaal een ander suffix. Hieronder volgt een lijst met de standaard suffixen en hun doel:

- -antwoord: Het complexType gebruikt binnen een antwoordbericht.

- -basis: Het complexType waar alle andere complexTypes van worden afgeleid via restriction.
- -historieFormeel: Het complexType met de elementen waarvoor formele historie is gedefinieerd in het informatiemodel.
- -historieFormeelRelatie: Het complexType voor de formele historie ten behoeve van onterecht gelegde relaties
- -historieMateriaal: Het complexType met de elementen waarvoor materiële historie is gedefinieerd in het informatiemodel
- -kennisgeving: Het complexType gebruikt binnen een kennisgevingbericht
- -kerngegevens: Het complexType met de kerngegevens. Dit complexType wordt bij het basis-complexType gedefinieerd, als er binnen de kennisgeving-complexTypes geen kerngegevensKennisgeving type wordt gedefinieerd of als het nodig is in de overige complexTypes
- -kerngegevensKennisgeving: Het complexType met de kerngegevens binnen een kennisgeving. Dit complexType wordt gebruikt voor de gerelateerde in een relatie.
- -vraag: Het complexType gebruikt binnen een vraagbericht voor zowel de selectie-elementen als het scope-element.
- -vraagScope: Het complexType gebruikt voor het scope-element in een vraagbericht, indien het scope-element een andere inhoud heeft dan de selectie-elementen.
- -vraagSelectie: Het complexType gebruikt voor de selectie-elementen in een vraagbericht, indien de selectie-elementen een andere inhoud hebben dan het scope-element.

Daarnaast staat het de ontwerper van een sectormodel vrij om nog meer complexTypes te definiëren voor bijvoorbeeld gerelateerden in een antwoordbericht of voor gebruik in vrije berichten. De suffix voor een complexType voor een gerelateerde in een antwoordbericht begint altijd met "-gerelateerde" gevolgd door een aanduiding van het soort gerelateerde.

In de nu volgende paragrafen worden de belangrijkste kenmerken van de hierboven gedefinieerde complexTypes beschreven.

Het "basis"-complexType
Het complexType met het suffix "-basis" definieert de maximale omvang van een antwoordbericht. De exacte inhoud ervan volgt uit de hieronder gegeven richtlijnen voor de overige typen, omdat deze allemaal van het "basis"-complexType worden afgeleid door middel van het restriction-mechanisme. Het "basis"-complexType bevat het attribute StUF:entiteittype met als waarde de mnemonic voor het entiteittype, maar dit attribute is niet verplicht, omdat het niet mag voorkomen binnen de complexTypes voor de historie. Daarnaast bevat het de StUF-attributes voor een StUF-entiteit. Het is aan de ontwerper van het sectormodel of de attributes StUF:sleutelOntvangend, StUF:sleutelGegevensbeheer en StUF:sleutelVerzendend al dan niet worden opgenomen. Binnen een "basis"-complexType zijn de complexTypes voor de gerelateerden van relaties en voor de historie-elementen altijd "basis"-complexTypes.

Voor het definiëren van de "basis"-complexTypes zijn uitsluitend "basis"-complexTypes nodig. Om deze reden wordt ervoor gekozen om de "basis"-complexTypes te definiëren in een apart schema. Het schema voor de "basis"-complexTypes include het schema met de simpleTypes voor het sectormodel en importeert zonodig de schema's met StUF-simpleTypes of simpleTypes uit andere sectormodellen. De geïmporteerde schema's bevatten zonodig de benodigde restrictions op de te importeren simpleTypes. Het schema met de "basis"-complexTypes krijgt als naam de sectormodelcode in kleine letters gevolgd door de sectormodelversie en gevolgd door "_basis.xsd". Voor het sectormodel bg0310 is de naam dus bg0310_basis.xsd. Bij het importeren van een ander sectormodel ten behoeve van het definiëren van restrictions wordt altijd het "basis"-schema gebruikt. Met andere woorden deze restrictions worden altijd gedefinieerd op de "basis"-complexTypes. Op deze manier wordt de import vanuit het andere sectormodel zo klein mogelijk gehouden.

Het "antwoord"-complexType
Het "antwoord"-complexType bevat zonodig de door StUF voorgeschreven elementen historieMaterieel, historieFormeel en in relaties ook historieFormeelRelatie met de hiervoor gedefinieerde complexTypes. Het historieMaterieel-element kan nul of meeer keren voorkomen en het historieFormeel- en historieFormeelRelatie-element nul of één keer. Voor de gerelateerden worden veelal specifiek voor een bepaald

antwoordbericht gedefinieerde complexTypes gebruikt. Relaties die volgens het informatiemodel maar eindig aantal keren kunnen voorkomen, maar waarvoor materiële historie is gedefinieerd, krijgen mogen in een antwoordbericht een onbeperkt aantal keren voorkomen (maxOccurs unbounded). De relatie van een persoon naar een verblijfsobject heeft in het informatiemodel bijvoorbeeld een kardinaliteit één, maar komt in een antwoordbericht meerdere keren voor, omdat een persoon in de loop van de tijd in verschillende verblijfsobjecten heeft gewoond. In de topfundamenteel, een relatie of een gerelateerde binnen een antwoordbericht mogen de attributes StUF:verwerkingssoort en StUF:scope niet voorkomen. Het attribute StUF:noValue mag bovendien niet voorkomen in de topfundamenteel en gerelateerden van een antwoordbericht.

Voor de in het informatiemodel meervoudig voorkomende relaties dient binnen het antwoordbericht nog de sortering te worden gedefinieerd. Dit wordt gedaan door binnen het "antwoord"-complexType een annotation met appinfo op te nemen. Binnen het appinfo element wordt de sorteringRelatie gespecificeerd door binnen het element sortering één of meer elementen element op te nemen met de Xpath expression voor het element waarop gesorteerd dient te worden. Als er in aflopende volgorde gesorteerd dient te worden, dan wordt binnen <element> het attribute order="DESC" opgenomen. Hieronder staat een voorbeeld voor deze annotation voor de huwelijksrelatie van een persoon.

```
<annotation>
    <appinfo>
        <StUF:sorteringRelatie>
            <StUF:element>gerelateerde/geslachtsnaam</StUF:element>
            <StUF:element order="DESC">datumSluiting</StUF:element>
        </StUF:sortering>
    </appinfo>
</annotation>
```

De inhoud van deze appinfo is in het StUF-schema gedefinieerd als het ~~complexType~~element s~~S~~orteringRelatie.

De "antwoord"-complexTypes worden gedefinieerd in een schema met als naam sss_vraagAntwoord.xsd dat een include doet van het schema sss_basis.xsd.

De "historie"- en "kerngegevens"-complexTypes
Het "historieMaterieel"- en "historieFormeel"-complexType bevatten de elementen van een entiteittype waarvoor in het informatiemodel materiële respectievelijk formele historie is gedefinieerd. Als er voor een entiteittype geen historie is gedefinieerd dan komen deze complexTypes niet voor. Het "historieMaterieel" complexType bevat altijd het element StUF:tijdvakGeldigheid met minOccurs="1" (verplicht) en als ook formele historie is gedefinieerd het element StUF:tijdstipRegistratie met minOccurs="0" (optioneel). Het "historieFormeel" complexType bevat altijd de elementen StUF:tijdvakGeldigheid en StUF:tijdstipRegistratie met minOccurs="1" (verplicht). Het "historieMaterieel"- en "historieFormeel"-complexType voor een entiteittype waarvoor formele historie is gedefinieerd bevatten een optioneel historieFormeel-element. Het "historieFormeelRelatie"-complexType bevat altijd een optioneel "historieFormeel"- en "historieFormeelRelatie"-element. Het "historieMaterieel"- en "historieFormeel"-complexType voor een relatie bevatten geen gerelateerde. Het "historieFormeelRelatie"-complexType voor een relatie bevat dezelfde elementen als het "historieFormeel"-complexType plus een verplichte gerelateerde met het "kerngegevens"-complexType. Alle drie de "historie"-complexTypes bevatten geen StUF-attributes.

De "historie"- en "kerngegevens"-complexTypes worden gedefinieerd in een schema met als naam sss_vraagAntwoord.xsd dat een include doet van het schema sss_basis.xsd.

Het "kennisgeving"- en "kerngegevensKennisgeving"-complexType
Het "kennisgeving"-complexType bevat geen historie-elementen. Als sleutelOntvangend wordt ondersteund, zijn alle elementen in een "kennisgeving"-complexType optioneel. Als sleutelOntvangend niet wordt ondersteund, dan zijn de kerngegevens verplicht in een kennisgeving. Relaties komen erin voor met de kardinaliteit voorgeschreven door het informatiemodel. Alleen de relaties die vanuit de topfundamenteel worden onderhouden worden opgenomen in het "kennisgeving"-complexType. In een "kennisgeving"-com-

plexType waarvoor historieMaterieel is gedefinieerd moet het element StUF:tijdvakGeldigheid als niet verplicht element worden opgenomen. In een “kennisgeving”-complexType waarvoor historieFormeel is gedefinieerd moet StUF:tijdstipRegistratie als niet verplicht element worden opgenomen. In de topfundamenteel, de relaties en de gerelateerde is het attribute StUF:verwerkingssoort verplicht en mag het attribute StUF:scope niet voorkomen. Alleen binnen relaties mag het attribute StUF:noValue voorkomen. Voor gerelateerden wordt meestal het “kerngegevensKennisgeving”-complexType, tenzij er van de gerelateerde meer gegevens dan alleen de kerngegevens mogen worden toegevoegd of gewijzigd. Er worden normaal gesproken geen aparte typen gedefinieerd voor een toevoegkennisgeving (mutatiesoort “T”), een wijzig- of correctiekennisgeving (mutatiesoort “C”, “F” of “W”) en een verwijderkennisgeving (mutatiesoort “V”).

De “kennisgeving”- en “kerngegevensKennisgeving”-complexTypes worden gedefinieerd in een schema met als naam sss_mutatie.xsd dat een include doet van het schema sss_basis.xsd.

Het “vraag”-complexType of het “vraagScope”- en “vraagSelectie”-complexType
Het “vraag”-complexType bevat geen historie-elementen en het attribute StUF:verwerkingssoort mag er niet in voorkomen. In een vraag-“complexType” mogen de elementen voor attributen en relaties maximaal één keer voorkomen. Alleen in de topfundamenteel van een “vraag”-complexType mag het attribute StUF:scope voorkomen. Voor een supertype zijn afzonderlijke “vraagScope”- en “vraagSelectie” complexTypes nodig. Het “vraagSelectie”-complexType bevat de elementen van het supertype en het “vraagScope”-complexType bevat alle subtypen van het supertype, omdat voor elk subtype gedefinieerd moet kunnen worden welke elementen je vraagt. Als een supertype als gerelateerde voorkomt in een entiteittype, dan zijn voor dat entiteittype ook afzonderlijke “vraagScope”- en “vraagSelectie”-complexTypes nodig. Hetzelfde geldt voor entiteittypen waarbinnen als gerelateerde een entiteittype voorkomt met afzonderlijke “vraagScope”- en “vraagSelectie”-complexTypes.

De “vraag”-, “vraagScope”- en “vraagSelectie”-complexTypes worden gedefinieerd in een schema met als naam sss_vraagAntwoord.xsd dat een include doet van het schema sss_basis.xsd.

Het definiëren van subtypen voor een reeds in een ander sectormodel gedefinieerd entiteittype
In de praktijk komt het geregeld voor dat van een persoon in ander sectormodel meer gegevens moeten worden vastgelegd dan beschikbaar zijn het sectormodel bg0310. Het is dan tevens vaak zo dat niet alle gegevens uit het sectormodel bg0310 relevant zijn. In dat geval wordt de volgende werkwijze aanbevolen:

- Maak in de namespace van het oorspronkelijke entiteittype als restriction een complexType aan dat precies de gewenste elementen bevat. Geef dit complexType als naam XXX-basis gevolgd door de code voor het sectormodel waarbinnen dit complexType gebruikt gaat worden. Zo'n complexType voor NPS binnen het sectormodel Zaken heet bijvoorbeeld NPS-basisZKN.
- Maak in het basisschema voor het sectormodel een complexType YYY-basis aan dat als eerste element een relatie isEen zonder elementen bevat met een gerelateerde van het zojuist gedefinieerde XXX-basisSS complexType. De mnemonic YYY mag gelijk zijn aan de mnemonic XXX als het niet gaat om een subtype, maar alleen om het toevoegen en verwijderen van elementen uit XXX.
- Voeg aan dit complexType YYY-basis alle elementen en relaties toe die extra benodigd zijn.

Op YYY-basis kunnen vervolgens op de normale manier weer restrictions gedefinieerd worden.

Er is een RFC op de StUF-standaard ingediend om de isEen constructie in de standaard op te nemen, zodat er geen relatie meer gebruikt hoeft te worden, maar aan het isEen element zelf al het type XXX-basisSS kan worden toegekend.

Tenslotte
In alle hierboven gedefinieerde complexTypes en in alle complexTypes die als restriction daarop gedefinieerd worden, dient te worden opgenomen final=”extension” om expliciet te specificeren dat er van deze complexTypes geen extension gedefinieerd mag worden. Deze specificatie is overigens niet waterdicht, want als je eerst een restriction definieert en vervolgens op die restriction weer een extension, dan is dat toegestaan.

Hiermee is de vertaling van het domeinmodel naar StUF-entiteittypen en complexTypes beschreven en hebben we de bouwstenen voor het definiëren van de koppelvlakken met hun berichten en services.

4 Koppelvlakken

De StUF-standaard wordt meestal ingezet in situaties waarin systemen elkaar mutaties willen doorgeven, het ene systeem een object wil synchroniseren met een ander systeem en waarbij het systemen elkaar willen bevragen. Omdat dit de meest voorkomende toepassingen voor StUF zijn worden hieronder de drie koppelvlakken voor deze toepassingen beschreven. Het staat de ontwerper van een sectormodel vrij om meer koppelvlakken of andere koppelvlakken te definiëren. Wat hieronder staat is een best practice en geen dwingend voorschrift.

4.1 Kennisgevingkoppelvlak

Mutaties worden doorgegeven met behulp van kennisgevingberichten. StUF kent langs twee dimensies zes verschillende soorten kennisgevingen. De ene dimensie is de afhandeling van het bericht: synchroon of asynchroon. De andere dimensie is het soort kennisgeving:

- enkelvoudige kennisgeving met betrekking tot één object (asynchroon: Lk01 en synchroon: Lk02)
- samengestelde kennisgeving over meerdere objecten (asynchroon: Lk03 en synchroon: Lk04)
- toekomstkennisgeving, waarbij de ingangsdatum van de mutatie in de toekomst ligt (asynchroon: Lk05 en synchroon: Lk06).

Een enkelvoudige kennisgeving en een toekomstkennisgeving krijgen als elementnaam de mnemonic van het entiteittype waarover de kennisgeving gaat in kleine letters gevolgd door de berichtcode voor de kennisgeving, wanneer het object-element als complexType XXX-kennisgeving heeft. Een asynchrone enkelvoudige kennisgeving voor een natuurlijk persoon krijgt dus als elementnaam npsLk01 en een synchrone toekomstkennisgeving voor een adres aoaLk06. Een samengestelde kennisgeving krijgt als elementnaam de berichtcode in kleine letters gevolgd door een “-” en de functie uit de stuurgegevens in camel case (de eerste letter een kleine letter, alle spaties verwijderd en elke letter na een spatie een hoofdletter).

Een enkelvoudige kennisgeving en een toekomstkennisgeving bevat drie verschillende elementen:

- stuurgegevens
- parameters
- object (één of twee keer)

Het complexType voor het stuurgegevens element wordt gedefinieerd als restriction op het StUF:Stuurgegevens-Lk0n complexType met als waarde voor entiteittype de mnemonic van het object waarover de mutatie gaat. Het complexType voor het parameters element wordt gebruikt uit het StUF-schema of als restriction gedefinieerd op ParametersKennisgeving uit het StUF-schema. Voor het object element wordt het complexType gebruikt dat hierboven als XXX-kennisgeving is beschreven. De XXX-kennisgeving complexTypes en de berichtelementen worden gedefinieerd in een schema met als naam de sectormodelcode in kleine letters gevolgd door de sectormodelversie en gevolgd door “msg_kennisgeving.xsd”. Voor het sectormodel bg0310 is de naam dus bg0310_msg_kennisgeving.xsd. De restrictions op de StUF-complexTypes worden gedefinieerd in een schema met als naam de sectormodelcode in kleine letters gevolgd door de sectormodelversie en gevolgd door “_kennisgeving_stuf.xsd”. Voor het sectormodel bg0310 is de naam dus bg0310_kennisgeving_stuf.xsd. Het schema bg0310_msg_kennisgeving.xsd doet een include van bg0310_mutatie.xsd en een import van bg0310_kennisgeving_stuf.xsd.

Een samengestelde kennisgeving bevat naast het stuurgegevens- en het parameters-element twee of meer kennisgevingberichten. Deze kennisgevingberichten kunnen apart voor de samengestelde kennisgeving worden gedefinieerd in het mutatie-schema voor het sectormodel of er kan met behulp van een 'ref=' worden verwezen naar reeds gedefinieerde enkelvoudige kennisgevingen of toekomstkennisgevingen. Speciaal gedefinieerde kennisgevingen krijgen als naam xxxLk0n gevolgd door “-” en een aanduiding van de aard van de restriction ten opzichte van het standaard xxxLk0n-bericht. De stuurgegegevens en eventuele parameters-Kennisgeving complexTypes worden gedefinieerd in het kennisgeving_stuf schema voor het sectormodel.

Het sss_msg_kennisgeving.xsd en het sss_kennisgeving_stuf.xsd schema definiëren tezamen het koppelvlak voor het doorgeven van mutaties voor het sectormodel sss.

4.2 Synchronisatiekoppelvlak

Het synchronisatiekoppelvlak wordt gedefinieerd in de schema's sss_msg_synchronisatie.xsd en sss_synchronisatie_stuf.xsd. sss_synchronisatie_stuf.xsd bevat de restrictions op de StUF:Stuurgegevens complexTypes voor synchronisatieberichten. Het element entiteittype krijgt als waarde de mnemonic van het entiteitttype dat wordt gesynchroniseeerd.

Een synchronisatiebericht krijgt als naam de mnemonic voor het entiteittype in kleine letters gevolgd door de berichtcode voor het bericht, bijvoorbeeld npsSh01 voor het asynchrone synchronisatie-historisch bericht voor een natuurlijk persoon.

Een vraag-om-synchronisatie bericht (Sa03, Sa04, Sh03 en Sh04) bevat na het stuurgegevens element de kerngegevens van het object waarom gevraagd wordt. Het hiervoor benodigde complexType XXX-kerngegevens wordt gedefinieerd in het schema sss_mutatie.xsd.

Een synchronisatie-actueel bericht (Sa01 en Sa02) bevat na het stuurgegevens element een toevoegkennisgeving. Voor deze toevoegkennisgeving wordt in het schema sss_msg_synchronisatie.xsd een complexType XXX-Lk0nT gedefinieerd met XXX de mnemonic voor het objecttype. Daarnaast wordt ook voor het synchronisatie-actueel bericht zelf een complexType XXX-Sa0n gedefinieerd, omdat dit bericht ook voorkomt in een synchronisatie-historisch bericht. Het berichtelement xxxSa0n wordt gedefinieerd met als complexType XXX-Sa0n.

Een synchronisatie-historisch bericht (Sh01 en Sh02) wordt gedefinieerd voor alle objecttypen waarvoor historie is gedefinieerd. Het bevat na de stuurgegevens een element actueel met een synchronisatie-actueel bericht gevolgd door een historie element met daarin eerst een element oudste met een toevoegkennisgeving voor de oudste situatie gevolgd door 0 of meer wijzigkennisgevingen voor de opbouw van de historie tot en met de laatste situatie. Voor deze wijzigingkennisgevingen wordt in het schema sss_msg_synchronisatie.xsd een complexType XXX-Lk0nW gedefinieerd.

Het schema sss_synchronisatie_stuf.xsd doet een include van het schema sss_kennisgeving_stuf.xsd. Het schema sss_msg_synchronisatie.xsd doet een include van het schema sss_msg_kennisgeving.xsd en een import van het schema sss_synchronisatie_stuf.xsd.

4.3 Vraag/antwoordkoppelvlak

Het vraag/antwoordkoppelvlak wordt gedefinieerd in de schema's sss_msg_vraagAntwoord.xsd en sss_vraagAntwoord_stuf.xsd. sss_vraagAntwoord_stuf.xsd bevat de restrictions op de StUF:Stuurgegevens complexTypes voor vraag- en antwoordberichten en de restrictions op de StUF:ParametersVraag complexTypes. In de restrictions op de StUF:Stuurgegevens complexType krijgt het element entiteittype weer als waarde de mnemonic van het entiteittype dat wordt gesynchroniseeerd.

In de restrictions op de StUF:ParametersVraag complexTypes worden de sorteringen voor een entiteittype gespecificeerd. Voor elk fundamenteel entiteitttype wordt een simpleType XXX-sortering gedefinieerd als restriction op StUF:Sortering met XXX de mnemonic voor het entiteittype. Hieronder worden nog nadere voorschriften gegeven voor de definitie van het simpleType XXX-sortering.
Er worden maximaal zes restrictions op StUF:ParametersVraag gedefinieerd:

1. ParameterVraagSynchroon waarin de elementen peiltijdstipMaterieel en peiltijdstipFormeel worden weggelaten en de elementen sortering en vervolgvraag verplicht zijn. Het element sortering krijgt als type XXX-sortering. XXX-ParametersVraagSynchroon wordt gebruikt in de Lv01, Lv07, Lv09, Lv11 en Lv13-vraagberichten.
2. ParametersVraagAsynchroon waarin de elementen indicatorAantal, peiltijdstipMaterieel en peiltijdstipFormeel worden weggelaten en de elementen sortering en vervolgvraag verplicht zijn. Het

element sortering krijgt als type XXX-sortering. XXX-ParametersVraagAsynchroon wordt gebruikt in de Lv02, Lv04, Lv10, Lv12 en Lv14-vraagberichten.

3. ParametersVraagSynchroonMaterieel waarin het element peiltijdstipFormeel worden weggelaten en de elementen sortering en vervolgvraag verplicht zijn. Het element sortering krijgt als type XXX-sortering. XXX-ParametersVraagSynchroonMaterieel wordt gebruikt in het Lv03-vraagbericht.
4. ParametersVraagAsynchroonMaterieel waarin de elementen indicatorAantal en peiltijdstipFormeel worden weggelaten en de elementen sortering en vervolgvraag verplicht zijn. Het element sortering krijgt als type XXX-sortering. XXX-ParametersVraagAsynchroonMaterieel wordt gebruikt in het Lv04-vraagbericht.
5. ParametersVraagSynchroonFormeel waarin de elementen sortering en vervolgvraag verplicht zijn. Het element sortering krijgt als type XXX-sortering. XXX-ParametersVraagSynchroonFormeel wordt gebruikt in het Lv05-vraagbericht.
6. ParametersVraagAsynchroonFormeel waarin het element indicatorAantal wordt weggelaten en de elementen sortering en vervolgvraag verplicht zijn. Het element sortering krijgt als type XXX-sortering. XXX-ParametersVraagAsynchroonFormeel wordt gebruikt in het Lv06-vraagbericht.

De restrictions ParametersVraagSynchroonMaterieel en ParametersVraagAsynchroonMaterieel worden alleen gedefinieerd als er materiële historie is gedefinieerd voor het entiteittype. De restrictions ParametersVraagSynchroonFormeel en ParametersVraagAsynchroonFormeel worden alleen gedefinieerd als er formele historie is gedefinieerd voor het entiteittype.

In de restriction XXX-Sortering op StUF:Sortering wordt via het facet <maxInclusive value=”nn”/> het aantal sorteringen voor het entiteittype XXX gezet. Daarnaast worden in een <annotation> van dit simpleType als <appinfo> de sorteringen gespecificeerd. <appinfo> bevat één of meer elementen <sorteringObject>. Het element <sorteringObject> bevat allereerst het element <nummer> met het nummer van de sortering gevolgd door één of meer <element> elementen met de elementen uit de sortering. Een sorteringselement wordt gespecificeerd als een Xpath expression. Wanneer op een element Descending gesorteerd wordt, dan wordt op het <element> element een attribute order=”DESC” toegevoegd. De sorteringen voor NPS worden bijvoorbeeld als volgt geannoteerd:

```
<annotation>
   <appinfo>
      <StUF:sorteringObject>
         <StUF:nummer>1</StUF:nummer>
         <StUF:element>geslachtsnaam</StUF:element>
         <StUF:element>voorletters</StUF:element>
         <StUF:element>voorvoegselGeslachtsnaam</StUF:element>
      </StUF:sorteringObject>
      <StUF:sorteringObject>
         <StUF:nummer>2</StUF:nummer>
         <StUF:element>verblijfsadres/aoa.identificatie</StUF:element>
      </StUF:sorteringObject>
      <StUF:sorteringObject>
         <StUF:nummer>3</StUF:nummer>
         <StUF:element>verblijfsadres/aoa.postcode</StUF:element>
         <StUF:element>verblijfsadres/aoa.huisnummer</StUF:element>
         <StUF:element>verblijfsadres/aoa.huisletter</StUF:element>
      </StUF:sorteringObject>
      <StUF:sorteringObject>
         <StUF:nummer>4</StUF:nummer>
         <StUF:element>verblijfsadres/gor.straatnaam</StUF:element>
         <StUF:element>verblijfsadres/aoa.huisnummer</StUF:element>
         <StUF:element>verblijfsadres/aoa.huisletter</StUF:element>
      </StUF:sorteringObject>
      <StUF:sorteringObject>
         <StUF:nummer>5</StUF:nummer>
         <StUF:element>sub.verblijfBuitenland/lnd.landcode</StUF:element>
         <StUF:element>sub.verblijfBuitenland/sub.adresBuitenland3</StUF:element>
         <StUF:element>sub.verblijfBuitenland/sub.adresBuitenland2</StUF:element>
```

```
            <StUF:element>sub.verblijfBuitenland/sub.adresBuitenland1</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>6</StUF:nummer>
            <StUF:element>sub.verblijfBuitenland/lnd.landnaam</StUF:element>
            <StUF:element>sub.verblijfBuitenland/sub.adresBuitenland3</StUF:element>
            <StUF:element>sub.verblijfBuitenland/sub.adresBuitenland2</StUF:element>
            <StUF:element>sub.verblijfBuitenland/sub.adresBuitenland1</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>7</StUF:nummer>
            <StUF:element>geboortedatum</StUF:element>
            <StUF:element>geslachtsnaam</StUF:element>
            <StUF:element>voorletters</StUF:element>
            <StUF:element>voorvoegselGeslachtsnaam</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>8</StUF:nummer>
            <StUF:element>inp.bsn</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>9</StUF:nummer>
            <StUF:element>inp.a-nummer</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>10</StUF:nummer>
            <StUF:element>sub.rekeningnummerBankGiro</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>11</StUF:nummer>
            <StUF:element>sub.telefoonnummer</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>12</StUF:nummer>
            <StUF:element>sub.emailadres</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>13</StUF:nummer>
            <StUF:element>rps.isEigenaarVan/gerelateerde/kvkNummer</StUF:element>
        </StUF:sorteringObject>
        <StUF:sorteringObject>
            <StUF:nummer>14</StUF:nummer>
            <StUF:element>anp.identificatie</StUF:element>
        </StUF:sorteringObject>
    </appinfo>
</annotation>
```

De inhoud van deze appinfo is in het StUF-schema gedefinieerd als het ~~complexType~~element s~~S~~orteringObject.

Omdat voor een entiteittype meerdere vraag-berichten worden gedefinieerd met allemaal dezelfde inhoud na het parameters-element wordt deze inhoud gedefinieerd als een group met als naam xxxVraagBody met xxx de mnemonic in kleine letters voor het entiteittype. XxxVraagBody bevat een sequence met de elementen gelijk, vanaf, totEnMet, scope en start. De elementen gelijk, vanaf en totEnMet krijgen als complexType XXX-vraagSelectie of als die niet gedefinieerd hoeft te worden XXX-vraag. Het element object binnen scope krijgt als type XXX-vraagScope of als die niet gedefinieerd hoeft te worden XXX-vraag. Het element object binnen start krijgt als type XXX-antwoord. De xxxVraagBody groups worden gedefinieerd in het sss_msg_vraagAntwoord.xsd schema. In dit schema worden ook de berichtelementen voor de vraagberichten gedefinieerd.

Een vraagbericht krijgt als naam de mnemonic voor het entiteittype in kleine letters gevolgd door de berichtcode voor het bericht, wanneer het het “vraag”- of het “vraagScope”- en het “vraagSelectie”-complexType gebruikt. Het synchrone vraagbericht zonder historie voor een natuurlijk persoon krijgt bijvoorbeeld als naam npsLk01. Een antwoordbericht krijgt als naam de mnemonic voor het entiteittype in kleine letters gevolgd door de berichtcode voor het bericht, wanneer het het “antwoord”-complexType gebruikt. Het synchrone antwoordbericht zonder historie voor een natuurlijk persoon krijgt bijvoorbeeld als naam npsLa01. De vraagbericht-elementen worden gedefinieerd in het schema sss_msg_vraagAntwoord.xsd met behulp van de in sss_vraagAntwoord_stuf.xsd gedefinieerde complexTypes en de in sss_msg_vraagAntwoord.xsd zelf gedefinieerd groups. De antwoordbericht-elementen worden ook gedefinieerd in het schema sss_msg_vraagAntwoord.xsd met behulp van de in sss_vraagAntwoord_stuf.xsd gedefinieerde complexTypes voor de stuurgegevens. Het object-element binnen het element antwoord krijgt als type XXX-antwoord.

Het schema sss_vraagAntwoord_stuf.xsd doet een include van het schema stuf0301.xsd. Het schema sss_msg_vraagAntwoord.xsd doet een include van het schema sss_vraagAntwoord.xsd en een import van het schema sss_vraagAntwoord_stuf.xsd.

4.4 Andere koppelvlakken

Hierboven is best practice beschreven voor het mutatie-, synchronisatie- en vraag/antwoordkoppelvlak. Andere koppelvlakken worden op een soortgelijke wijze gedefinieerd. Voor een koppelvlak wordt altijd een schema met als naam sss_msg_kkk.xsd gemaakt met kkk de naam van het koppelvlak in camel case. Binnen sss_msg_kkk.xsd worden de berichtelementen gedefinieerd. Daarnaast worden zonodig schema's met restrictions op reeds gedefinieerde complexTypes gedefinieerd als sss_kkk_rrr.xsd met _rrr niet aanwezig als het gaat om restrictions op de complexTypes in sss_basis.xsd, rrr stuf voor het geval het gaat om restrictions op StUF-complexTypes en rrr een sectormodel als het gaat om restrictions op complexTypes uit een ander sectormodel.

5 Services

Het definiëren van services is beschreven in document stuf.bindingen.030100.pdf. Hieronder worden de belangrijkste punten uit dit document herhaald.

Een systeem is verplicht voor de hierboven gedefinieerde berichtelementen de volgende services te ondersteunen:

- OntvangAsynchroon
  Deze service dient alle ondersteunde asynchrone berichten te kunnen ontvangen.
- BeantwoordVraag
  Deze service dient alle ondersteunde synchrone vraagberichten te kunnen verwerken
- VerwerkSynchroneKennisgeving
  Deze service dient alle ondersteunde synchrone kennisgeving-, synchronisatieberichten te kunnen verwerken.
- VerstrekSynchronisatieBericht
  Deze service dient alle ondersteunde synchrone vraag-om-synchronisatieberichten te kunnen verwerken.
- VerwerkSynchroonVrijBericht
  Deze service dient alle ondersteunde synchrone vrije berichten te kunnen verwerken.
- VerwerkTriggerbericht
  Deze service dient het triggerbericht te kunnen verwerken, indien dit wordt ondersteund.

Het staat een systeem vrij om naast deze services ook nog andere services voor StUF-berichten te ondersteunen. Hiervoor dient het systeem dan zijn eigen wsdl te definiëren.

De namen voor de wsdl's voor de bovengenoemde services zijn voorgeschreven als eindigend op sss_ontvangAasynchroon.wsdl, sss_beantwoordVraag.wsdl, sss_verwerkSsynchroneKkennisgeving.wsdl,

sss_verstrekSs~~s~~ynchronisatiebericht.wsdl, sss_verwerkSs~~s~~ynchroonV~~v~~rijbericht.wsdl en verwerkTt~~t~~riggerbericht.wsdl.

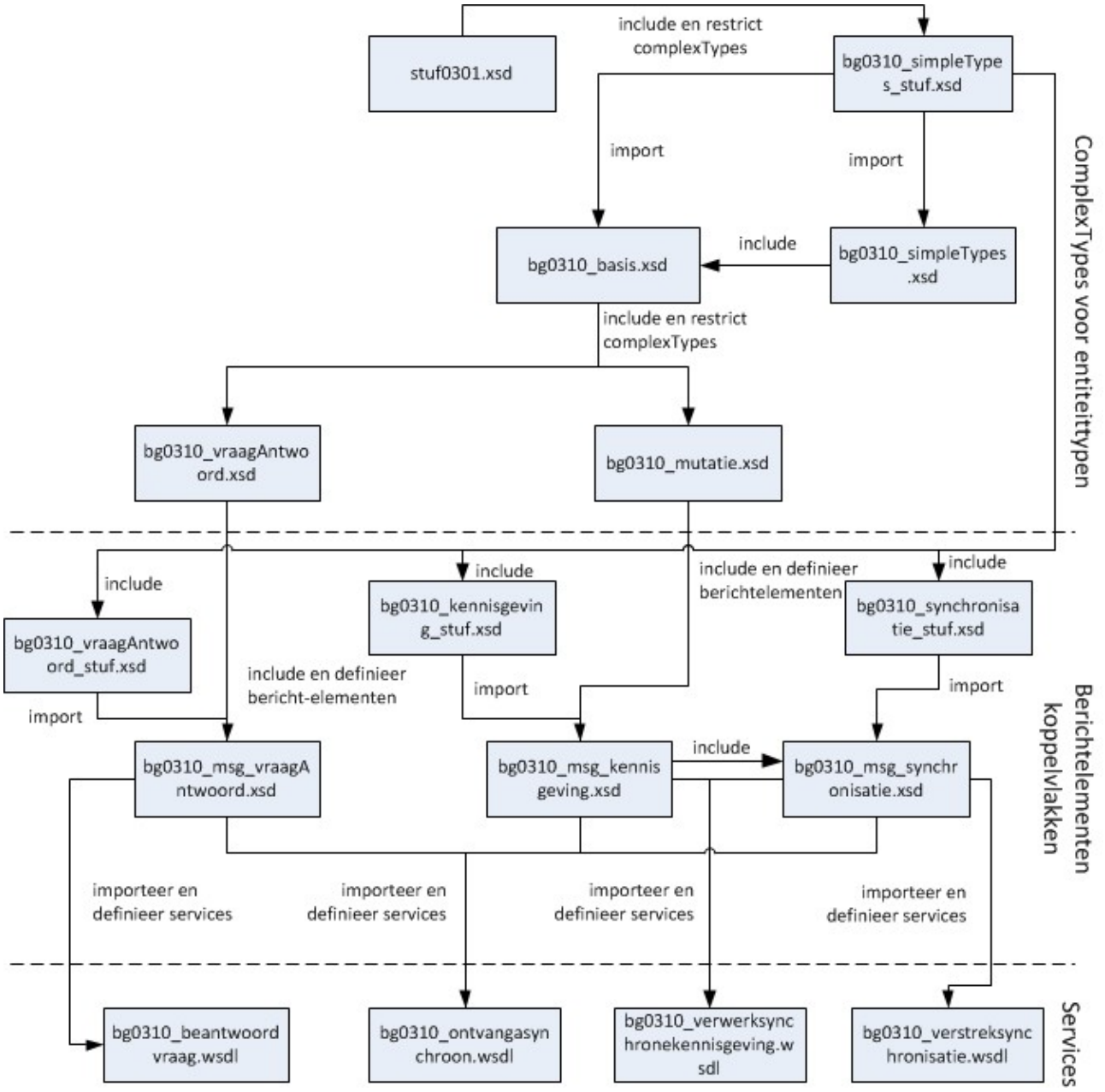
6 Illustratie

Het bovenstaande verhaal is behoorlijk ingewikkeld. Daarom volgt hieronder een schema met een uitwerking van de bovenstaande structuur voor het sectormodel bg0310. De figuur is in drieën gedeeld. Bovenin staan de schema's voor het definiëren van de verschillende complexTypes voor de entiteittypen. De belangrijkste schema's zijn bg0310_basis.xsd met de basis-complexTypes en de schema's met de daarvan afgeleide complexTypes gebruikt binnen de koppelvlakken voor vraag/antwoordberichten, kennisgevingberichten en synchronisatieberichten. De simpleTypes en de daarvan afgeleide complexTypes met de attributes StUF:noValue en StUF:exact worden gedefinieerd in een apart schema bg0310_simpleTypes.xsd. Daarnaast zijn er nog wat restrictions nodig op complexTypes gedefinieerd in StUF. Deze worden gedefinieerd in het schema bg0310_~~basis~~simpleTypes_stuf.xsd.

In het midden staan de schema's voor het definiëren van de berichtelementen voor de drie standaard koppelvlakken voor vraag/antwoordberichten, kennisgevingberichten en synchronisatieberichten. Voor de berichtelementen zijn restrictions nodig op StUF-complexTypes voor stuurgegevens en parametersVraag. Daarnaast importeren de koppelvlak schema's, herkenbaar aan "_msg" in hun naam de schema's met de complexType definities.

Onderin staan de wsdl's voor de in StUF gedefinieerde services. De service ontvangAsynchroon importeert berichtelementen uit alle drie de standaard koppelvlakken en verwerkSynchroneKennisgeving uit de koppelvlakken kennisgeving en synchronisatie. De service beantwoordVraag heeft genoeg aan het koppelvlak vraagAntwoord en verstrekSynchronisatieBericht aan het koppelvlak synchronisatie.

Bij het toevoegen van het BAG-WOZ koppelvlak worden simpelweg de schema's ~~bg0310_bagwoz_stuf.xsd~~ en bg0310_msg_~~bagwoz~~koppelvlakBAG.xsd en bg0310_koppelvlakBAG_stuf.xsd toegevoegd in een folder koppelvlakBAG binnen de folder bg0310. Het is in dit geval niet nodig om ook een schema toe te voegen in het bovenste deel van de figuur, omdat er geen nadere restrictions worden gedefinieerd op de complexTypes in bg0310_mutatie.xsd. De berichten in het nieuwe koppelvlak worden toegevoegd in ~~de desbetreffende~~een wsdl met de naam van het koppelvlak in de naam, in dit geval dus bg0310_koppelvlakBAG_ontvangAsynchroon.wsdl. ~~Bij het toevoegen van een nieuw koppelvlak wijzigen de wsdl's. Dit is geen probleem, omdat de wsdl's toch voor elk systeem moeten worden toegesneden op de functionaliteit van dat systeem.~~Het is aan de gebruikers van het koppelvlak om in de wsdl voor hun systeem de benodigde onderdelen te kopiëren vanuit deze wsdl. Het toevoegen van een nieuw koppelvlak leidt nooit tot wijzigingen in reeds bestaande schema's. Er worden slechts schema's toegevoegd aan het sectormodel.


stuf0301.xsd
include en restrict complexTypes
bg0310_simpleTypes_stuf.xsd
import
import
bg0310_basis.xsd
include
bg0310_simpleTypes.xsd
include en restrict complexTypes
bg0310_vraagAntwoord.xsd
bg0310_mutatie.xsd
ComplexTypes voor entiteittypen
include
include
include en definieer berichtelementen
include
bg0310_vraagAntwoord_stuf.xsd
bg0310_kennisgeving_stuf.xsd
bg0310_synchronisatie_stuf.xsd
include en definieer bericht-elementen
import
import
import
bg0310_msg_vraagAntwoord.xsd
bg0310_msg_kennisgeving.xsd
include
bg0310_msg_synchronisatie.xsd
Berichtelementen koppelvlakken
importeer en definieer services
importeer en definieer services
importeer en definieer services
importeer en definieer services
bg0310_beantwoordvraag.wsdl
bg0310_ontvangasynchroon.wsdl
bg0310_verwerksynchronekennisgeving.wsdl
bg0310_verstreksynchronisatie.wsdl
Services