Standaard Uitwisseling Formaat voor applicaties

StUF 02.04: EGEM Aanbeveling

Inhoudsopgave

Wijzigingshistorie

Versie 02:

- ERR207: De laatste alinea van paragraaf 6.1.3 verplaatst naar het begin van paragraaf 6.1 en daaronder de specificatie voor datum type elementen strakker beschreven.
- Verwijzing naar stuf20.xsd vervangen door stuf204.xsd.
- ERR212: Extra regel toegevoegd aan paragraaf 6.2.4.

1 INLEIDING

De afgelopen jaren is de uitwisseling van gegevens binnen de overheid steeds belangrijker geworden. Hierdoor is er in toenemende mate behoefte aan een standaard uitwisselingsformaat waarmee eenvoudig gegevens kunnen worden uitgewisseld tussen allerhande systemen over allerlei type infrastructuren. Een standaard uitwisselingsformaat voorkomt dat steeds opnieuw maatwerk ontwikkeld moet worden. Bovendien is niet telkens overleg nodig over het realiseren van de gegevensuitwisseling.

In de behoefte aan uitwisseling van persoonsgegevens tussen gemeenten en allerlei organisaties is een aantal jaren geleden al voorzien door middel van de Gemeentelijke Basis Administratie (GBA). Ten behoeve van de uitvoering van de wet-WOZ (Waardering Onroerende Zaken) is de uitwisseling van gegevens tussen gemeenten, waterschappen, de belastingdienst, taxatiebureaus, en het Centraal Bureau voor de Statistiek geregeld. Voor beide vormen van gegevensuitwisseling is een eigen standaard gedefinieerd. Ten behoeve van de uitwisseling van gegevens tussen de verschillende systemen binnen een gemeente is onder auspiciën van de VNG het STandaard Uitwisseling Formaat (StUF) gedefinieerd. De berichten die hierop zijn gebaseerd maken gebruik van TCP/IP en sluiten qua structuur aan op de GBA-berichten.

Inmiddels is de techniek verder voortgeschreden en hebben XML, als notatietaal voor het definiëren van berichten, en SOAP, als protocol voor het versturen van berichten, hun intrede gedaan. In deze vernieuwde StUF 2.04-standaard wordt verder gebouwd op StUF 01.05, dat ontwikkeld is onder auspiciën van de Vereniging van Nederlandse Gemeenten (VNG). Het beschrijft een uitwisselingsformaat met ruwweg dezelfde functionaliteit als StUF 01.05, maar nu gebruikmakend van XML, SOAP en WSDL[1].

De StUF 2.04 standaard schrijft XML voor als notatietaal voor berichten en geeft invulling aan keuzes die daarbij gemaakt worden. Het gebruik van SOAP en WSDL voor het inrichten van het berichtentransport is optioneel. Dit document beschrijft hoe StUF-berichten geïmplementeerd kunnen worden met behulp van de twee laatst genoemde standaarden, maar schrijft dit niet dwingend voor. De StUF-standaard gaat over de inhoud van berichten en niet over de envelop (communicatieprotocol). StUF-berichten zouden bijvoorbeeld ook gebonden kunnen worden aan het ebMS-protocol [EBMS], een SOAP-variant uit de ebXML-familie

Bij het definiëren van StUF 2.04 is veel aandacht besteed aan de definitie van de structuur van het bericht onafhankelijk hoe de uitwisseling plaats vindt. Deze berichtdefinities worden in zogenaamde sectormodellen gedefinieerd. Zo'n sectormodel is gebaseerd op het GFO, dat het domein beschrijft waarover gegevens worden uitgewisseld. Het sectormodel zegt niets over hoe de berichten worden uitgewisseld. Dit is onafhankelijk van sectormodellen beschreven in dit document.

Bij het ontwerp van iets complex als een standaard uitwisselingsformaat spelen een groot aantal afwegingen een rol. In hoofdstuk 2 wordt eerst de globale functionaliteit en de uitgangspunten van StUF beschreven. Dit hoofdstuk is bedoeld om in niet-technische termen de structuur en de functionaliteit te verhelderen. In hoofdstuk 3 wordt beschreven hoe StUF in een gemeentelijke omgeving wordt gepositioneerd en gebruikt. De hoofdstukken 2 en 3 richten zich tot mensen die in StUF geïnteresseerd zijn, maar geen technische achtergrond hebben.
Hoofdstuk 4 beschrijft de syntax voor de gegevens ten behoeve van de adressering en de aansturing van de berichtverwerking. Hoofdstuk 5 beschrijft de syntax voor de uit te wisselen gegevens. Hoofdstuk 6 beschrijft hoe de berichten exact aangemaakt moeten worden. Hoofdstuk 7 beschrijft de uitwisseling van berichten op basis van SOAP/WSDL door middel van een datanetwerk en via alternatieve media als diskettes, tape of CD-ROM. De hoofdstukken 4 tot en met 7 bevatten de volledige specificatie voor de berichtverwerkende software. Deze hoofdstukken richten zich primair op de ontwerpers en bouwers van software.
Het laatste hoofdstuk wil een handreiking bieden aan degenen die op basis van deze standaard berichtdefinities willen maken voor een bepaalde sector.

1.1 Syntax-conventies

De syntax in dit document volgt de volgende conventies:

Italic : in *italic* geformatteerde tekst specificeert een StUF-gerelateeerd begrip.
`Courier`: in `Courier` geformatteerde tekst bevat een gespecificeerd XML-element uit het StUF-berichtenschema.

[1] Zie ook XML, SOAP en WSDL in bijlage Referenties

2 GLOBALE FUNCTIONALITEIT EN OPZET VAN STUF

2.1 Inleiding

Organisaties of organisatieonderdelen registreren geregeld onafhankelijk van elkaar gegevens over dezelfde objecten in de werkelijkheid. Gemeentelijke afdelingen hebben bijvoorbeeld eigen systemen, waarin de basisgegevens over personen, bedrijven en vastgoed meervoudig worden vastgelegd. Het is lastig gegevens gelijk te houden die op verschillende plaatsen worden onderhouden. Er is daarvoor behoefte aan:

1. Het op de hoogte gehouden worden van wijzigingen in gegevens beheerd door andere organisaties of organisatieonderdelen
2. Het kunnen opvragen van gegevens bij anderen

De rijksoverheid onderkent deze problematiek ook gegeven de discussies over een stelsel van authentieke registraties en over het startpakket gegevensverstrekking voor de GBA. Ook bij multinationals speelt het, waar gegevens over klanten, medewerkers, artikelen en dergelijke in verschillende systemen worden vastgelegd. Zij gebruiken hiervoor de term Master-Data Management.

Binnen gemeenten zijn al vanaf midden jaren negentig oplossingen voorhanden. De problematiek speelt daar binnen één niet al te grote organisatie en is daarom relatief eenvoudig op te lossen. Er is bovendien een markt, omdat de oplossing in een kleine 500 gemeenten geïmplementeerd kan worden. De eerste oplossing was gebaseerd op leverancierspecifieke interfaces en werkte daarom slechts voor de applicaties van de specifieke leverancier. Een redelijk aantal gemeenten gebruikt echter applicaties van verschillende leveranciers. Daarom hebben de leveranciers van gemeentelijke systemen samen met de Vereniging van Nederlandse Gemeenten tussen 1996 en 1998 een open standaard ontwikkelt, de zogenaamde StUF-standaard. Met deze standaard kunnen gegevens worden uitgewisseld tussen systemen van verschillende leveranciers.

Dit hoofdstuk analyseert op basis van de ervaringen met de eerste versie van de StUF-standaard de problematiek rond het gelijk houden van in verschillende organisaties of organisatieonderdelen bijgehouden gegevens. Allereerst worden drie belangrijke uitgangspunten voor een oplossing besproken:

1. het gebruik van berichten
2. het gebruik van een domeinonafhankelijke standaard als basis voor de berichtspecificatie
3. De werkelijkheid als semantische basis voor de berichtuitwisseling en niet de database

Het derde uitgangspunt wordt na zijn introductie nog dieper uitgewerkt. Daarnaast wordt nog ingegaan op het omgaan met historische gegevens en de eisen die vanuit gegevensbeheer gesteld worden aan de berichtuitwisseling.

2.2 Uitgangspunten

Het gebruik van berichten

Het gebruik van berichten voor het doorgeven van wijzigingen ligt voor de hand. Het GBA-stelsel werkt ook op deze manier. Op een hoog abstractieniveau kan je stellen dat een afnemer een kennisgevingbericht krijgt, als er gegevens zijn gewijzigd. Op een kennisgevingbericht hoeft een ontvanger niet te reageren. Gegeven de technologische ontwikkelingen rond XML, SOAP en webservices (WSDL) is ook het opvragen van gegevens eenvoudig met berichten te realiseren. De gevraagde gegevens kunnen onmiddellijk worden geleverd (synchroon berichtenverkeer) of ze worden pas na verloop van tijd geleverd (asynchroon berichtenverkeer). De in de inleiding gevraagde functionaliteit kan gerealiseerd worden met asynchrone kennisgevingberichten voor het doorgeven van wijzigingen en met synchrone of asynchrone vraag/antwoordberichten voor het opvragen van gegevens.

Domein-onafhankelijke standaard

In de inleiding hebben we gezien dat het probleem van het op verschillende plaatsen bijhouden van gegevens speelt binnen heel verschillende sectoren en voor een breed scala aan gegevens. De probleemstelling is in wezen ook onafhankelijk van de gegevens waar het om gaat. Een bericht heeft altijd betrekking op een concreet object. Als bijvoorbeeld kan gedacht worden aan het bericht waarmee de geboorte van een persoon wordt doorgegeven aan de belastingdienst of het bericht waarmee een sociale dienst bij de Rijksdienst voor het Wegverkeer (RDW) informatie opvraagt over de auto's die een persoon op zijn naam heeft staan. De twee berichten hierboven hebben betrekking op verschillende sectoren: de GBA en het kentekenregister van de RDW. Het is daarom voor de hand liggend ze in verschillende schema's te definiëren, al was het maar om te voorkomen dat er één allesomvattend schema moet komen.

Bij het definiëren van de berichten voor deze twee sectoren kan prima gebruik gemaakt worden van een template-berichtdefinitie. Deze template-berichtdefinitie beschrijft domeinonafhankelijk de syntax en semantiek van de berichten. Om met deze template-berichtdefinitie een schema voor een sector te definiëren, heb je ook een objectmodel nodig dat de werkelijkheid van een sector modelleert door de relevante objecten en hun eigenschappen te definiëren. Een ontwerper kan op basis van dit objectmodel en de template-berichtdefinitie een schema met de berichtdefinities voor een sector maken, het zogenaamde sectormodel.

Uitgangspunt voor de rest van dit hoofdstuk is het beschrijven van de gewenste functionaliteit op het niveau van een template-berichtdefinitie die samen met een objectmodel voor een sector de basis vormt voor een schema met berichtendefinities. De ambitie van dit hoofdstuk is om zoveel functionaliteit te beschrijven dat voldaan kan worden aan de eisen van gemeenten, de authentieke registraties van de rijksoverheid en het Master-Data Management binnen grote bedrijven.

Berichten hebben allereerst betrekking op de werkelijkheid
Berichten worden normaliter uitgewisseld tussen geautomatiseerde systemen. Een kennisgevingbericht wordt aangemaakt naar aanleiding van een wijziging van gegevens in een database en in een antwoordbericht worden de gevraagde gegevens uit de database van het antwoordende systeem verstuurd. Toch is het voor de betekenis van de berichten beter om niet uit te gaan van inserts, updates en deletes in de databases. De gegevens zitten in die database, omdat een afbeelding van de werkelijkheid nodig is. De werkelijkheid en niet de database is dus het semantisch relevante niveau, want aan een wijziging van de database zal een gebeurtenis in de werkelijkheid vooraf gaan. Uiteindelijk is die gebeurtenis in de werkelijkheid ook de aanleiding voor het verzenden van een kennisgeving.

De verdere analyse zal er daarom vanuit gaan dat berichten over gebeurtenissen (kennisgevingen) of toestanden (vraag/antwoord) in de werkelijkheid worden uitgewisseld tussen organisaties of onderdelen daarvan. Kennisgevingsberichten bevatten gegevens die de eraan ten grondslag liggende gebeurtenis beschrijven. Antwoordberichten bevatten de afbeelding van de wereld van de bevraagde organisatie. Voor de verdere analyse maakt het feitelijk niet uit of het gaat om een uitwisseling van documenten tussen mensen of van geautomatiseerde berichten tussen systemen. In de nu volgende paragraaf ‘Relatie berichten tot de werkelijkheid’ worden de consequenties van dit uitgangspunt in meer detail beschreven.

2.3 Relatie berichten tot de werkelijkheid

Vraag/antwoord berichten
Bij vraag/antwoord berichten is de relatie van de berichten tot de werkelijkheid eenvoudig. Het vraagbericht specificeert wat de vragende partij wil weten over de werkelijkheid. In het antwoordbericht vertelt de antwoordende partij de hem bekende toestand in de werkelijkheid aan de vragende partij.

Soorten kennisgevingberichten
Bij kennisgevingberichten is de relatie tot de werkelijkheid complexer. Een eerste vraag is op hoeveel objecten een kennisgevingbericht betrekking heeft. Een verstandig uitgangspunt met het oog op reductie van complexiteit is: een kennisgevingbericht heeft betrekking op precies één object in de werkelijkheid. De volgende veranderingen in de werkelijkheid zijn dan aanleiding voor een kennisgeving:

1. Een object ontstaat, bijvoorbeeld de geboorte van een kind
2. Een object krijgt andere eigenschappen, bijvoorbeeld een nieuw telefoonnummer
3. Een object houdt op te bestaan, bijvoorbeeld het overlijden van een persoon

Het lijkt voor de hand te liggen deze gebeurtenissen te onderscheiden binnen de kennisgevingberichten. Echter, als je de relatie van de zendende partij tot de werkelijkheid en de problematiek rond gegevensmanagement meeneemt in de analyse, dan kom je tot een andere karakterisering van de relatie van kennisgevingberichten tot de werkelijkheid:

1. Een object is relevant geworden voor de zendende partij, bijvoorbeeld de vestiging van een persoon in de gemeente of de geboorte van een persoon. De zendende partij zal het object in zijn registratie vastleggen en communiceert dit door middel van een toevoegkennisgeving.
2. Een object krijgt in de werkelijkheid andere eigenschappen. Er is met het object iets gebeurd in de werkelijkheid: een gebeurtenis of event. De zendende partij zal de geregistreerde gegevens aanpassen en communiceert dit door middel van een wijzigkennisgeving.
3. De gegevens die de zendende partij heeft over een object bleken onjuist en ze zijn gecorrigeerd. Er is met het object in de werkelijkheid niets gebeurd. De zendende partij zal de geregistreerde gegevens corrigeren en communiceert dit door middel van een correctiekennisgeving.

4. Een object is niet langer relevant voor de zendende partij, bijvoorbeeld omdat een kind niet langer leerplichtig is. De zendende partij zal het object in zijn registratie verwijderen en communiceert dit door middel van een verwijderkennisgeving.

Het relevant en irrelevant worden van een object is belangrijke informatie ten behoeve van het gegevensmanagement, want een zendende partij stelt alleen belang in voor hem relevante objecten. Het ontstaan en ophouden te bestaan van objecten staat hier los van. Een overleden persoon kan bijvoorbeeld nog een tijd lang relevant blijven, ook al kunnen zijn eigenschappen niet meer wijzigen. Correcties van gegevens kunnen nog forse consequenties hebben, denk aan de gevolgen voor de verdeling van de erfenis van het na het overlijden registreren van de erkenning van een kind. Het ontstaan en ophouden te bestaan kan in kennisgevingberichten eenvoudig worden gecommuniceerd als de gebeurtenis die ten grondslag ligt aan het bericht.

Speciale waarden van eigenschappen
Bij het definiëren van een waardebereik voor een eigenschap is het van belang te realiseren dat een object twee typen eigenschappen heeft:
1. Eigenschappen die een waarde hebben zodra een object bestaat, bijvoorbeeld de geboortedatum
2. Eigenschappen die niet altijd een waarde hoeven te hebben, bijvoorbeeld de overlijdensdatum of een gironummer.

In het waardebereik van een eigenschap dienen dus naast de gebruikelijke waarden onderkend te worden de waarden ‘Onbekend’ en ‘Heeft geen waarde’. De waarde ‘Heeft geen waarde’ kan uiteraard alleen maar voorkomen bij eigenschappen die niet altijd een waarde hoeven te hebben. Ook deze twee waarden dienen in berichten gecommuniceerd te kunnen worden. Het maakt qua betekenis een groot verschil of je in een bericht opneemt dat de overlijdensdatum onbekend is (bijvoorbeeld in het geval van een vermissing) of dat je meent te weten dat een persoon niet overleden is (overlijdensdatum heeft als waarde ‘Heeft geen waarde’).

Attributen en relaties als eigenschappen
Tot nu toe hebben we losjes gesproken over eigenschappen van objecten. De theorie van datamodellering heeft ons echter geleerd dat er twee soorten eigenschappen zijn:
1. Attributen
   Een attribuut is een kenmerk van een object dat alleen afhankelijk is van het object zelf, bijvoorbeeld de geslachtsnaam of de geboortedatum van een persoon
2. Relaties
   Een relatie is een kenmerk van een object die afhankelijk is van een ander object, bijvoorbeeld de ouder van een persoon.

Of een eigenschap wordt gezien als een relatie is arbitrair. De ontwerper van het objectmodel van de werkelijkheid hoeft namelijk niet het objecttype te onderkennen waarnaar een relatie ligt. Het adres van een persoon kan als een verzameling attributen bij het objecttype persoon worden gemodelleerd en als een relatie van het objecttype persoon naar het objecttype adres. De GBA is hierin nog verder gegaan. Ook al wordt het objecttype persoon onderkend, toch zijn in de GBA de ouder- en kindgegevens een verzameling attributen bij een persoon. De reden hiervoor is dat in de GBA in wezen de oorspronkelijke papieren persoonskaarten zijn gemodelleerd en geen personen.
Net zo arbitrair is de vraag of een relatie niet beter gemodelleerd kan worden als een afzonderlijk objecttype. Een huwelijk kan gezien worden als een relatie tussen twee personen met een aantal eigenschappen en als een afzonderlijk objecttype met twee relaties naar de huwelijkspartners. De ontwerper van het objectmodel is verantwoordelijk voor dit soort beslissingen.

Een attribuut kan alleen maar van waarde veranderen. Bij relaties zijn er meer mogelijkheden:
1. Het toevoegen van een relatie
   Een relatie is relevant geworden voor de zender (vaak zal de relatie ook zojuist ontstaan zijn, maar dit hoeft niet)
2. Het wijzigen van eigenschappen van de relatie
   De eigenschappen kunnen zowel wijzigen naar aanleiding van een gebeurtenis in de werkelijkheid als naar aanleiding van de vaststelling dat er verkeerde gegevens waren vastgelegd (correctie). Hieronder valt ook het corrigeren van het beëindigen van een relatie
3. Het beëindigen van een relatie
   Een relatie bestaat niet langer. Ook hier kan het zowel gaan om een gebeurtenis in de werkelijkheid als een correctie.

4. Het vervangen van een relatie
   Een relatie wordt beëindigd en onmiddellijk vervangen door een andere relatie. Ook hier kan het zowel gaan om een gebeurtenis in de werkelijkheid als een correctie.
5. Het niet meer relevant zijn van een relatie
   De relatie is niet langer relevant voor de zender. De relatie kan nog wel in de werkelijkheid bestaan.
6. Het corrigeren van de toevoeging van een relatie
   De relatie heeft nooit bestaan bij het object.

Het communiceren over relaties is dus complex. Al deze mogelijkheden dienen semantisch en syntactisch beschreven te worden in een template-berichtdefinitie. Zo niet, dan is geen adequate informatie-uitwisseling mogelijk.

2.4 Historische gegevens

De eigenschappen van een object kunnen in de tijd variëren. Een persoon verhuist bijvoorbeeld een aantal keren in zijn leven. Bij het denken over historie moeten we onderscheid maken tussen een historisch object en een historisch gegeven. Een historisch object is een object dat in de werkelijkheid niet meer bestaat maar nog wel van belang is. Een historisch gegeven is een gegeven van een object dat niet meer overeenkomt met de werkelijkheid. Een historisch object heeft dus actuele gegevens, als hun waarden overeenstemmen met de laatste (actuele) waarden voordat het object ophield te bestaan.

Er zijn bij gegevensuitwisseling voorzieningen nodig voor het uitwisselen van historische gegevens. Hierbij spelen twee problemen:
1. Is de semantiek van de standaard voldoende rijk om historische gegevens uit te wisselen?
2. Zijn de gebruikers van de standaard bereid om de extra complexiteit ten gevolge van historische gegevens te implementeren in de systemen die van de standaard gebruik maken?

In een template-berichtdefinitie ligt het voor de hand voorzieningen te treffen om historische gegevens uit te wisselen. Tegelijkertijd wil je niet aan alle gebruikers de eis opleggen al deze voorzieningen te implementeren. Het moet daarom mogelijk zijn de extra functionaliteit rond historische gegevens te negeren en toch zinvol berichten te verwerken. In StUF 2.04 is er daarom voor gekozen om het wijzigen van historische gegevens niet te ondersteunen.

Wanneer een bericht het gevolg is van een gebeurtenis, dan bepaalt die gebeurtenis de einddatum geldigheid van de oude waarde en de begindatum geldigheid van de nieuwe waarde. Bij een correctie van een waarde ligt dit minder eenvoudig, want er zijn drie soorten correcties mogelijk:
1. Het corrigeren van een waarde zonder dat het tijdvakgeldigheid gecorrigeerd hoeft te worden. Er is bijvoorbeeld ten gevolge van een tikfout een onjuiste waarde gecommuniceerd.
2. Het alleen corrigeren van het tijdvak geldigheid van een waarde. Er is een verkeerd tijdvak geldigheid gecommuniceerd, maar de nieuwe waarde is wel correct.
3. Het zowel corrigeren van de waarde als van het tijdvak geldigheid. Het meest voorkomende geval is hier het ten onrechte wijzigen van een waarde en het in een correctie terugdraaien van die wijziging. Het nieuwe tijdvak geldigheid heeft daardoor nooit bestaan. Het complexere geval dat er zowel een verkeerde waarde als een verkeerd tijdvak geldigheid is gecommuniceerd is in feite een combinatie van (1) en (2) en kan ook het beste zo worden opgelost.

In een template-berichtdefinitie dient in elk geval de onder (1) genoemde situatie en het corrigeren van een onterechte wijziging ondersteund te worden. Voorwaarde voor het ook ondersteunen van correcties op het tijdvak geldigheid is dat systemen die niet in historie geïnteresseerd zijn deze berichten eenvoudig kunnen negeren.

De historie van attribuutwaarden bij een object kan worden gespecificeerd met behulp van een begin- en einddatum geldigheid voor een waarde. Dit kan op twee manieren worden geïmplementeerd:
1. Door per waarde een begin- en einddatum geldigheid in de berichten op te nemen
2. Door voor alle attribuutwaarden in een object op objectniveau een en dezelfde begin- en einddatum geldigheid op te nemen.

Bij de eerste keuze is het mogelijk om in één bericht attribuutwaarden meervoudig op te nemen met elk een eigen tijdvak geldigheid. Het ene attribuut komt één keer voor in het bericht, een andere vijf keer en weer een andere twee keer. Bij de tweede keuze worden meerdere voorkomens van het object in het bericht opgenomen met elk een eigen tijdvak geldigheid. Alle attribuutwaarden zijn geldig gedurende dat tijdvak geldigheid. Als een bericht meerdere voorkomens bevat, dan is er precies één met de actuele waarden. De andere voorkomens bevatten

historische waarden. De tweede keuze leidt tot een simpeler berichtverwerking, omdat de actuele gegevens eenvoudig te vinden zijn. De tweede keuze sluit ook beter aan bij de wijze waarop in veel databases met historische gegevens wordt omgegaan.

Bij relaties is niet alleen van belang wat het tijdvak geldigheid is van eigenschappen van een relatie, maar ook het bestaanstijdvak gedurende welke een relatie bestaat. Het ontstaan en beëindigen van een relatie zijn op te vatten als wijzigingen in het object. Zowel de begindatum relatie als de einddatum relatie markeren dus een tijdvak geldigheid voor het object. Relaties kunnen op verschillende manieren wijzigen en ze hebben hun eigen bestaanstijdvak en tijdvak geldigheid. De eenvoudigste oplossing is om wijzigingen in relaties onafhankelijk van de wijzigingsdatum voor de attribuutwaarden in kennisgevingen op te nemen. Gegeven dit feit en gegeven het grote aantal manieren waarop een relatie bij een object kan wijzigen, is het verstandig in een template-berichtdefinitie het bestaanstijdvak en de gewenste functionaliteit eromheen te definiëren.

In antwoordberichten kan historie het beste worden gecommuniceerd door meerdere voorkomens van een object op te nemen met elk een eigen tijdvak geldigheid. Daarnaast dient de steller van de vraag de mogelijkheid te hebben om te specificeren of hij al dan niet historische gegevens wil ontvangen. Een systeem dat niet geïnteresseerd is in historische gegevens vraagt er niet om en krijgt altijd antwoordberichten met slechts één actueel voorkomen van een object.

Voor kennisgevingberichten lijkt een andere keuze eenvoudiger. Neem in kennisgevingberichten bij wijzigingen in de attribuutwaarden één voorkomen van het object op met de waarden geldig tot de wijzigingsdatum (het ‘oude’ voorkomen) en één voorkomen met de waarden geldig vanaf de wijzigingsdatum (het ‘nieuwe’ voorkomen). In het ‘oude’ voorkomen zit zonodig ook de ‘oude’ situatie voor een relatie en in het ‘nieuwe’ voorkomen de ‘nieuwe’ situatie. Het bestaanstijdvak en het tijdvak geldigheid relatie specificeren samen de benodigde informatie ten behoeve van de opbouw van historie. Het tijdvak geldigheid op objectniveau is hiervoor niet relevant. Dit heeft uitsluitend betrekking op de attribuutwaarden.

Voorwaarde voor het kunnen negeren van de informatie over de historie is dat er nooit kennisgevingen worden verzonden met wijzigingen in historische gegevens. Bij het opbouwen van historie via kennisgevingen moet dus altijd de oudste situatie het eerst worden gezonden, daarna de één-na-oudste situatie enzovoorts, totdat de actuele situatie is bereikt. Een systeem dat historie negeert, kan dan al deze berichten gewoon verwerken en eindigt dan met de actuele gegevens.

2.5 Gegevensmanagement

Vanuit het oogpunt van gegevensmanagement zijn twee problemen belangrijk:
1. Hoe worden objecten geïdentificeerd?
2. Wat verwacht de zender van de ontvanger aangaande de verwerking?

Objectidentificatie
Een persoon wordt kan worden geïdentificeerd aan de hand van een groot aantal gegevens. Sommige gegevens zijn op zich in principe al identificerend zoals het A-nummer en het SoFi-nummer en andere gegevens zijn alleen in combinatie met andere gegevens identificerend, zoals geboortedatum of adres in combinatie met geslachtsnaam en voorletters. Bij een tweeling op hetzelfde adres met gelijke voorletters (grapje van de ouders) zijn ook de voornamen nodig om ze te identificeren buiten het A- en SoFi-nummer om. Een template-berichtdefinitie dient voorzieningen te bevatten, waarmee de ontwerpers van berichten eenvoudig kunnen specificeren welke gegevens in elk geval ten behoeve van identificatie in een bericht horen te zitten. Je zou deze minimaal vereiste set gegevens de kerngegevens van een objecttype kunnen noemen en kunnen eisen dat alle kerngegevens verplicht in kennisgevingen moeten worden opgenomen. Voor vraagberichten hoef je op dit vlak niets te specificeren, want de vragensteller kan zelf aangeven welke gegevens hij wil ontvangen.

De berichten worden over het algemeen uitgewisseld tussen geautomatiseerde systemen en deze systemen identificeren om technische redenen objecten veelal met een eigen sleutel. Deze technische sleutels komen niet voor in het objectmodel van de werkelijkheid, maar het is wel handig om de sleutel van een object in het zendende systeem (`sleutelZendendSysteem`) en in het ontvangende systeem (`sleutelOntvangendSysteem`) te kunnen communiceren. In veel gevallen wordt de berichtuitwisseling ingericht met een centraal systeem dat zorgt voor de distributie en het gegevensmanagement. Ook dit systeem kan objecten identificeren met een eigen sleutel (`sleutelGegevensbeheer`). Het is handig om ook deze sleutel in de berichten te kunnen communiceren. Bij het routeren van berichten kan dit centrale systeem, dan zijn eigen sleutel in het bericht opnemen en het bericht doorsturen met als zender nog steeds het systeem dat het ter distributie heeft aangeboden. Voor de ontvanger blijft dan duidelijk welk systeem een bericht heeft geïnitieerd.

Verwerking
Het is voor een zender plezierig om in een bericht te kunnen aangeven, wat voor soort verwerking van de ontvanger verwacht wordt. Er zijn hier twee mogelijkheden:

1. Het bericht is puur informatief bedoeld.
2. De zender verwacht dat de ontvanger de gegevens uit het bericht overneemt (de exacte wijze van overname is uiteraard afhankelijk van het type bericht).

In het tweede geval verwacht de zender dat hij bij het later bevragen van de ontvanger in het antwoord de gegevens krijgt aangeleverd conform de nieuwe waarden in het bericht. In het eerste geval heeft de zender geen enkele verwachting betreffende de inhoud van het antwoord. Deze functionaliteit is in het bijzonder van belang voor een centraal gegevensmanagement systeem dat het berichtenverkeer tussen de aangesloten systemen controleert.

2.6 Tot slot

In het voorgaande hebben de belangrijkste uitgangspunten en functionaliteit de revue gepasseerd voor een template-berichtdefinitie die de basis vormt voor het definiëren van berichten waarmee organisaties hun afbeeldingen van de werkelijkheid gelijk kunnen houden. Deze uitgangspunten zijn voor het eerst uitgewerkt in de StUF 01.02-standaard uitgaande van een berichtsyntax afgeleid van het GBA-stelsel. Inmiddels is XML de basis geworden voor berichtenverkeer. De rest van dit document beschrijft de implementatie van deze uitgangspunten in een op XML-gebaseerde standaard.

3 GEBRUIK VAN STUF IN HET GEMEENTELIJK DOMEIN

In het voorgaande hoofdstuk zijn de algemene uitgangspunten geschetst waarop StUF is gebaseerd. In dit hoofdstuk zal een vertaling worden gemaakt naar het gebruik van StUF in het gemeentelijke domein. De problematiek rondom het meervoudig registreren en onderhouden van gegevens speelt ook binnen de gemeentelijke organisatie. In dit hoofdstuk wordt meer informatie gegeven hoe StUF in deze omgeving te gebruiken.

3.1 Relatie StUF en GFO

Bij de uitwisseling van gegevens kunnen twee belangrijke aspecten worden onderscheiden; **wat** voor gegevens worden er uitgewisseld en **hoe** wordt dit gerealiseerd.

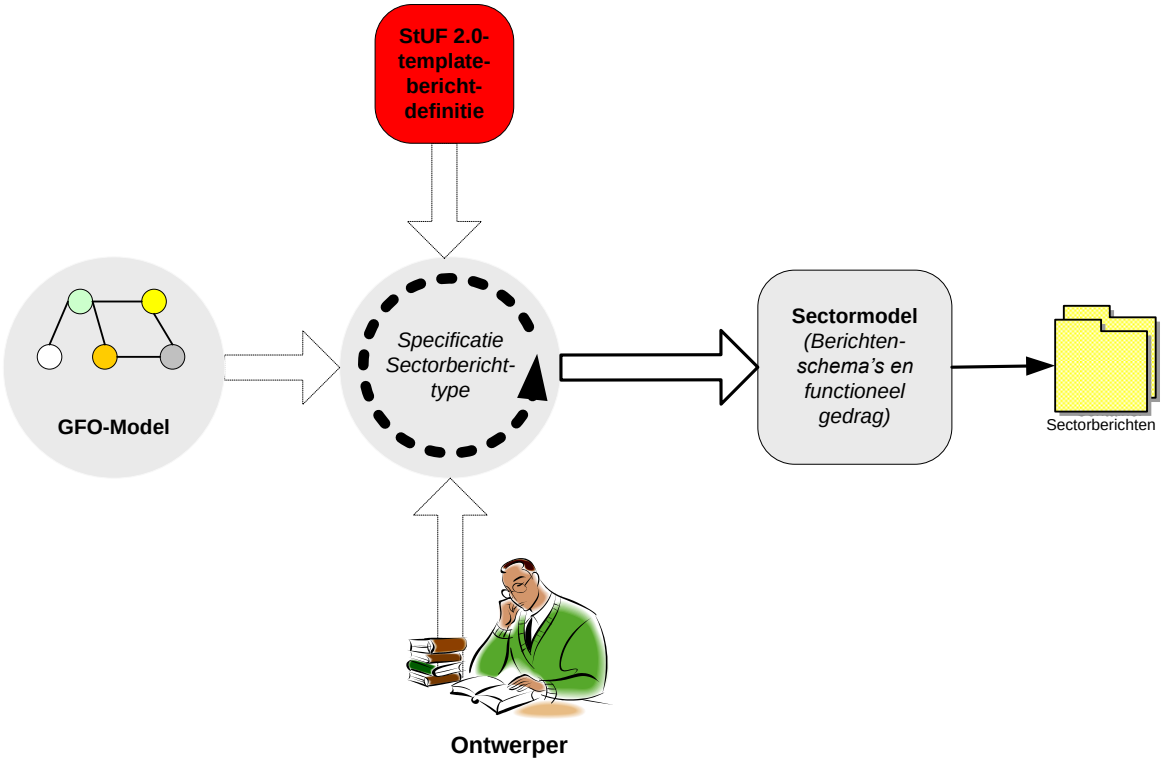
Gegevens worden binnen het gemeentelijke domein gedefinieerd in zogenaamde Gemeentelijke Functionele Ontwerpen (GFO). Deze GFO's beschrijven voor een zeker gemeentelijk beleidsdomein de relevante objecten en hun relaties. Voor een aantal GFO's is een ERD opgesteld met daarin een specificatie van de objecten en eigenschappen (incl. relaties). Het belangrijkste doel van deze specificatie is om de semantiek van deze objecten en hun eigenschappen formeel te specificeren. Tussen mensen is deze semantiek impliciet aanwezig, maar bij de gegevensuitwisseling tussen systemen zal deze notie expliciet moeten worden gemaakt. Bij het uitwisselen van gegevens is het belangrijk dat er gemeenschappelijke overeenstemming is over welke informatie wordt uitgewisseld en wat de betekenis is (de semantiek van de gegevens).

Bij het *hoe* de gegevens worden uitgewisseld ligt de nadruk meer op de structuur ofwel de syntax van de berichten. StUF definieert een template-berichtdefinitie op basis waarvan berichten worden gespecificeerd. Deze template-berichtdefinitie schrijft de structuur van het bericht voor. De berichten worden uitgewisseld tussen een zendende en een ontvangende applicatie, respectievelijk de StUF-zender en StUF-ontvanger. In de volgende paragraaf zullen we nader ingaan op de berichtuitwisseling.

In het GFO-model worden relevante objecten binnen een beschouwd domein geïdentificeerd en de eigenschappen van het object gedefinieerd. Verder worden er tussen objecten relevante relaties gedefinieerd. Op deze manier wordt de beschouwde werkelijkheid (probleem domein) uitgedrukt in objecten, eigenschappen en relaties. De eigenschappen van de verschillende objecten kunnen vervolgens op attributen in het object worden afgebeeld. Op het moment dat dit model door verschillende systemen wordt gebruikt als basis voor de gegevensuitwisseling, is er sprake van een gemeenschappelijke notie van de (geabstraheerde) werkelijkheid. Over objecten worden in de berichten gegevens uitgewisseld en berichten bevatten waarden van de attributen van een object.

Op dit moment zijn de GFO-modellen primair bedoeld voor menselijke communicatie tussen de domeinexpert en de berichtenontwerper. Zij definiëren voor een bepaald domein een abstractie van die werkelijkheid, uitgedrukt in een natuurlijke taal.

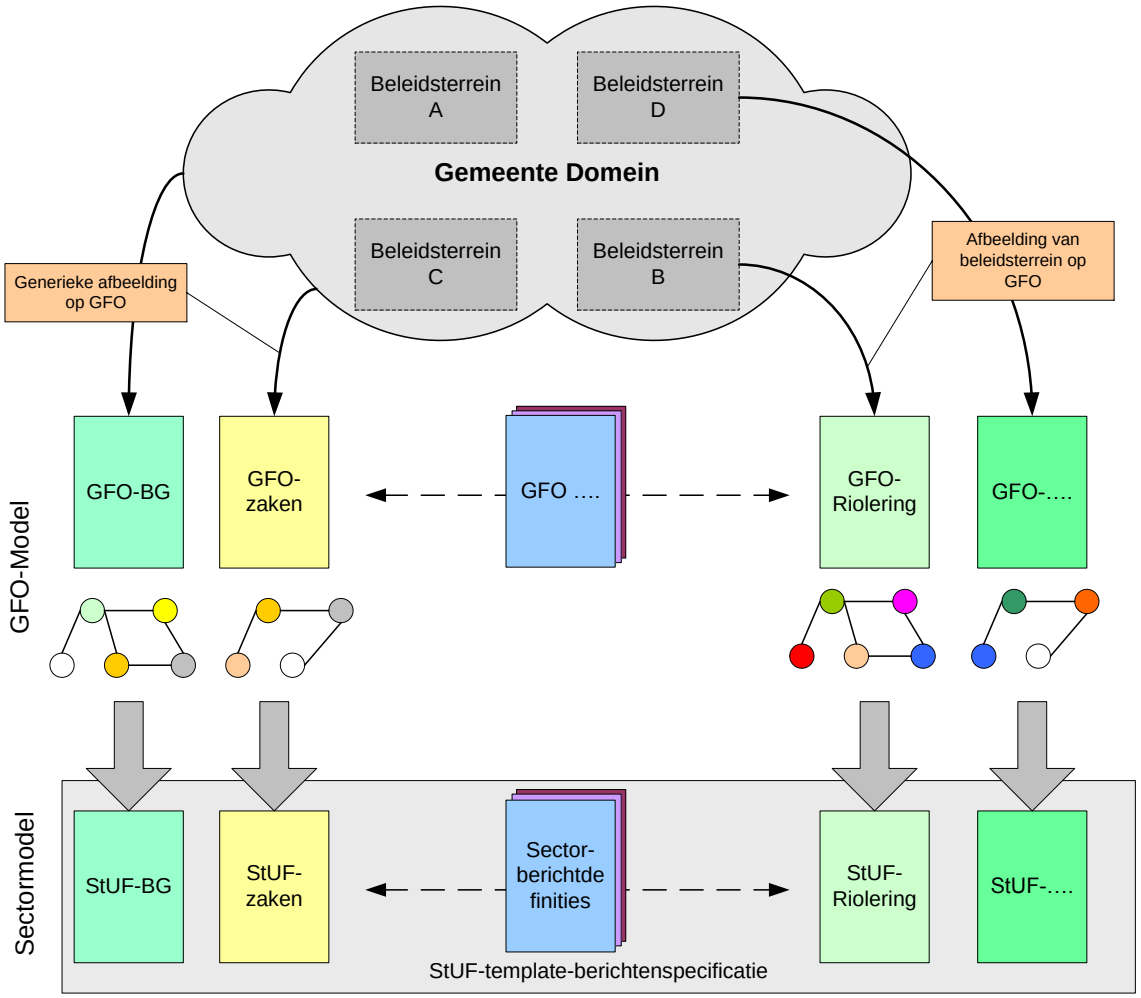
Naast de GFO's is er de StUF 2.04-standaard, die een template-berichtdefinitie specificeert voor berichtenschema's. Deze template-berichtdefinitie vormt mede de basis voor de sectorspecifieke berichtenschema's, die de concreet uit te wisselen berichten voor een bepaalde sector definiëren. Deze sectorberichten worden in de uitwisseling gebruikt en bevatten de waarden van de eigenschappen van de objecten uit het GFO-model. De sectorspecifieke berichtenschema's zijn de uitkomst van een compositieproces waarbij het genoemde GFO-model, de StUF-template-berichtdefinitie en ontwerpbeslissingen van de berichtenontwerper de input vormen. Deze sectorspecifieke berichtenschema's vormen de basis voor de sectorberichten, de concrete instanties die tussen StUF-zender en StUF-ontvanger worden uitgewisseld. Samengevat kunnen we dus drie niveaus berichtspecificatie identificeren: De StUF template-berichtdefinitie, de sectorspecifieke berichtenschema's en de sectorberichten.



Figuur 1 Proces om te komen tot specificatie voor sectorberichttype

Deze standaard specificeert de StUF- template-berichtdefinitie. Enkel in voorbeelden zal gebruik gemaakt worden van het sectormodel en/of sectorberichten. Deze standaard specificeert dus **niet** de sectorberichtenschema's.

In Figuur 2 is het beeld van objecten en berichttypen op een schematische manier weergegeven. De objectdefinities en hun relaties zijn gedefinieerd in de GFO's. In de GFO's is onderscheid gemaakt in GFO's die specifiek voor een gemeentelijk beleidsterrein zijn opgesteld en meer generiek te gebruiken GFO's (bijvoorbeeld GFO-BG of GFO-Zaken). De generieke GFO's definiëren gemeenschappelijke objecten die in meerdere beleidsspecifieke GFO's worden gebruikt. Hiermee wordt getracht de gemeenschappelijke objecten in de GFO's op een eenduidige manier te definiëren Het objectmodel in een GFO wordt met behulp van een ERD-diagram uitgedrukt. In dit ERD-diagram worden de eigenschappen van objecten en de relatie tussen de objecten gespecificeerd. Het GFO is vervolgens de basis voor de definitie van het schema met de sectorberichten (bijvoorbeeld StUF-BG, StUF-zaken).



Figuur 2 Relatie GFO en StUF-berichten

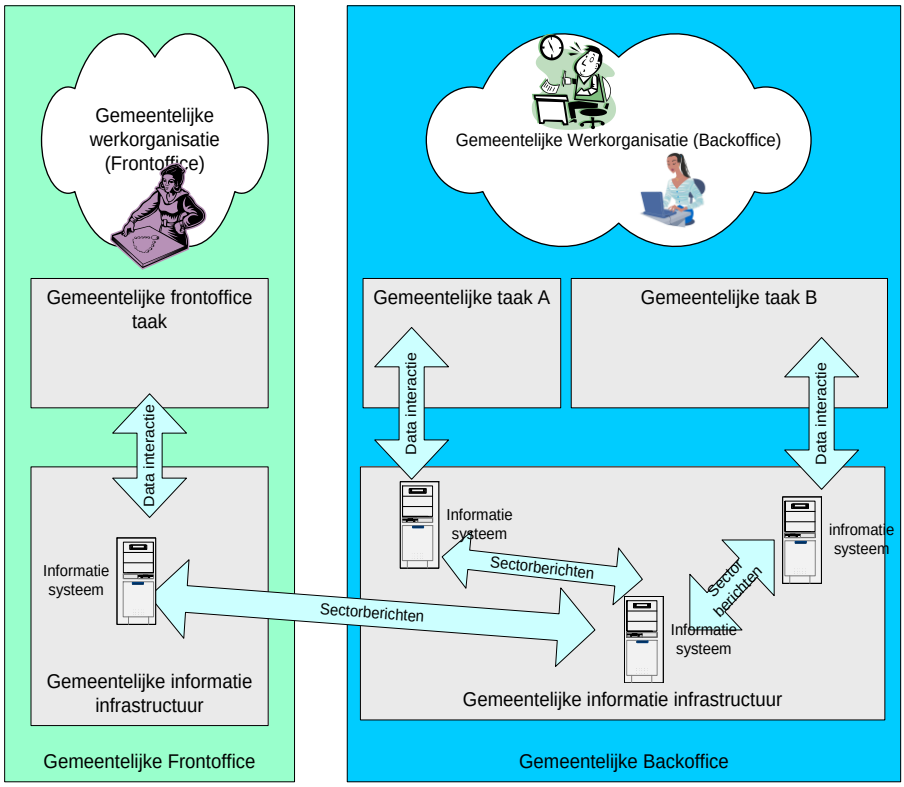
3.2 Gebruik van StUF-sectorberichten

In dit hoofdstuk wordt de positionering van StUF en de daarop gebaseerde StUF-sectorberichten vanuit verschillende perspectieven beschreven. Het doel hiervan is om de rol van de StUF-sectorberichten te definiëren die tussen gemeentelijke systemen worden uitgewisseld.

De door StUF gedefinieerde sectorberichten zijn er op gericht om verschillende informatiesystemen te kunnen voorzien van gegevens (kennisgeving) en om deze gegevens te kunnen opvragen (vraag & antwoord). Door het gebruik van generieke berichttypen kunnen verschillende informatiesystemen “loosely coupled” met elkaar worden verbonden met als resultaat dat ze beschikken over eenduidige gegevens.

Figuur 3 geeft een schematisch overzicht van een gemeentelijke organisatie die kan worden opgedeeld in een backoffice en een frontoffice. Hoewel er tegenwoordig volop wordt gesproken over een midoffice, laten we deze hier nog buiten beschouwing. De functionaliteit die binnen deze eenheid wordt gedefinieerd is nog onvoldoende gespecificeerd.
Binnen de gemeentelijke organisatie kunnen we drie relevante abstracties onderscheiden:

- Gemeentelijke werkorganisatie (w.o. procesorganisatie)
- Gemeentelijke taken,
- Gemeentelijke informatie-infrastructuur.

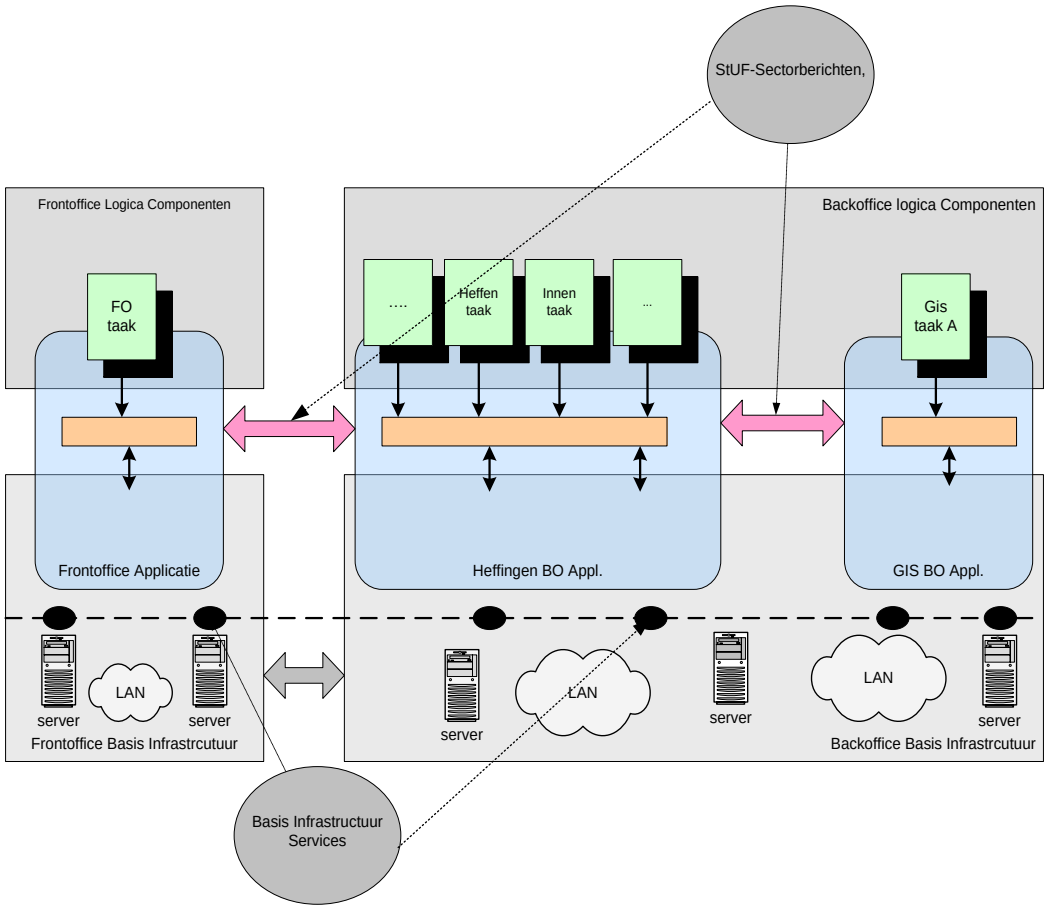


Figuur 3 Gebruik van sectorberichten in gemeentelijk organisatie

De gemeentelijke werkorganisatie is primair verantwoordelijk voor het kunnen aanspreken van gemeentelijke taken en de uitvoering en organisatie van die taken. Hierbij kan gedacht worden aan medewerkers die kunnen zorgen voor de uitgifte van een paspoort, het registreren van een nieuwe wereldburger of het afgeven van een bouwvergunning. Een aantal van die taken die binnen een gemeente worden uitgevoerd worden via een service aangeboden aan potentiële gebruikers (serviceafnemers). Als voorbeeld kan gedacht worden aan de service “aangifte geboorte”. Deze service registreert de benodigde gegevens en zal dan de gemeentelijke taak “registreren nieuwe wereldburger” initiëren. Deze taak is verantwoordelijk voor het checken van de gegevens en het informeren van de verschillende relevante informatiesystemen.

Tijdens het uitvoeren van deze taak wordt er gebruik gemaakt van de diensten die een informatiesysteem biedt. Naast het beheer van gegevens bieden deze systemen functies waarmee de gegevens kunnen worden gecreëerd en bewerkt. Hierbij kan ook gedacht worden aan het uitvoeren van controleacties op de gegevens. De services die deze systemen bieden ondersteunen de uitvoering van gemeentelijke taken. De informatiesystemen bevinden zich in de gemeentelijke informatie-infrastructuur.

Op dit moment zijn de verschillende informatiesystemen gebonden aan specifieke applicaties. Door de informatie tussen applicaties uit te wisselen kan worden gezorgd voor eenduidigheid van gegevens. De sectorberichten worden gebruikt voor de verspreiding van deze gegevens tussen verschillende applicaties.



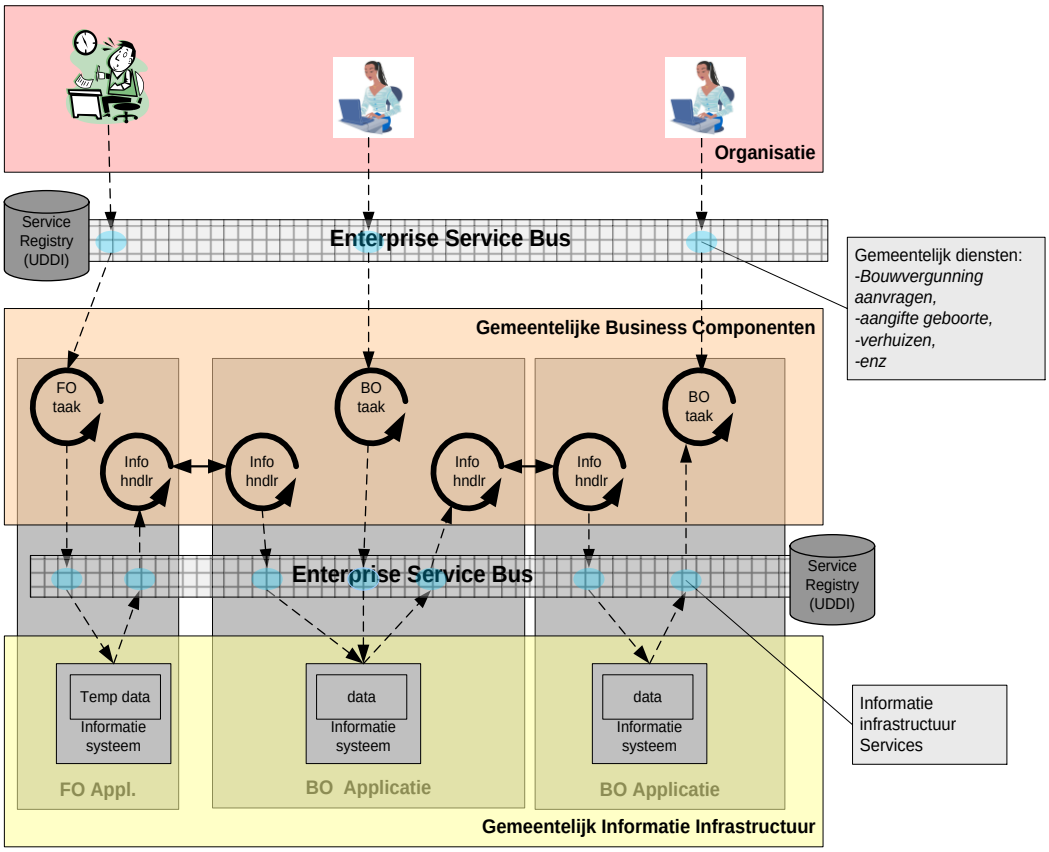
Figuur 4 Voorbeeld van StUF-sectorberichten in relatie tot gemeentelijke applicaties

In Figuur 4 is in groter detail aangegeven hoe de gemeentelijke taken worden ondersteund door applicaties. Met een applicatie wordt hier bedoeld een of meerdere programma's die een bepaalde taak uitvoeren of ondersteunen. De applicaties die we hier geschetst hebben bestaan uit informatiesystemen en gemeentelijke modules (met gemeentelijke logica). Een informatiesysteem is binnen een applicatie de gegevensverstrekker en zorgt voor de toegang en het beheer van de gegevens. Een gemeentelijke module kan worden beschouwd als een component die de uitvoering van een gemeentelijke taak implementeert en daarin door informatiesyste(e)m(en) wordt ondersteund. Een gemeentelijke module is een softwarecomponent die een gemeentetaak volledig of deels realiseert.

In bovenstaande figuur is via de horizontale pijl tussen de applicaties aangegeven waar de sectorberichten worden uitgewisseld. Deze sectorberichten worden uitgewisseld tussen een zendende en ontvangende applicatie in de frontoffice en backoffice als ook binnen de backoffice zelf.

Applicaties hebben functionaliteit die er voor zorgt dat de sectorberichten kunnen worden ontvangen en verstuurd en die er voor zorgt dat de gegevens kunnen worden bewerkt voordat ze in het eigen informatiesysteem worden opgenomen. Deze bewerking is nodig omdat de syntax van de uitgewisselde gegevens anders is (kan zijn) dan de syntax die wordt gebruikt in het eigen informatiesysteem. Immers het is een gemeenschappelijke uitwisselingsdefinitie en die kan verschillen met de interne definitie. Er zal dus een mapping moeten worden gemaakt tussen de gegevens van de gemeenschappelijke- en de interne informatiedefinitie.

Als voorbeeld zijn in Figuur 4 in de backoffice twee applicaties geschetst, een GIS- en een heffingenapplicatie. Beide applicaties maken gebruik van eigen informatiesystemen, alleen de basisinfrastructuur wordt gedeeld. De sectorberichten kunnen worden uitgewisseld tussen de verschillende applicaties in zowel front- als backoffice. De uitwisseling van deze sectorberichten kan door de applicatie worden gebruikt om de eigen gegevens te actualiseren.
De basisinfrastructuur van front- en backoffice bestaat hier uit zowel hardware- als softwaresystemen en levert onder andere netwerk-, transport-, en processingservices.



Figuur 5 Gegevensverkeer tussen applicaties ondersteund door een enterprise service bus

In Figuur 5 is, aan de hand van een veel gebruikt 3-lagen model, schematisch aangegeven hoe de uitwisseling van gegevens verloopt tussen applicaties en hoe de diensten van deze applicaties de gemeentelijke organisatie ondersteunen (weergegeven via de presentatie/organisatielaag). Tevens laat de figuur zien hoe de businesscomponenten en de informatie-infrastructuur hierin zijn gepositioneerd. In deze figuur is uitgegaan van een servicegeoriënteerde opzet waarin de gemeentelijke informatie-infrastructuur en de logicacomponenten (FO-taak en BO-taak) diensten leveren op een dienstenbus.[2]

De essentie van dit concept is de ontkoppeling van de aanroep en de implementatie van een dienst. Met dit concept wordt er een “loosly coupled” relatie tussen serviceaanbieder en servicegebruiker gerealiseerd. Ter illustratie, de businesscomponenten zijn de servicegebruikers van de onderliggende informatie-infrastructuur services. Maar ook de businesscomponenten bieden diensten die door de menselijke gebruikers kunnen worden gebruikt.

Uitgaande van een concept waarin de dienstenbus transparant diensten aanbiedt, is er ook ruimte voor de uitwisseling van gegevens tussen applicaties. In de huidige specificatie van de StUF-sectorberichten kunnen generieke diensten worden onderscheiden. De dienstenbus biedt de omgeving om deze berichten uit te wisselen tussen applicaties, de inhoudelijke interpretatie van de berichten blijft binnen de applicatie thuishoren.

In de figuur worden een tweetal backoffice-taken en één frontoffice-taak getoond. In het voorbeeld zijn de taken geïmplementeerd door twee applicaties in zogenaamde gemeentelijke modules (logicacomponent). Als voorbeeld wordt getoond hoe de frontoffice-applicatie communiceert met een gebruiker om een geboorte te registreren. Nadat de gegevens door de gebruiker zijn ingegeven en gecheckt op correctheid (binnen bepaalde grenzen), wordt de informatie in het frontoffice-informatiesysteem geplaatst. Van hieruit zal de informatie worden gecommuniceerd met een backoffice-applicatie. De dienstenbus levert hierbij services waarmee de frontoffice- de backoffice-applicatie een kennisgeving kan sturen. De informatiehandler in de frontoffice-applicatie genereert hiertoe een sectorbericht en stuurt dit naar de betreffende backoffice-applicatie. De informatiehandler in de ontvangende applicatie kan de gemeenschappelijke semantiek van de inhoud van het bericht begrijpen en vertalen naar de representatie van het interne informatiesysteem. Vervolgens kan de applicatie deze informatie in

[2] Op dit moment kan over de dienstenbus gesproken worden als een concept waarin software-gerelateerde diensten worden aangeboden die door een softwarefunctie worden geïmplementeerd. We zien dit concept terug in zogenaamde Enterprise Service Bus (ESP) concepten.

het eigen informatiesysteem opslaan. Op een vergelijkbare manier kunnen ook andere backoffice-informatiesystemen worden geïnformeerd.
Bij de verdere afhandeling van de geboorteregistratie kan een gebruiker in de backoffice gebruik maken van een service die door de dienstenbus wordt aangeboden. Een gemeentelijke module zorgt voor de realisatie van de dienst, en maakt daarbij gebruik van diensten die het informatiesysteem aanbiedt.

De services zijn in de figuur aangegeven met de grijze bolletjes die het virtuele aanroeppunt van een service symboliseren. Hiermee wordt duidelijk gemaakt dat de service en de realisatie van die service in een taak twee verschillende begrippen zijn. De implementatie van een service blijft voor de servicegebruiker verborgen, terwijl de implementatie kan veranderen zonder dat de servicegebruiker die in de gaten heeft.

In de figuur zijn twee instanties van een dienstenbus getekend. Echter, dit betekent niet dat het hier per definitie gaat om twee instanties van een dienstenbus. Afhankelijk van de eisen en de omgeving kan hier prima gekozen worden voor een enkele instantie van een dienstenbus die verschillende type diensten kan aanbieden. Het concept van de dienstenbus blijft gelijk.

4 DE STURING VAN DE BERICHTVERWERKING

Dit hoofdstuk bespreekt hoe de verwerking van de berichten wordt aangestuurd vanuit de stuurgegevens. Eerst worden de stuurgegevens besproken die voor alle berichten relevant zijn en daarna de specifieke stuurgegevens voor de verschillende berichtsoorten. In het kort wordt aangegeven in welke functionele behoefte ieder stuurgegeven voorziet. In bijlage 2 wordt de structuur van de stuurgegevens gespecificeerd in een XML-schema. Dit schema is bedoeld als referentie bij de implementatie van StUF.

Bij de bespreking van de stuurgegevens wordt niet ingegaan op het al dan niet verplicht zijn van de stuurgegevens. Dit is gedefinieerd in het XML-schema voor de header.

4.1 Algemene stuurgegevens

4.1.1 Berichtsoort

De in hoofdstuk 2 onderkende berichtsoorten worden in StUF niet met één stuurgegeven gecodeerd maar met twee, te weten de *berichtcode* en het *entiteittype*.
Het stuurgegeven *berichtcode* codeert de generieke berichtsoorten:

- Lk01: een kennisgevingbericht
  In paragraaf 4.2 staat hoe de verschillende varianten worden onderscheiden.
- Lv01: een synchroon vraagbericht
- La01: een synchroon antwoordbericht
- Lv02: een asynchroon vraagbericht
- La02: een asynchroon antwoordbericht
- Bv01: een ontvangstbevestigingsbericht
- Fo01: een foutbericht

Een bericht in StUF heeft altijd betrekking op objecten van één entiteittype en de generieke berichtsoort wordt verfijnd tot een berichtsoort voor een bepaald entiteittype door middel van het stuurgegeven *entiteittype.* In het sectormodel staat welke generieke berichtsoorten kunnen worden gebruikt bij een bepaald entiteittype. Dit wil niet zeggen dat een bericht slechts gegevens van één entiteittype bevat. Een kennisgevingbericht verstuurd naar aanleiding van de verhuizing van een persoon bevat bijvoorbeeld gegevens van het entiteittype *persoon o*m de persoon te identificeren en van het entiteittype *adres* met het oude en het nieuwe adres.

4.1.2 Versie StUF en sectormodel

Met StUF kunnen berichten worden uitgewisseld voor verschillende sectoren die elk een eigen sectormodel hanteren. Een ontvanger kan dus berichten ontvangen gemaakt op basis van verschillende sectormodellen. Een sociale dienst systeem zal in het kader van de binnengemeentelijke gegevensuitwisseling bijvoorbeeld werken met het sectormodel ‘binnengemeentelijk’ en voor de uitwisseling binnen de sociale sector met het sectormodel ‘sociale zekerheid’. Voor een systeem dat te maken heeft met berichten uit verschillende sectoren is het essentieel om te weten uit welke sector een bericht afkomstig is. In het stuurgegeven *sectormodel* moet daarom worden aangegeven welk sectormodel is gebruikt bij het aanmaken van het bericht.

Van een sectormodel en StUF kunnen verschillende versies naast elkaar gebruikt worden. De stuurgegevens *versie sectormodel* en *versie StUF* geven aan welke versies zijn gebruikt bij de aanmaak van een bericht. Bij de verwerking van een bericht kan hier desgewenst op gereageerd worden.

4.1.3 Adressering zender en ontvanger

Net zoals in een brief worden in een bericht de geadresseerde (ontvanger) en de afzender (de zender) opgenomen. Hiertoe zijn de stuurgegevens *Zender* en *Ontvanger* gedefinieerd. Deze twee stuurgegevens maken gebruik van een gemeenschappelijke adrestype-definitie. Hieronder worden de verschillende onderdelen van het adrestype nader gespecificeerd.

Met StUF worden berichten niet uitgewisseld tussen personen, maar tussen geautomatiseerde systemen. Zo’n geautomatiseerd systeem wordt beheerd door een organisatie. Het hoogste niveau in de adressering van een bericht is daarom de organisatie.

Het komt regelmatig voor dat een applicatie verschillende gegevensverzamelingen beheert. Kleine sociale diensten laten bijvoorbeeld hun uitkeringenadministratie uitvoeren door een grotere gemeente in de regio. Deze grotere gemeente gebruikt dan haar eigen sociale dienst applicatie voor het beheren van twee verschillende gegevensverzamelingen: haar eigen gegevensverzameling en de gegevensverzameling van de kleine gemeente die het werk heeft uitbesteed. Ook is het mogelijk dat een gemeente voor een applicatie zowel een productie- als een

testomgeving inricht. Omdat een systeem een applicatie is die een eigen gegevensverzameling beheert, is er in StUF voor gekozen om een systeem te identificeren met behulp van twee adresgegevens: de *applicatie* en de *administratie*. Met het adresgegeven *administratie* kan onderscheid worden gemaakt tussen de verschillende gegevensverzamelingen die een applicatie beheert.

Net zoals E-mail biedt StUF de mogelijkheid om een bericht naar een individuele gebruiker te zenden. Hiertoe is het adresgegeven *gebruiker* in StUF opgenomen. Dit adresgegeven kan ook gebruikt worden ten behoeve van autorisatie in het geval, dat bepaalde vraagberichten alleen beantwoord mogen worden, als de vraagsteller geautoriseerd is voor de gevraagde gegevens. Het antwoordende systeem kan aan de hand van de gebruiker nagaan of dit het geval is. StUF bevat geen voorschriften met betrekking tot autorisatie, maar het biedt dankzij dit stuurgegeven wel de mogelijkheid om autorisatiemechanismen in te bouwen in de berichtverwerkende software. In het sectormodel kunnen afspraken worden vastgelegd over de codering van gebruikers en de autorisatiemechanismen.

Samenvattend, bij de definitie van *zender* en o*ntvanger* wordt gebruikt gemaakt van een generiek adrestype. Dit adrestype bestaat uit de volgende vier gegevens:
1. Organisatie
2. Applicatie
3. Administratie
4. Gebruiker

4.1.4 Identificatie van berichten
Berichten worden geïdentificeerd met een *referentienummer*. StUF schrijft niet voor hoe het referentienummer opgebouwd moet worden. Berichten die onafhankelijk van elkaar zijn aangemaakt door verschillende systemen kunnen daardoor toevallig hetzelfde referentienummer hebben. StUF eist wel dat de combinatie van *referentienummer* en *zender* (verzendende organisatie, applicatie, administratie en gebruiker) uniek is. Elk bericht van een verzendend systeem moet dus een ander referentienummer krijgen.

Ontvangstbevestigingsberichten zijn geen functioneel onafhankelijke berichten. Een ontvangstbevestigingsbericht verzekert dat een bericht in goede orde ontvangen is. Dat wil zeggen dat de ontvanger de header van het bericht kan begrijpen en het bericht in zijn geheel heeft opgeslagen voor verdere verwerking. Ontvangstbevestigingsberichten krijgen als referentienummer een door de zender van het ontvangstbevestigingsbericht gedefinieerd nummer.

4.1.5 De volgorde waarin de berichten worden verwerkt
Een organisatie kan vanuit allerlei bronnen berichten toegezonden krijgen. Deze berichten dienen in de juiste volgorde verwerkt te worden. Het lijkt zinnig om de verwerkingsvolgorde primair te laten sturen door het tijdstip waarop het bericht is aangemaakt. Daartoe is het stuurgegeven *tijdstip bericht* gedefinieerd waarin tot op een honderdste seconde nauwkeurig het tijdstip van de aanmaak van het bericht kan worden gespecificeerd. Het staat het verzendende systeem vrij om zelf te bepalen hoe nauwkeurig het tijdstip wordt opgegeven. Een systeem dat bijvoorbeeld dagelijks één bericht verzendt, zou ervoor kunnen kiezen om het tijdstip te coderen als de datum (EEJJMMDD) aangevuld met acht nullen voor het tijdstip. StUF stelt wel als randvoorwaarde dat het *tijdstip bericht* groter is dan het *tijdstip bericht* van alle eerder door het systeem verzonden berichten.

Berichten afkomstig uit verschillende systemen kunnen uiteraard toevallig hetzelfde tijdstip bericht hebben. Als de berichten gesorteerd op het *tijdstip bericht* worden verwerkt, is het mogelijk dat berichten uit verschillende systemen door elkaar verwerkt worden met alle ongewenste gevolgen van dien. Het is daarom beter de berichten te sorteren op de combinatie van *tijdstip bericht* en *zender* (d.w.z. verzendende organisatie, applicatie, en administratie).

4.1.6 Berichtcycli
Wanneer een antwoord- of een foutbericht wordt ontvangen is het noodzakelijk om te weten op welk bericht wordt gereageerd. Hiervoor wordt in deze berichten het stuurgegeven *cross-referentienummer* opgenomen. Het *cross-referentienummer* wordt gevuld met de waarde van het referentienummer van het bericht waarop wordt gereageerd. De zender van een antwoord- of foutbericht vult het referentienummer met een eigen unieke waarde. Op deze manier worden de antwoord- en foutberichten geïdentificeerd. De combinatie van zender en referentienummer moet uniek zijn. Ook in een ontvangstbevestigingsbericht wordt het crossreferentienummer gevuld met het referentienummer van het bericht naar aanleiding waarvan het ontvangstbevestigingsbericht wordt verzonden.

De tot nu toe besproken stuurgegevens komen met uitzondering van het *cross-referentienummer* voor in elk bericht. Daarnaast is er nog een aantal stuurgegevens specifiek voor een bepaalde berichtsoort.

4.2 Stuurgegevens voor kennisgevingberichten

In paragraaf 2.3 zijn vier varianten binnen een kennisgevingbericht onderkend:

- *Toevoeging*
  Bij een toevoeging is in het zendende systeem een occurrence toegevoegd, omdat is vastgesteld dat in de werkelijkheid een voor het zendende systeem relevant object bestaat.
- *Wijziging*
  Bij een wijziging is in het zendende systeem een occurrence gewijzigd, omdat is vastgesteld dat er in de werkelijkheid eigenschappen (gegevens) zijn veranderd van het object waar naar die occurrence verwijst.
- *Verwijdering*
  Bij een verwijdering is in het zendende systeem een occurrence verwijderd, omdat is vastgesteld dat in de werkelijkheid het object waarnaar de occurrence verwijst, niet meer bestaat of niet meer relevant is voor het zendende systeem.
- *Correctie*
  Bij een correctie is in het zendende systeem een occurrence gewijzigd, omdat is vastgesteld dat de vastgelegde waarden niet correct waren. Bij een correctie is het object in de werkelijkheid waarnaar de occurrence verwijst, zelf niet gewijzigd.

Deze verschillende varianten worden als volgt gecodeerd in het stuurgegeven *mutatiesoort*:

- 'T': *Toevoeging*
- 'W': *Wijziging*
- 'V': *Verwijdering*
- 'C': *Correctie*

Het is aan het ontvangende systeem om te interpreteren in hoeverre een kennisgeving relevant is en of de kennisgeving het gevolg is van het ontstaan of verdwijnen van een object c.q. van het relevant worden of niet meer relevant zijn van het object voor het zendende systeem. Het kennisgevingbericht bevat geen aanduiding van de gebeurtenis die aanleiding gaf tot de wijziging van de gegevens (zie paragraaf 2.3).

Naast de mutatiesoort is in een kennisgevingbericht ook relevant hoe het ontvangende systeem geacht wordt te reageren op het bericht. Een kennisgevingbericht kan puur informatief bedoeld zijn: het ontvangende systeem mag zelf beslissen of de kennisgeving al dan niet wordt verwerkt in de eigen gegevens. Daarnaast kan het verplicht zijn voor het ontvangende systeem om de gegevens over te nemen. Of een kennisgevingbericht informatief is of verplicht over te nemen geeft het stuurgegeven *indicator overname* aan met 'I' (informatief) respectievelijk 'V' (verplicht). Aanvullende afspraken over de omgang met deze rubriek kunnen worden vastgelegd in het sectormodel.

4.3 Stuurgegevens voor vraag- en antwoordberichten

Wanneer een systeem gegevens vraagt, dan zal het antwoordende systeem willen weten waar precies om wordt gevraagd en hoe de gevraagde gegevens moeten worden teruggegeven. Het gaat om:

- *Sortering*: in welke volgorde moeten de occurrences worden teruggegeven
  Synchrone vraagberichten worden vaak gebruikt om gegevens op te halen die in een lijst aan de gebruikers worden getoond. De gebruiker verwacht als antwoord een bericht met daarin de gegevens in een bepaalde, eventueel zelfgekozen, sortering.
- *Selectie*: van welke occurrences of objecten worden gegevens gevraagd
  Een systeem kan van grote aantallen objecten gegevens bevatten en het vragende systeem is dikwijls geïnteresseerd in een specifiek object of in een klein aantal objecten met specifieke kenmerken.
- *Scope*: welke gegevens moeten worden teruggegeven
  In een vraagbericht wordt altijd gevraagd naar occurrences van het entiteittype gespecificeerd in het stuurgegeven entiteittype. Een dergelijk entiteittype kan een groot aantal attributen en relaties hebben. Het is dus wenselijk om in het vraagbericht te kunnen specificeren om welke gegevens het precies gaat en of historische gegevens al dan niet gewenst zijn.
- *Maximum aantal*: hoeveel occurrences mogen maximaal in één keer worden teruggegeven
  Wanneer bij een synchroon vraagbericht een groot aantal occurrences voldoet aan de selectiecriteria, dan kan het in een keer teruggeven van alle occurrences leiden tot onacceptabele responstijden.

Alleen de *sortering*, de *indicator historisch* en het *maximum aantal* zijn stuurgegevens in de header. De *selectie* en de *scope* worden gedefinieerd in de body en verder besproken in paragraaf 6.3.1 en 6.3.4.

4.3.1 Sortering
Het stuurgegeven *sortering* is bedoeld om aan te geven volgens welke sortering de occurrences teruggegeven moeten worden. Het sectormodel dient de sorteringen per entiteittype te specificeren. Bij het entiteittype *persoon* zou bijvoorbeeld gespecificeerd kunnen worden dat eerst aflopend wordt gesorteerd op geboortedatum en daarbinnen oplopend op geslachtsnaam en voorletters. Een andere sortering zou kunnen zijn oplopend op achtereenvolgens postcode, huisnummer, geslachtsnaam en voorletters. De mogelijke sorteringen bij een entiteittype worden gecodeerd met een getal tussen de 1 en de 99.

4.3.2 Historie
Met StUF kan de historie van gegevens worden opgevraagd. Standaard vraagt een vraagbericht echter alleen om actuele gegevens. Teneinde ook historische gegevens te ontvangen, kan het vragende systeem het stuurgegeven *indicator historisch* met ‘`true`’ vullen. Als dit stuurgegeven niet met ‘`true`’ is gevuld, dan worden in het antwoordbericht alleen actuele gegevens teruggegeven.

4.3.3 Maximum aantal
Met het stuurgegeven *maximum aantal* kan worden aangegeven hoeveel occurrences er maximaal in het antwoordbericht mogen worden teruggestuurd. Zeker bij synchrone vraagberichten is het in het kader van de performance van belang om het maximum aantal niet te groot te kiezen, bijvoorbeeld altijd kleiner dan 100 of gelijk aan het aantal occurrences dat in één keer in een lijst getoond kan worden. Bij asynchrone vraag- en antwoordberichten voorkomt een maximum aantal dat verkeerd geformuleerde vraagberichten leiden tot het terugsturen van veel antwoordberichten met alle gevolgen voor schijfcapaciteit en performance van dien. Bij asynchrone vraag- en antwoordberichten kan de gebruiker niet tussentijds ingrijpen.

4.3.4 Het berichtscenario voor vraag- en antwoordberichten
Wanneer in het vraagbericht een maximum aantal terug te geven occurrences is gespecificeerd, dan hoeven niet alle occurrences te zijn teruggezonden die aan de selectiecriteria voldoen. Dit wordt in het antwoordbericht aangegeven met het stuurgegeven *indicator vervolgvraag.* De waarde ‘`true`’ geeft aan dat er nog meer occurrences zijn die aan de selectiecriteria voldoen. De waarde ‘`false`’ geeft aan dat alle occurrences zijn verstuurd.

Indien het vragende systeem meer occurrences wil ontvangen dan kan een vervolgvraag worden gesteld door een nieuw vraagbericht te versturen. Met het stuurgegeven *indicator vervolgvraag* wordt in een vraagbericht aangegeven of het om een vervolgvraag gaat. De waarde ‘`false`’ geeft aan dat het een nieuwe vraag betreft en de waarde ‘`true`’ een vervolgvraag. In paragraaf 6.3.5 staat hoe in een vervolgvraag wordt doorgegeven waar verder gegaan moet worden met het teruggeven van occurrences. Het antwoordende systeem kan elk vraagbericht op zich beantwoorden en hoeft niet bij te houden op welke vragen nog geen volledig antwoord is gegeven. Het vragende systeem kan na elk antwoordbericht beslissen of het al dan niet een vervolgvraag wil stellen.

4.3.5 Gegevensbeheer
Vanuit het oogpunt van gegevensbeheer kan het nuttig zijn om in een vraagbericht aan te geven of het vragende systeem in de toekomst kennisgevingberichten wil ontvangen voor de gevraagde occurrences. Dit kan met behulp van het stuurgegeven *indicator afnemerindicatie*. De waarde ‘`true`’ geeft aan dat het vragende systeem op de hoogte gehouden wil worden van wijzigingen. De indicator *afnemerindicatie* mag alleen op ‘`true`’ gezet worden bij asynchrone vraagberichten, omdat bij synchrone vraagberichten niet te voorspellen is welke occurrences een gebruiker zal vragen en wat hij vervolgens met de gevraagde occurrences doet. Immers bij synchrone vraagberichten zal er vaak sprake zijn van een menselijke gebruiker die het vragende systeem aanstuurt. Bij een asynchroon vraagbericht zal er eerder sprake zijn van een geautomatiseerd systeem dat informatie bij een ander systeem opvraagt.

In een antwoordbericht geeft de waarde ‘`true`’ voor het stuurgegeven *indicator afnemerindicatie* aan, dat in het antwoordende systeem bij de teruggegeven occurrences afnemerindicaties zijn geplaatst voor het vragende systeem. Dit kan in het bijzonder van belang zijn bij asynchrone antwoordberichten die zijn verstuurd zonder vraagbericht.

4.3.6 Foutmelding
Soms kan een bericht wel verwerkt worden, maar is niet aan alle verwachtingen van de vragende partij voldaan. Om dit te kunnen aangeven is het stuurgegeven *foutmelding* gedefinieerd. Voor een paar gevallen definieert de standaard het gebruik van dit veld. In een sectormodel kunnen altijd extra foutmeldingen gedefinieerd worden.

4.4 Schema voor de header met stuurgegevens

Dit hoofdstuk geeft een beschrijving van alle stuurgegeven die in een StUF-bericht kunnen voorkomen. Een deel van de stuurgegevens zijn generiek en een ander deel is specifiek voor een type bericht. Hieronder staat een tabel die een samenvattend overzicht geeft welke stuurgegevens in welke type StUF-bericht voorkomen.

Stuurgegeven	Kennis-gevings-bericht	Vraag-bericht	Ant-woord-bericht	Beves-tigings-bericht	Fout-bericht
berichtsoort	X	X	X	X	X
entiteittype	X	X	X	X	X
sectormodel	X	X	X	X	X
versieStUF	X	X	X	X	X
versieSectormodel	X	X	X	X	X
zender	X	X	X	X	X
ontvanger	X	X	X	X	X
referentienummer	X	X	X	X	X
tijdstipBericht	X	X	X	X	X
mutatiesoort	X				
indicatorOvername	X				
tijdstipMutatie	O				
sortering		O			
maximumAantal		O			
indicatorHistorisch		O			
indicatorVervolgvraag		X	X		
indicatorAfnemerIndicatie		O	O		
crossRefNummer			X	X	X
foutmelding			O		

Tabel 1 Overzicht headergegevens per type StUF-bericht (X=verplicht, O=Optioneel)

5 DE SYNTAX EN SEMANTIEK VAN EEN OBJECT IN EEN BERICHT

StUF is bedoeld om gegevens eenvoudig te kunnen uitwisselen tussen systemen. In het vorige hoofdstuk is beschreven hoe de berichtverwerking wordt aangestuurd. Dit hoofdstuk gaat in op de kern van de zaak en beschrijft hoe de gegevens van een object in de body van een bericht worden opgenomen.

De eerste paragraaf van dit hoofdstuk gaat in op de gewenste functionaliteit en de bijbehorende semantische aspecten. De tweede paragraaf gaat dieper in op de syntax voor een object . Het volgende hoofdstuk beschrijft wanneer en hoe objecten en hun gegevens in de berichten worden opgenomen.

5.1 Gewenste functionaliteit en semantiek

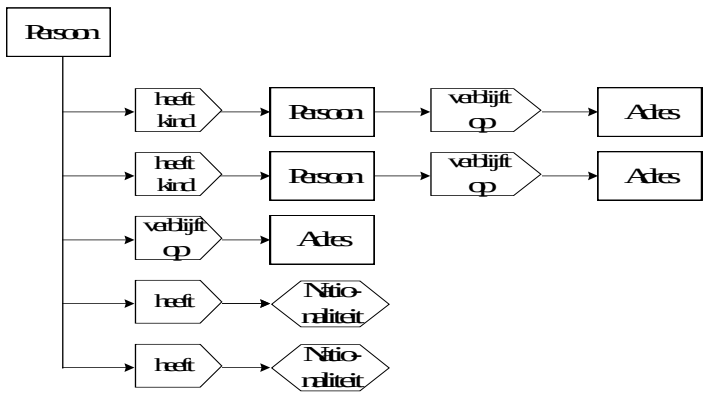
5.1.1 Soorten entiteittypen en hun plaats in een bericht

StUF onderscheidt verschillende soorten entiteittypen:

1. *fundamentele entiteittypen*
   Fundamentele entiteittypen representeren in de werkelijkheid bestaande objecten. Voorbeelden zijn adres, persoon, niet-natuurlijk persoon, kadastraal object en verblijfsobject. Het uitwisselen van gegevens over dit soort objecten is de bestaansgrond voor StUF. Voor fundamentele entiteittypen zullen normaliter zowel kennisgevingberichten als vraag- en antwoordberichten worden gedefinieerd in het sectormodel.
2. *tabelentiteittypen*
   Tabelentiteittypen staan voor tabellen en niet voor reëel in de werkelijkheid bestaande objecten. Tabellen worden vaak gebruikt om een set toegestane waarden te definiëren voor een bepaalde eigenschap, Tabellen zijn in het bijzonder zinvol, als de set toegestane waarden in de loop van de tijd kan veranderen. Binnen de GBA worden bijvoorbeeld tabellen gebruikt met de toegestane waarden voor land, gemeente, en nationaliteit. De verzameling gemeenten verandert bijvoorbeeld bij een gemeentelijke herindeling. Omdat deze tabellen slechts sporadisch wijzigen, zullen systemen ze normaal gesproken zelf onderhouden en er geen vragen over stellen aan andere systemen. Voor tabelentiteittypen zullen in een sectormodel normaal gesproken alleen kennisgevingberichten worden gedefinieerd.
3. *relatie-entiteittypen*
   Een relatie-entiteittype representeert een relatie, bijvoorbeeld tussen een persoon en een adres. Tussen persoon en adres bestaan verschillende relaties, bijvoorbeeld PERSOON.verblijft op.ADRES[3], PERSOON.ontvangt post op.ADRES, en ADRES.heeft erop gevestigd.PERSOON. Een relatie-entiteittype definieert een verband tussen het linker object en het rechter object uitgaande van het linker object (de richting van de relatie is hier dus relevant). De relatie PERSOON.verblijft op.ADRES geeft de verblijfplaats van een persoon (een eigenschap van een persoon) en de relatie ADRES.heeft erop gevestigd.PERSOON geeft de personen die op een bepaald adres zijn gevestigd (eigenschappen van dat adres).
   Voor elke onderkende relatie moet in het sectormodel een relatie-entiteittype worden gedefinieerd. In het bovenstaande voorbeeld zijn er twee relaties tussen PERSOON en ADRES, en één tussen ADRES en PERSOON. In totaal zijn dit dus drie relatie-entiteittypen.
   Een relatie als een huwelijk tussen twee personen (PERSOON.is (was) gehuwd met.PERSOON) heeft zelf weer eigenschappen als de datum huwelijkssluiting en het land huwelijkssluiting. Die zijn onderdeel van het relatie-entiteittype.
   Voor een relatie-entiteittype kunnen geen berichten worden gedefinieerd, omdat een relatie-entiteit afhankelijk is van het object van waaruit de relatie ligt. Wanneer een huwelijk gedefinieerd is als een relatie-entiteittype, dan kunnen er dus voor een huwelijk geen berichten worden gedefinieerd. Indien dit bezwaarlijk is, kan het huwelijk ook gedefinieerd worden als een fundamenteel entiteittype met twee relaties naar het fundamentele entiteittype persoon. De beperking dat er voor een relatie-entiteittype geen berichten gedefinieerd kunnen worden legt dus geen beperking op aan de functionaliteit van de gegevensuitwisseling.

[3] Relatie-entiteittypen worden genoteerd als een omschrijving van de relatie tussen twee punten met in hoofdletters links ervan het entiteittype van waaruit de relatie ligt en rechts ervan het entiteittype waarnaar de relatie ligt.

5.1.2 De opbouw van een object in een bericht



Figuur 6: Opbouw van een bericht uit de verschillende entiteittypen

Met behulp van de fundamentele, relatie- en tabelentiteittypen kunnen complexe gegevensstructuren worden geconstrueerd. De nevenstaande figuur illustreert dit aan de hand van een bericht over een persoon. De rechthoek linksboven in de figuur stelt de persoon voor. Rechthoeken worden gebruikt om een fundamenteel entiteittype aan te duiden. Een dergelijke rechthoek is de container met de gegevens van de persoon. Het bericht bevat ook gegevens over zijn twee kinderen, zijn adres en zijn twee nationaliteiten. Deze gegevens worden aan de persoon gekoppeld met relatie-entiteittypen, aangeduid met een blokpijl met daarin een omschrijving van de relatie. De nationaliteit als tabelentiteittype wordt aangeduid met een zeshoek. Van de kinderen omvat het bericht ook het adres: het relatie-entiteittype ‘verblijft op’ koppelt een adres aan het kind.

Het object persoon wordt dus in een bericht opgenomen met behulp van
- Een fundamentele entiteit met de attributen van de persoon
- Nul of meer relatie-entiteiten met de attributen van de relaties van de persoon
- Evenveel fundamentele en tabelentiteiten voor de gerelateerden van de relaties als er relaties zijn

Voor het opnemen van entiteiten in een bericht gelden de volgende regels:
- Op het hoogste niveau mogen alleen fundamentele en tabelentiteiten voorkomen
- Een fundamentele entiteit en een relatie-entiteit mag nul of meer relatie-entiteiten hebben
- Een tabelentiteit mag geen relatie-entiteit hebben
- Een relatie-entiteit heeft als gerelateerde altijd een fundamentele entiteit of een tabelentiteit

Een voorbeeld van een relatie-entiteit binnen een relatie-entiteit is een huwelijk als relatie-entiteit, van waaruit via een relatie-entiteit wordt verwezen naar de echtscheidingsadvocaat.

5.1.3 Identificatie

Een steeds terugkerend probleem bij de uitwisseling van gegevens is om vast te stellen op welk object in de werkelijkheid de gegevens betrekking hebben. Geslachtsnaam, voorletters, en adres zijn bijvoorbeeld niet altijd voldoende om een persoon te identificeren, omdat twee gezinsleden met dezelfde voorletters op hetzelfde adres wonen. Problemen met onvolledige gegevens (niet alle voorletters zijn meegestuurd) of onjuiste gegevens (een tikfout in de geslachtsnaam, de voorletter van de roepnaam in plaats van de officiële voornaam, of een foutief adres) laten we dan nog buiten beschouwing.

Om problemen met de identificatie te vermijden kan in het sectormodel het beste per entiteittype worden beschreven met welke combinaties van gegevens occurrences kunnen worden geïdentificeerd. Deze identificerende gegevens samen worden ook wel aangeduid als de kerngegevens van een entiteittype. Bij personen kan bijvoorbeeld voor de volgende identificerende combinaties gekozen worden:
- SoFi-nummer
- A-nummer
- Geslachtsnaam, voorvoegsels, voorletters, en verblijfsadres
- Geslachtsnaam, voorvoegsels, voorletters, en geboortedatum

Kennisgevingberichten en asynchrone antwoordberichten worden normaliter zonder menselijk ingrijpen verwerkt. Om te waarborgen dat systemen niet vervuild raken, moeten alle kerngegevens volledig in een kennisgevingbericht en een asynchroon antwoordbericht voorkomen. In het sectormodel dienen de kerngegevens te worden gedefinieerd als een verzameling elementen en relaties. Er wordt geen apart element gedefinieerd voor de kerngegevens, omdat ook relaties onderdeel kunnen zijn van de kerngegevens (bijvoorbeeld het adres van een persoon).

5.1.4 Systeemsleutels

Systemen identificeren de occurrences van een entiteittype met een al dan niet betekenisloze unieke sleutel. Ook een dergelijke systeemsleutel kan in het berichtenverkeer gebruikt worden om de identificatie te vereenvoudigen. In de praktijk blijken drie systeemsleutels relevant te zijn:

- de sleutel in het verzendende systeem
- de sleutel in het ontvangende systeem
- de sleutel in een systeem dat zorgt voor gegevensbeheer of te wel het gegevensbeheersysteem

Zeker betekenisloze sleutels zullen niet voorkomen in het sectormodel, waar per slot van rekening de werkelijkheid wordt gemodelleerd en niet de opslag van gegevens in een database. Omdat deze sleutels van belang zijn bij de identificatie van occurrences, worden deze sleutels in StUF als afzonderlijk attribuut onderkend. In het sectormodel hoeven deze sleutels niet opgenomen te worden.

5.1.5 Gegevensgroepen

Bij het doorgeven van wijzigingen is het vaak nuttig niet alleen het gewijzigde gegeven door te geven, maar ook nauw daaraan gerelateerde gegevens. Bij wijziging van een huisnummertoevoeging is het handig om expliciet aan te geven of het huisnummer al dan niet ook gewijzigd wordt. Als bij een vrouw de geslachtsnaam gewijzigd wordt, is het handig om expliciet aan te geven of de voorvoegsels en de voorletters al dan niet ook gewijzigd worden. Als niet alle systemen voor alle gegevens dezelfde waarde registreren, is dit absoluut noodzakelijk om te voorkomen dat een afwijkende waarde onbedoeld wordt gewijzigd. Het ene systeem heeft als adres Appelstraat 3 vastliggen, wijzigt dit in Appelstraat 3 B, en geeft dit door. Het andere systeem kan om wat voor reden dan ook als adres Twijnstraat 17 hebben geregistreerd. De wijziging van het adres Appelstraat 3 in 3 B, mag dan niet leiden tot de wijziging van het adres Twijnstraat 17 in 17 B.

Om dit soort problemen te voorkomen is het verstandig om in het sectormodel bij een entiteittype zogenaamde gegevensgroepen te definiëren. Zodra één gegeven uit een gegevensgroep wordt opgenomen in een kennisgeving- of een antwoordbericht zonder vraagbericht, worden ook de andere gegevens uit die groep in het bericht opgenomen. Bij persoon zouden bijvoorbeeld de geslachtsnaam, de voorvoegsels geslachtsnaam en de voorletters een gegevensgroep kunnen vormen. Bij adres zouden de gebruikelijke adresseringsgegevens een gegevensgroep kunnen vormen.

In het XML-schema zijn gegevensgroepen heel eenvoudig te definiëren als een al dan niet optioneel element dat de gegevens uit de gegevensgroep als verplichte elementen bevat. Het StUF schrijft overigens niet voor dat gegevensgroepen op deze manier gedefinieerd worden. Er kan ook volstaan worden met het in het sectormodel definiëren van de gegevensgroep als een set elementen zonder dat dit in het schema wordt afgedwongen.

5.1.6 Historische gegevens

Onder een historisch gegeven verstaan we een gegeven waarvan de waarde niet meer overeenkomt met de waarde in de werkelijkheid. Om historische gegevens uit te kunnen wisselen dient het tijdvak te worden gespecificeerd waarin een gegeven geldig is (geweest). In StUF is er voor gekozen om niet per individueel gegeven een tijdvak geldigheid in het bericht op te nemen, maar per occurrence. Er zijn drie redenen voor deze keuze:

1. De implementatie van StUF is simpeler omdat niet per gegeven maar per entiteit een tijdvak kan worden gespecificeerd,
2. Er wordt aangesloten bij de verwerking in databases, waar meestal ook met tijdvakken geldigheid per record wordt gewerkt,
3. Bij het gebruik van datums is het noodzakelijk om onderscheid te kunnen maken tussen een onbekende datum en een niet bestaande datum. Bij implementatie in XML kunnen datums met een dergelijk waardebereik niet eenvoudig gedefinieerd worden als attributen maar kunnen zij beter als element worden gespecificeerd.

Het is niet verplicht om een tijdvak geldigheid bij een entiteit op te nemen. Op het moment dat dit ontbreekt dient de ontvangende partij ervan uit te gaan dat de entiteit actuele gegevens bevat.

Een tijdvak geldigheid is van toepassing op alle in de entiteit voorkomende attributen van een entiteittype. Het geldt niet voor gekoppelde relatie-entiteiten en de daarachter liggende occurrences. In het voorbeeld in heeft het tijdvak geldigheid dus alleen betrekking op de gegevens behorend bij het blokje persoon en niet op de gegevens van de relatie-entiteiten ‘heeft kind’, ‘verblijft op’, ‘heeft nationaliteit’ en de daarachter liggende occurrences. Het op het eerste gezicht voor de hand liggende alternatief om het tijdvak geldigheid te laten gelden voor alle gegevens uit het voorbeeld werkt niet omdat de gegevens van de als kind gerelateerde personen een eigen tijdvak geldigheid hebben. Dit tijdvak geldigheid dient bij die gerelateerde occurrence gespecificeerd te kunnen worden.

5.1.7 De structuur van objecten in berichten

Het StUF legt een beperking op aan de structuur van objecten in kennisgevingberichten. Een kennisgevingbericht kan zonder expliciete afspraken in het sectormodel slechts drie typen wijzigingen doorgeven:

1. Het toevoegen, verwijderen, wijzigen en corrigeren van fundamenteel entiteit
2. Het toevoegen of beëindigen van een relatie
   Het toevoegen en verwijderen van alle relaties die direct gekoppeld zijn aan het fundamentele entiteittype in de kop van het bericht kan via een kennisgevingbericht worden doorgegeven. Wanneer dit niet het geval is, dient dit expliciet in het sectormodel te worden vermeld.
3. Een wijziging in de gegevens van relatie-entiteit
   Een wijziging in de gegevens van een relatie-entiteit die direct gerelateerd is aan een fundamentele entiteit uit de kop van het bericht kan via een kennisgevingbericht worden doorgegeven. Wanneer deze wijziging niet op deze manier kan worden doorgegeven dient dit expliciet in het sectormodel te worden vermeld[4].

In het voorbeeld in zal een kennisgevingbericht voor het entiteittype persoon het toevoegen en verwijderen van personen kunnen doorgeven. Een dergelijk kennisgevingbericht kan wijzigingen doorgeven van gegevens van het entiteittype *persoon* linksboven in de figuur en van de daaraan gerelateerde relatie-entiteittypen ‘heeft kind’ (2x), ‘verblijft op’ en ‘heeft’ (ook 2x). Voor de gerelateerde kinderen, het verblijfsadres en de nationaliteit kunnen geen wijzigingen in de gegevens worden doorgegeven, tenzij dit voor een fundamentele gerelateerde expliciet is toegestaan in het sectormodel. Omdat bij het toevoegen van een relatie het gerelateerde object niet hoeft voor te komen in het ontvangende systeem, mag een gerelateerde fundamentele entiteit wel worden toegevoegd. Een kennisgevingbericht kan bij een persoon bijvoorbeeld een relatie doorgeven naar een nog niet in het ontvangende systeem voorkomend adres. Gerelateerde tabelentiteiten mogen niet worden toegevoegd. Van een gerelateerde fundamentele entiteit mogen tenzij anders gespecificeerd in het sectormodel alleen de kerngegevens worden opgenomen. Door deze regels hebben kennisgevingberichten een relatief simpele structuur.

Voor antwoordberichten legt het StUF geen beperkingen op aan de structuur van het bericht. Een antwoordbericht mag een complexe structuur hebben. Hiervoor zijn twee redenen:

1. De vragende partij kan zelf bepalen welke gegevens hij exact wil ontvangen. Door de structuur van een antwoordbericht niet op dezelfde wijze te beperken als van kennisgevingberichten heeft men bij het ontwerpen van een sectormodel de vrijheid de complexiteit van de software die de vragen stelt af te wegen tegen de complexiteit van de antwoordende software (zie ook paragraaf 6.3).
2. Asynchrone antwoordberichten kunnen gebruikt worden om de inhoud van een hele database te verzenden. In dit geval is het plezierig om in één bericht alle gegevens relevant voor een fundamenteel entiteittype te kunnen opnemen.

5.1.8 Aantal objecten in een bericht

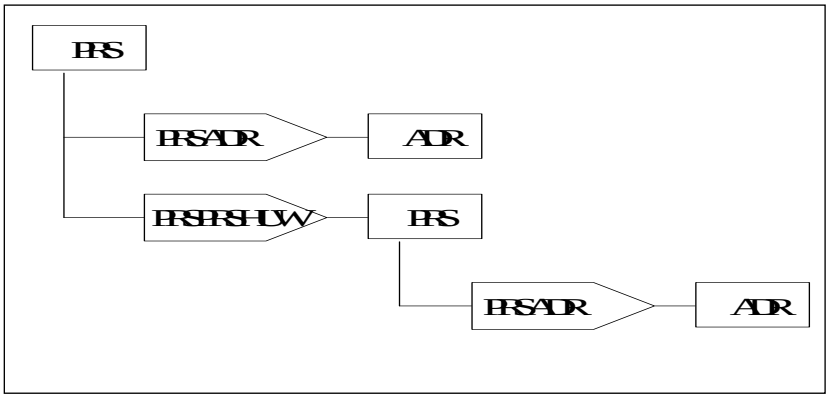
Bij vragen zijn vaak gegevens van verschillende objecten nodig. Een sociale dienst systeem wil bijvoorbeeld in één synchroon antwoordbericht alle personen terugkrijgen die op een bepaald adres verblijven. Antwoordberichten kunnen daarom gegevens van meerdere objecten bevatten. Een kennisgevingbericht wordt door een zender gebruikt om een wijziging in de gegevens van een object te melden. Er is daarom voor gekozen om in een kennisgevingbericht de gegevens van precies één object op te nemen. Bij een toevoeging- en verwijderkennisgeving bevat het bericht één occurrence van het object en bij een wijzig- of correctiekennisgeving twee occurrences: de eerste met de ‘oude’ waarden en de tweede met de ‘nieuwe’ waarden.

5.2 De syntax voor een object in een bericht

5.2.1 Voorbeeld van de opbouw van een bericht

 heeft aan de hand van het voorbeeld van een persoon laten zien dat van een persoon in een bericht niet alleen de attributen van het entiteittype persoon voorkomen, maar ook andere entiteittypen met hun attributen. In staat een ander voorbeeld. Deze figuur geeft de entiteittypen binnen een bericht over het huwelijk en de partner van een persoon. Het blokje linksboven staat voor het fundamentele entiteittype *persoon* (PRS). Dit blokje bevat de direct bij de persoon behorende gegevens. Van de persoon is ook het adres in het bericht opgenomen door middel van het relatie-entiteittype PRSADR en het fundamentele entiteittype ADR. Na het adres volgen de gegevens over het huwelijk in het relatie-entiteittype PRSPRSHUW en de gegevens over de partner in het fundamentele entiteittype PRS. Van de partner wordt ook weer het adres gegeven.

[4] In deze gevallen kan een relatie-entiteittype als fundamenteel entiteittype worden gespecificeerd en kunnen op deze manier de wijzigingen in een relatie-entiteittype worden uitgewisseld.

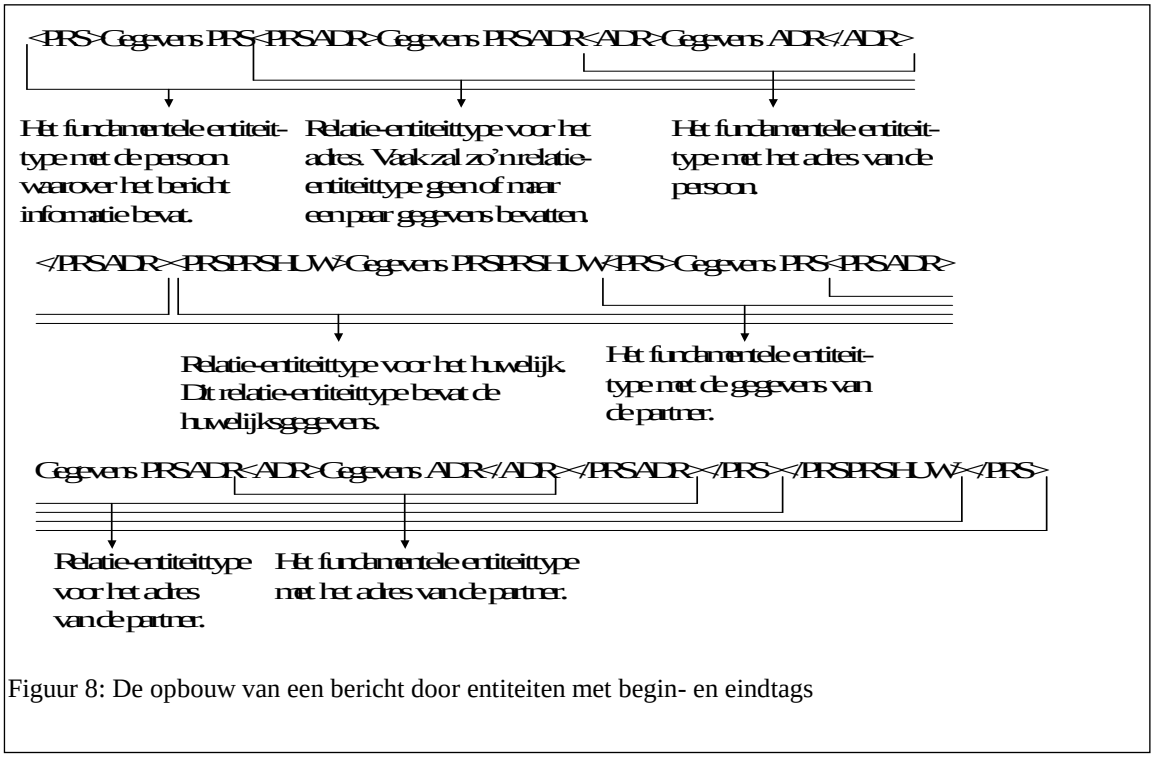


geeft schematisch de opbouw van de berichtbody in XML weer. Met elk blok in correspondeert een begin- en een eindtag. Voor PRS bijvoorbeeld is de begintag <PRS> en de eindtag </PRS> en voor PRSADR respectievelijk <PRSADR> en </PRSADR>. Tussen het begin- en eindtag staan alle gegevens die behoren tot het entiteittype. Zo begint het bericht met de tag <PRS> en eindigt het met </PRS>, want alle gegevens in het bericht behoren tot de persoon corresponderend met het blokje linksboven. De tags <PRSADR> en </PRSADR> omsluiten zowel de gegevens over de relatie tussen een persoon en zijn verblijfsadres als de gegevens over het adres zelf. De tags <ADR> en </ADR> omsluiten alleen de adresgegevens. De figuur legt verder zichzelf uit. Bij het verblijfsadres is de enige functie van het relatie-entiteittype te kunnen aangeven in welk tijdvak de persoon op het adres verbleef. Het relatie-entiteittype huwelijk bevat zelf ook nog gegevens over het huwelijk. Een relatie-entiteittype volgt altijd na een fundamenteel entiteittype of een ander relatie-entiteittype. Een relatie-entiteittype wordt altijd gevolgd door een fundamenteel entiteittype of een tabelentiteittype en kan gevolgd worden door een relatie-entiteittype.

5.2.2 Metagegevens over een fundamenteel, een relatie- en een tabelentiteittype

Over een fundamenteel, een relatie- of een tabelentiteittype kunnen een aantal metagegevens in het bericht worden opgenomen. Het gaat om de volgende metagegevens:

- *Soort entiteit*
  Dit metagegeven geeft aan of het gaat om een fundamenteel, een relatie- of een tabelentiteittype. Het attribuut soort entiteit heeft het volgende waardebereik:
  - 'F': Fundamentele entiteit
  - 'R': Relatie-entiteit
  - 'T': Tabelentiteit
- *Begindatum geldigheid*
  Dit metagegeven geeft aan vanaf welke datum de gegevens geldig zijn. Onder het geldig zijn van de gegevens wordt verstaan, dat de eigenschappen van het object waarop de gegevens betrekking hebben op de begindatum tijdvak geldigheid allemaal de waarde hebben (c.q. hadden, indien het object op die datum ophield te bestaan) zoals die in het navolgende gegevens blok is gegeven. De begindatum geldigheid heeft alleen betrekking op de eigen gegevens van de entiteit en niet op eventuele daarna nog volgende relatie-entiteiten. Indien dit metagegeven niet aanwezig is, dan wordt ervan uitgegaan dat de gegevens actueel zijn. De *begindatum geldigheid* wordt opgenomen als een element binnen het element `tijdvakGeldigheid`, dat weer een element is binnen een fundamenteel of een relatie-entiteittype. De datum wordt gecodeerd met acht cijfers in het formaat EEJJMMDD. Wanneer de tijdvakdatum niet op de dag nauwkeurig bekend is, dan wordt de datum gevuld met een geldige datum en wordt een attribuut `indOnvolledigeDatum` bij het tijdvakdatum-element gezet. Als wel de maand, maar niet de dag bekend is, dan krijgt dit attribuut de waarde 'D' en als wel het jaar maar niet de maand bekend is, dan krijgt dit attribuut de waarde 'M'. De default waarde voor dit attribuut is 'V', dat wil zeggen de datum is volledig.
- *Einddatum geldigheid*
  Dit metagegeven geeft aan tot welke datum de gegevens geldig zijn. Onder het geldig zijn van de gegevens wordt verstaan, dat de eigenschappen van het object waarop de gegevens betrekking hebben vanaf de begindatum tijdvak geldigheid tot de einddatum tijdvak geldigheid allemaal de gespecificeerde waarde hebben. De einddatum tijdvak geldigheid heeft alleen betrekking op de eigen gegevens van de entiteit en niet op eventuele daarna nog volgende relatie-entiteiten. Indien dit element niet aanwezig is, dan wordt ervan uitgegaan dat de gegevens ook in de toekomst geldig zijn. De *einddatum geldigheid* wordt opgenomen als een element binnen het element `tijdvakGeldigheid`, dat weer een element is binnen een fundamenteel of een relatie-entiteittype. De datum wordt gecodeerd met acht cijfers in het formaat EEJJMMDD. Onvolledige datums worden gespecificeerd zoals hierboven beschreven bij de begindatum tijdvak geldigheid.



Figuur 8: De opbouw van een bericht door entiteiten met begin- en eindtags

- *Begindatum relatie*
  Dit metagegeven komt alleen voor bij relatie-entiteiten en geeft aan op welke datum de relatie ontstaan is. De begin- en einddatum relatie worden als metagegeven gedefinieerd, omdat de standaard voor het ondersteunen van historische gegevens specificaties geeft voor het werken met deze twee metagegevens. De *begindatum relatie* wordt opgenomen als een element binnen het element `tijdvakRelatie`, dat weer een element is binnen een fundamenteel of een relatie-entiteittype De datum wordt gecodeerd met acht cijfers in het formaat EEJJMMDD. Onvolledige datums worden gespecificeerd zoals hierboven beschreven bij de begindatum tijdvak geldigheid.
- *Einddatum relatie*
  Dit metagegeven komt alleen voor bij relatie-entiteiten en geeft aan vanaf welke datum de relatie niet meer bestaat (het is dus een tot-datum: de datum volgend op de dag waarop de relatie voor het laatst bestond). De *einddatum relatie* wordt opgenomen als een element binnen het element `tijdvakRelatie`, dat weer een element is binnen een fundamenteel of een relatie-entiteittype De datum wordt gecodeerd met acht cijfers in het formaat EEJJMMDD. Onvolledige datums worden gespecificeerd zoals hierboven beschreven bij de begindatum tijdvak geldigheid.
- *Sleutel in het verzendende systeem*
  Dit metagegeven bevat de sleutel in het verzendende systeem.
- *Sleutel in het ontvangende systeem*
  Dit metagegeven bevat de sleutel in het ontvangende systeem.
- *Sleutel in het gegevensbeheersysteem*
  Dit metagegeven bevat de sleutel van een gegeven in het gegevensbeheersysteem. Deze sleutel zorgt voor een systeemoverschrijdende identificatie van een object. Dit metagegeven mag voorkomen in fundamentele en relatie-entiteiten in zowel kennisgeving-, antwoord- als vraagberichten.
- *Verwerkingssoort*
  Dit in kennisgevingberichten verplichte metagegeven geeft aan op welke wijze de tot het entiteittype behorende gegevens verwerkt moeten worden. Het gebruik van dit metagegeven wordt uitgelegd in paragraaf 6.2. Het metagegeven heeft het volgende waardebereik:
  - T: Een entiteit wordt toegevoegd
  - V: Een entiteit wordt verwijderd
  - W: Gegevens van een entiteit worden gewijzigd of gecorrigeerd
  - E: Een relatie-entiteit wordt beëindigd
  - I: De entiteit bevat alleen identificerende gegevens
  - R: Een relatie-entiteit wordt vervangen door een nieuwe relatie-entiteit
  - S: De sleutel van een fundamentele entiteit wordt gewijzigd.

Onderstaande tabel geeft de maximale lengte, de implementatie als element of attribuut en het al dan niet verplicht zijn van de metagegevens.

Naam	Type	Element/ Attribuut	Optioneel/Verplicht
SoortEntiteit	Char	Attribuut	Verplicht
begindatumGeldigheid	Date	Element	Afhankelijk van de berichtinhoud
einddatumGeldigheid	Date	Element	Afhankelijk van de berichtinhoud
begindatumRelatie	Date	Element	Afhankelijk van de berichtinhoud en alleen in relatie-entiteiten
einddatumRelatie	Date	Element	Afhankelijk van de berichtinhoud en alleen in relatie-entiteiten
sleutelVerzendend	String[40]	Attribuut	Verplicht in kennisgevingberichten en asynchrone antwoordberichten, wanneer de sleutel beschikbaar is. Mag worden weggelaten als het zendende systeem de sleutel niet heeft.
sleutelOntvangend	String[40]	Attribuut	Optioneel
sleutelGegevensbeheer	String[40]	Attribuut	Optioneel
verwerkingssoort	Char	Attribuut	Verplicht in kennisgevingberichten

Tabel 2 Overzicht metagegevens per entiteit

In het XML-schema in bijlage 2 zijn de attributen van een fundamenteel, relatie- en tabelentiteittype gedefinieerd in de attribute groups `fundamenteel`, `relatie` en `tabel`. Ook het `complexType` voor het element `tijdvakGeldigheid` met de elementen `begindatumGeldigheid` en `einddatumGeldigheid` en het element `tijdvakRelatie` met de elementen `begindatumRelatie` en `einddatumRelatie` zijn daar gedefinieerd.

5.2.3 Het opnemen van niet in het sectormodel gedefinieerde elementen

De in het sectormodel gedefinieerde elementen en hun tags worden door alle partijen binnen de sector ofwel ondersteund zoals ze gedefinieerd zijn ofwel niet ondersteund. Soms zal een deel van de partijen in een sector met elkaar afspraken willen maken over elementen die niet voor anderen bindend zijn. Om hierin te voorzien zijn de tags `extraElement` en `extraElementen` gedefinieerd. Hieronder staan de corresponderende definities.

```
<complexType name="extraElement">
   <simpleContent>
    <extension base="string">
      <attribute name="elementnaam" type="string" use="required"/>
      <attribute name="exact" type="StUF:exact" default="true"/>
      <attribute name="noValue" type="StUF:noValue"/>
    </extension>
   </simpleContent>
</complexType>
<element name="extraElementen" type=”StUF:extraElementen” minOccurs=”0”>
   <sequence maxOccurs="unbounded">
      <element name=”extraElement” type=”StUF:extraElement” nillable=”true”
       minOccurs="0"/>
   </sequence>
</complexType>
```

Van deze elementnamen hoeft in het sectormodel geen definitie te worden opgenomen. De uitgegeven elementnaam wordt geregistreerd samen met de aanvrager en desgewenst een definitie of omschrijving. Een dergelijk elementnaam mag niet een tweede keer worden uitgegeven.

6 HET VULLEN VAN DE BERICHTBODY

We weten nu hoe de objecten eruit zien, maar de voorschriften om de objecten te vullen en om objecten in een bericht op te nemen ontbreken nog. Met kennis van alleen de syntax kun je nog geen berichten samenstellen. Je moet ook weten wanneer en hoe objecten en waarden in een bericht worden opgenomen. Dit hoofdstuk beschrijft de regels daarvoor.

6.1 Het opnemen van objecten, elementen en relatie-entiteittypen in een bericht

Of een object of een element in een bericht wordt opgenomen en zo ja hoe, is afhankelijk van een aantal factoren. Deze paragraaf gaat daar dieper op in.

Soms zal een ontvanger van een bericht entiteittypen of attributen aantreffen die hij niet kent. Deze entiteittypen en gegevens worden bij de verwerking van het bericht genegeerd. Het is dus niet noodzakelijk dat zender en ontvanger met precies dezelfde versie van een sectormodel werken. De ontvanger moet entiteiten en elementen negeren die hij niet kent. Bij een wijziging van een sectormodel hoeft dus niet bij alle gebruikers tegelijkertijd een nieuwe versie van de berichtverwerkende software te worden geïmplementeerd.

In alle gevallen waarin de elementinhoud leeg is vereist de syntax van XML Schema's dat aan het attribuut `xsi:nil` de waarde `"true"` wordt toegekend. Daarnaast stelt StUF de verplichting dat in zulke gevallen ook het attribuut `StUF:noValue` met een daaraan toegekende waarde (`"nietOndersteund"`, `"nietGeautoriseerd"`, `"waardeOnbekend"`, of `"geenWaarde"`) wordt opgenomen om aan te geven waarom het element is leeg gelaten.

Indien aan een element waarmee een datum wordt gedefinieerd een waarde wordt toegekend dan dient dit altijd een valide datum te zijn, dus ook als het attribuut `StUF:noValue` de waarde `"waardeOnbekend"`, of `"geenWaarde"` heeft.

6.1.1 Het opnemen van objecten in een bericht

Een object wordt alleen in een bericht opgenomen, indien voor minimaal één van de kerngegevens een waarde bekend is of als het attribuut `sleutelOntvangend` bekend is. Wanneer geen object in het bericht wordt opgenomen en het bericht toch verzonden moet worden, dan wordt een lege berichtbody opgenomen (`<body></body>` of `<body/>`). Een relatie-entiteit en het gerelateerde object worden alleen in een bericht opgenomen, indien voor het gerelateerde object minimaal voor één kerngegeven een waarde bekend is of als het attribuut `sleutelOntvangend` bekend is. Bij de bespreking van de regels voor de verschillende berichtsoorten wordt dieper ingegaan op het opnemen van entiteiten in een bericht.

6.1.2 Het opnemen van elementen in een entiteit

Er zijn redenen waarom van een element niet altijd een geldige waarde in een bericht kan worden opgenomen. Deze redenen worden onderscheiden met het attribuut `StUF:noValue` en worden besproken aan de hand van de beslisboom in Figuur 9.

De eerste beslissing is of het element door de zender wordt ondersteund. Als het niet wordt ondersteund, dan wordt gekeken of het verplicht is of niet. In de volgende gevallen moet een element verplicht in een bericht worden opgenomen:

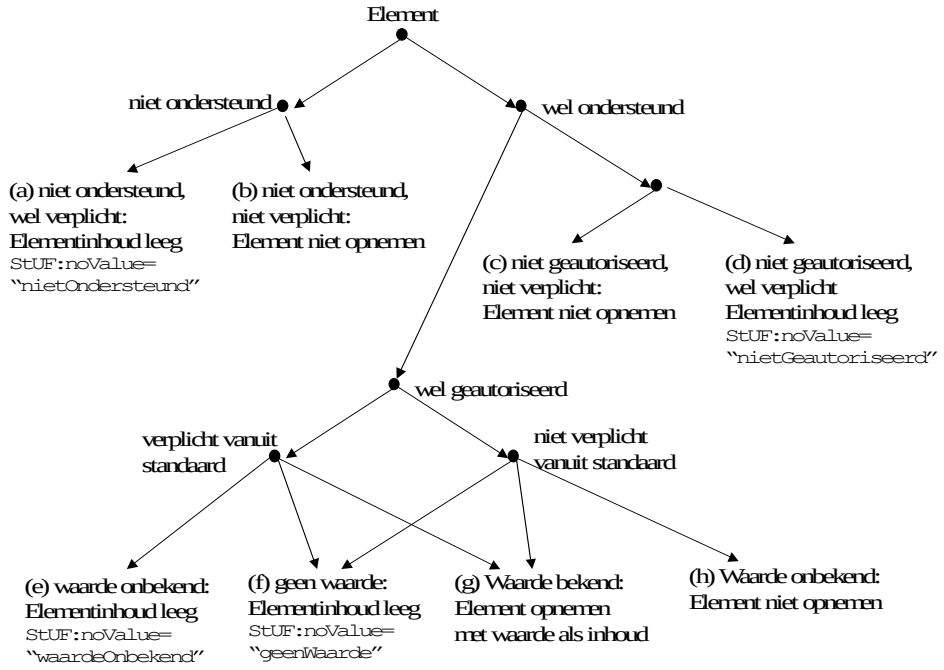
1. Het element is een kerngegeven en het bericht is een kennisgevingbericht, waarbij het attribuut `sleutelOntvangend` niet voorkomt in de entiteit;
2. Het element maakt deel uit van een gegevensgroep, waarvan minimaal één ander element met een waarde in het bericht wordt opgenomen en het bericht is een kennisgevingbericht;
3. Het element is gevraagd in het vraagbericht waarop het bericht een antwoord is;
4. Het sectormodel specificeert dat het element in een kennisgevingbericht of een antwoordbericht moet voorkomen.

Als het element verplicht is, dan wordt het opgenomen met het attribuut `StUF:noValue="nietOndersteund"` (a) en een lege elementinhoud. Een dergelijk element kan door de ontvanger worden genegeerd. Als het niet verplicht is, dan wordt een niet-ondersteund element niet opgenomen in het bericht (b).

Wanneer het element wel wordt ondersteund, wordt er gekeken of de ontvanger van het bericht geautoriseerd is voor dit element. Als het element niet verplicht is, dan wordt het niet opgenomen in het bericht (c). Als het verplicht is, dan wordt het niet-geautoriseerd zijn expliciet gemeld door het element op te nemen met het attribuut `StUF:noValue="nietGeautoriseerd"` (d) en een lege elementinhoud.

Indien de ontvanger van het bericht is geautoriseerd om het element te ontvangen, zijn er drie mogelijkheden voor de waarde van het element:
1. Element heeft geen waarde,
2. Element heeft onbekende waarde,
3. Element heeft geldige waarde

Als het element in de werkelijkheid geen waarde heeft, wordt het element in het bericht opgenomen voorzien van het attribuut `StUF:noValue="geenWaarde"` (f) en een lege elementinhoud. Indien het element wel een waarde heeft maar de waarde is bij de zender niet bekend en het element is verplicht, dan wordt dit met `StUF:noValue="waardeOnbekend"` (e) en een lege elementinhoud. Dit kan bijvoorbeeld gelden voor een kerngegeven als de geboortedatum van een persoon waarvan het zendende systeem niet altijd de waarde kent. Als het element niet verplicht is en de waarde is onbekend, dan wordt het niet in het bericht opgenomen (h). Als de waarde bekend is, dan wordt het element met zijn waarde als inhoud opgenomen (g).



Figuur 9: Het opnemen van een element in een bericht

6.1.3 Het opnemen van relatie-entiteit in een entiteit

Voor het in een entiteit opnemen van relatie-entiteiten gelden soortgelijke principes:

- *Het verzendende systeem ondersteunt een relatie niet en deze relatie is een kernrelatie, onderdeel van een gegevensgroep of gevraagd in een vraagbericht.*
  Het element voor de relatie-entiteit wordt opgenomen met het attribuut `StUF:noValue="nietOndersteund"` zonder elementinhoud.
- *Het ontvangende systeem is niet geautoriseerd voor een relatie bij een object en de relatie is een kerngegeven, onderdeel van een gegevensgroep of gevraagd in een vraagbericht*
  Het element voor de relatie-entiteit wordt opgenomen met het attribuut `StUF:noValue="nietGeautoriseerd"` zonder elementinhoud.
- *Het verzendende systeem weet dat een relatie niet voorkomt bij een object en de relatie is een kerngegeven, onderdeel van een gegevensgroep of gevraagd in een vraagbericht*
  Het element voor de relatie-entiteit wordt opgenomen met het attribuut `StUF:noValue="geenWaarde"` zonder elementinhoud.
- *Het verzendende systeem kent de relatie*
  De relatie wordt opgenomen met het element voor de relatie-entiteit gevuld met de eigen elementen en met een element voor het gerelateerde entiteittype dat ook weer elementen bevat.
- *Het verzendende systeem weet niet of de relatie al dan niet voorkomt en de relatie is een kerngegeven, een onderdeel van een gegevensgroep of gevraagd in een vraagbericht*

Het element voor de relatie-entiteit wordt opgenomen zonder elementinhoud en met het attribuut `StUF:noValue="waardeOnbekend"`.

- *Het verzendende systeem weet niet of de relatie al dan niet voorkomt en de relatie is niet verplicht*
  De relatie-entiteit wordt niet opgenomen.
- *Het verzendende systeem weet dat de relatie niet voorkomt en de relatie is niet verplicht*
  De relatie-entiteit wordt niet opgenomen.

6.2 Regels voor kennisgevingberichten

Kennisgevingberichten mogen alleen worden gebruikt om actuele gegevens te wijzigen of te corrigeren en niet voor het wijzigen c.q. corrigeren van historische gegevens. De verwerking van wijzigingen in historische gegevens is erg complex en er is bewust voor gekozen het doorvoeren van dergelijke wijzigingen niet te ondersteunen binnen de StUF-standaard. Al beëindigde relaties worden als actuele relaties beschouwd, tenzij ze vervangen zijn door een andere relatie. Na een verhuizing mogen wijzigingen op het historische verblijfsadres niet meer in een kennisgevingbericht worden opgenomen.

Een kennisgevingbericht bevat gegevens van precies één object van het entiteittype gespecificeerd in de header van het bericht. Bij een toevoeging of verwijdering bevat het bericht één occurrence met gegevens van dat object en bij een wijziging en correctie twee occurrences: de eerste met de oude waarden en de tweede met de nieuwe waarden. Hieronder worden de regels besproken voor het vullen van het element corresponderend met het fundamentele entiteittype uit de header van het bericht, van de elementen voor relatie-entiteiten en van de elementen voor gerelateerde entiteiten.

6.2.1 Het vullen van het element corresponderend met het entiteittype uit de header van het bericht

Het toevoegen, wijzigen en verwijderen van objecten van het entiteittype uit de header van het bericht is relatief eenvoudig. De exacte regels zijn verschillend voor een fundamenteel entiteittype en een tabelentiteittype.

Fundamenteel entiteittype

Bij een fundamenteel entiteittype in de header van het bericht kunnen zich volgende situaties voordoen:

1. Een object is toegevoegd in het zendende systeem.
2. Elementen van het object zijn gewijzigd in het zendende systeem
3. De sleutel in het zendende systeem is gewijzigd
4. Elementen van het object zijn gecorrigeerd in het zendende systeem
5. Het object is verwijderd uit het zendende systeem
6. Alleen relaties of gerelateerde entiteiten zijn gewijzigd

Een kennisgevingbericht met een fundamenteel entiteittype in de header van het bericht kan ook verstuurd worden zonder dat er gegevens in dat entiteittype zelf zijn veranderd. De wijzigingen zitten in relatie-entiteiten of in gerelateerde entiteiten. De entiteit uit de header van het bericht dient in dat geval alleen voor de identificatie. Voor dit geval is de verwerkingssoort ‘I’ (Identificatie) gedefinieerd.

Onderstaande tabel vat de regels samen voor de opbouw van het bericht. Deze regels hebben betrekking op het vullen van de elementen `begindatumGeldigheid` en `einddatumGeldigheid`, het attribuut `verwerkingssoort` en van de overige elementen binnen het element corresponderend met het entiteittype uit de header van het bericht. Het attribuut `sleutelVerzendend` is niet in de tabel opgenomen, omdat dit attribuut hoort te worden opgenomen als het zendende systeem de sleutel kent. Als het zendende systeem om wat voor reden dan ook geen sleutel heeft, dan wordt het attribuut `sleutelVerzendend` niet opgenomen. De attributen `sleutelOntvangend` en `sleutelGegevensbeheer` zijn niet in de tabel opgenomen omdat ze altijd optioneel zijn.

Soort kennisgeving	Mutatie soort	Oud/ Nieuw	begin-datum-Geldigheid	eind-datum-Geldigheid	verwer-kings-soort	Overige elementen
Toevoeging	T	Nvt	N	N	T	Alle bekende elementen
Wijziging	W	Oud	O	N	W	Kerngegevens als `sleutelOntvangend` ontbreekt + te wijzigen elementen
		Nieuw	N	N	W	Kerngegevens als `sleutelOntvangend` ontbreekt + gewijzigde elementen
Correctie	C	Oud	O	O	W	Kerngegevens als `sleutelOntvangend` ontbreekt + te corrigeren elementen
		Nieuw	N	N	W	Kerngegevens als `sleutelOntvangend` ontbreekt + gecorrigeerde elementen
Sleutelwijziging	C	Oud	-	-	S	Kerngegevens en zo mogelijk `sleutelOntvangend`
		Nieuw	-	-	S	Kerngegevens en zo mogelijk `sleutelOntvangend`
Verwijdering	V	Nvt	-	-	V	Kerngegevens als `sleutelOntvangend` ontbreekt
Identificatie	W of C	Oud	-	-	I	Kerngegevens als `sleutelOntvangend` ontbreekt
		Nieuw	-	-	I	Kerngegevens als `sleutelOntvangend` ontbreekt

Tabel 3 Regels voor de opbouw van een bericht

Mutatiesoort	De codes voor het veld mutatiesoort gedefinieerd in paragraaf 4.2	
`begindatumGeldigheid` en `einddatumGeldigheid` Als de één voorkomt is de ander verplicht!	-	Mag niet voorkomen
	O	Oorspronkelijke waarde, indien ondersteund
	N	Nieuwe waarde, indien ondersteund
verwerkingssoort	De codes voor het veld verwerkingssoort gedefinieerd in paragraaf 5.2.2	

Tabel 4 Legenda tabel: Regels voor de opbouw van een bericht

De elementen `begindatumGeldigheid` en `einddatumGeldigheid` mogen alleen opgenomen worden, als:
1. in het sectormodel voor het betreffende fundamentele entiteit gespecificeerd is dat historie relevant is.
2. de entiteit minstens één element bevat waarvoor in het sectormodel is gespecificeerd dat historie relevant is.

Een wijzigkennisgeving bevat twee occurences van het object, de oude occurrence en de nieuwe occurrence. De oude occurrence specificeert in het element `einddatumGeldigheid` de einddatum tijdvak geldigheid van het 'oude' gegeven. Dit is het enige gegeven in de 'oude' occurrence dat een nieuwe waarde mag hebben. In de 'nieuwe' occurrence geven de `begindatumGeldigheid` en de `einddatumGeldigheid` het tijdvak geldigheid van de nieuwe gegevens aan.

Bij een correctie wordt een foutieve waarde vervangen door de juiste waarde. Een correctie heeft geen impact op de begin- en einddatum van het tijdvak geldigheid. Alleen als de wijziging betrekking heeft op het ontstaan of verdwijnen van het betrokken object zal de corresponderende datum van het tijdvak veranderen. Een dergelijke correctie verandert ook het tijdvak geldigheid van de gegevens, omdat het tijdvak waarin het object bestaat verandert. Bij alle andere correcties is er in de werkelijkheid niets veranderd en verandert het tijdvak geldigheid dus ook niet.

Bij een sleutelwijziging en een verwijdering is er in de werkelijkheid niets gebeurd, zodat het tijdvak geldigheid niet verandert. Bij een sleutelwijziging wordt een sleutel vervangen en bij een verwijdering vindt het zendende systeem het object niet langer relevant. Ook bij het in het bericht opnemen van een occurrence voor identificatiedoeleinden is het tijdvak geldigheid niet relevant.

Tabelentiteittype
In een tabelentiteit kunnen alleen de mutatiesoorten 'T' (toevoegen), 'W' (wijzigen), en 'V' (verwijderen) voorkomen. Bij tabelentiteiten mag het element `tijdvakGeldigheid` niet voorkomen. Wanneer de begin- en einddatum geldigheid relevant zijn, dienen deze te zijn gedefinieerd als elementen binnen de tabelentiteit. Bij tabelentiteiten mogen ook de attributen `sleutelVerzendend`, `sleutelOntvangend` en `sleutelGegevensbeheer` niet voorkomen, omdat tabelentiteiten geen sleutels hebben. Bij een tabelentiteit kan de verwerkingssoort 'I' niet voorkomen.

6.2.2 Het vullen van relatie-entiteiten en gerelateerde entiteiten
Een wijziging kan ook betrekking hebben op relaties met andere objecten. Het afhandelen hiervan is minder eenvoudig, omdat een relatie op een aantal verschillende manieren kan wijzigen. De tabel volgend op onderstaande opsomming bevat de regels. Hieronder worden eerst de verschillende mogelijkheden opgesomd en waar nodig worden de regels in de tabel toegelicht.

1. E*en relatie wordt toegevoegd in een toevoegkennisgeving*
   In een toevoegkennisgeving kunnen in het kennisgevingbericht ook de relevante[5] relaties naar andere entiteittypen worden gespecificeerd. Er worden net zoveel occurrences van een gerelateerd object opgenomen als er relevante relaties zijn. Wanneer een persoon vier kinderen heeft, dan wordt vier keer de relatie-entiteit van een persoon naar zijn kinderen opgenomen.

2. *Een relatie wordt toegevoegd in een wijzig- of correctiekennisgeving*
   Het specificeren van een relatie geeft nu aan dat er iets is veranderd in het object: het object heeft een nieuwe relatie gekregen. In de 'oude' occurrence wordt de relatie-entiteit opgenomen met het attributen `StUF:noValue="geenWaarde"` en een lege elementinhoud. De 'nieuwe' occurrence bevat de toegevoegde relatie.

3. *De gegevens van een relatie-entiteit worden gewijzigd*
   Als een huwelijk wordt ontbonden, worden bijvoorbeeld de datum, plaats en reden van de huwelijksontbinding binnen het relatie-entiteittype huwelijk gewijzigd. Dit kan alleen gebeuren bij de mutatiesoort 'W' (Wijziging). Het betekent dat in de werkelijkheid eigenschappen van de relatie zijn gewijzigd. Een dergelijke wijziging kan alleen plaatsvinden bij een nog niet beëindigde relatie, omdat een beëindigde relatie in de werkelijkheid niet meer bestaat en derhalve ook niet meer gewijzigd kan worden. De begin- en einddatum relatie kunnen bij een wijziging nooit geraakt worden. De begindatum kan alleen door middel van een correctie veranderen en de einddatum kan alleen een waarde krijgen bij het beëindigen van een relatie of na het beëindigen van de relatie gecorrigeerd worden. De elementen `begindatumRelatie` en `einddatumRelatie` worden alleen in de relatie-entiteit opgenomen, als ze deel uitmaken van de kerngegevens. Omdat er bij een wijziging nog geen `einddatumRelatie` is, zal het element `einddatumRelatie` altijd met een lege elementinhoud met als attribuut `StUF:noValue="geenWaarde"` worden opgenomen.

4. *De gegevens van een relatie-entiteit worden gecorrigeerd*
   Dit kan alleen gebeuren bij de mutatiesoort 'C' (Correctie). Het betekent dat er in de werkelijkheid niets is gebeurd met het object, maar dat een administratieve fout in de gegevens wordt gecorrigeerd. Ook de `begindatumRelatie` en de `einddatumRelatie` kunnen worden gecorrigeerd. De `begindatumRelatie` en de `einddatumRelatie` worden dus in een correctie anders behandeld dan in een wijziging.

5. *Een relatie wordt verwijderd*
   Dit betekent dat de relatie niet langer relevant is voor het zendende systeem en daarom in het zendende systeem is verwijderd. Het feit dat een relatie wordt verwijderd, impliceert niet dat de relatie is beëindigd. De beëindiging van een relatie dient in een apart kennisgevingbericht te worden doorgegeven voorafgaand aan het kennisgevingbericht over de verwijdering.
   Omdat een verwijdering onderdeel is van een correctiekennisgeving, is er zowel een 'oude' als een 'nieuwe' occurrence. In de 'oude' occurrence wordt de te verwijderen relatie opgenomen en in de 'nieuwe' occurrence wordt de relatie opgenomen met als attribuut `StUF:noValue="geenWaarde"` en een lege elementinhoud.

6. *Een relatie wordt beëindigd*
   Dit betekent dat de relatie niet langer in de werkelijkheid bestaat, bijvoorbeeld het niet langer hebben van een bepaalde verblijfstitel. Dit kan alleen voorkomen bij de mutatiesoort 'W' in de header van het bericht. Het feit dat een relatie wordt beëindigd impliceert niet dat het zendende systeem de relatie niet meer relevant vindt. Als het zendende systeem ook wil doorgeven dat het de relatie niet meer relevant vindt, dan dient een tweede kennisgevingbericht te worden verstuurd waarin de relatie-entiteit is opgenomen met de verwerkingssoort 'V'.
   Bij een beëindiging mag alleen het element `einddatumRelatie` een nieuwe waarde krijgen. Deze nieuwe waarde wordt gespecificeerd in de 'oude' occurrence van de relatie. Een relatie kan overigens ook beëindigd worden voor relatie-entiteiten waarin de elementen `begindatumRelatie` en `einddatumRelatie` niet voorkomen. Het verzendende systeem kan dan niet specificeren op welk moment de relatie beëindigd is, maar dit wordt ook niet nodig geacht.
   Het tijdvak geldigheid mag bij een beëindiging niet gespecificeerd worden. De `einddatumGeldigheid` van de oorspronkelijke gegevens wordt door de beëindiging gelijk aan de `einddatumRelatie` evenals de begindatum tijdvak geldigheid van de nieuwe gegevens. De `einddatumGeldigheid` van de nieuwe gegevens bestaat niet. Het element wordt dus opgenomen met als attribuut

---

5 Reeds beëindigde relaties kunnen nog wel relevant zijn en worden daarom opgenomen in het kennisgevingbericht.

`StUF:noValue="geenWaarde"` en een lege elementinhoud.

7. *Een relatie wordt vervangen*
   Dit betekent dat de oorspronkelijke relatie wordt beëindigd en een nieuwe relatie begint. Een voorbeeld hiervan is de verhuizing van een persoon van het ene adres naar het andere. Bij een vervanging wordt de oorspronkelijke relatie een historisch gegeven van het object van waaruit de relatie ligt. Een vervanging heeft in StUF dus een iets andere betekenis dan het achtereenvolgens beëindigen van een relatie en het toevoegen van een nieuwe relatie, omdat het beëindigen van een relatie niet impliceert dat de beëindigde relatie historisch is geworden.
   De verwerkingssoort krijgt de waarde 'R'. Het element `begindatumRelatie` bevat in de 'oude' occurrence de oorspronkelijke begindatum relatie en het element `einddatumRelatie` de nieuwe einddatum relatie. In de 'oude' occurrence wordt het tijdvak geldigheid niet opgenomen, omdat het enige element dat wijzigt de einddatum relatie is. De einddatum tijdvak geldigheid van de oude relatie is gelijk aan de einddatum relatie. De 'nieuwe' occurrence bevat in de elementen `begindatumGeldigheid` en `einddatumGeldigheid` het tijdvak geldigheid van de nieuwe relatie en in de elementen `begindatumRelatie` en `einddatumRelatie` de begin- en einddatum relatie.

8. *Een relatie die in de werkelijkheid nooit heeft bestaan wordt vervangen of verwijderd*
   Een relatie die per abuis is toegevoegd, maar nooit heeft bestaan, is verwijderd in het zendende systeem. Omdat het hier gaat om een correctie, kan dit alleen voorkomen bij de mutatiesoort 'C' in de header van het bericht. Dit bericht heeft dezelfde opbouw als een verwijdering (er komt geen andere relatie voor in de plaats) of een vervanging (er komt wel een andere relatie voor in de plaats), alleen is de mutatiesoort nu 'C'.

9. *Identificatie*
   De relatie is alleen opgenomen in het bericht omdat deze deel uitmaakt van de kerngegevens van het object uit de header van bericht of omdat er gegevens in een gerelateerde entiteit worden gewijzigd. Met de relatie-entiteit zelf gebeurt niets. Bij een verwijdering van een fundamentele entiteit waarbij een gerelateerde entiteit onderdeel is van de kerngegevens, worden bijvoorbeeld de relatie-entiteit en de gerelateerde entiteit ter identificatie opgenomen.

Onderstaande tabel beschrijft in detail de regels voor het vullen van de attributen en elementen binnen een relatie-entiteit. Een relatie-entiteit bevat een gerelateerde entiteit. De vulling van de elementen van de gerelateerde entiteit wordt gespecificeerd na onderstaande tabel. De attributen `sleutelVerzendend`, `sleutelOntvangend` en `sleutelGegevensbeheer` van de relatie-entiteit zijn niet in deze tabel opgenomen, omdat deze attributen altijd optioneel zijn wanneer ze vanuit de logica van het bericht kunnen voorkomen. Net zoals bij fundamentele entiteiten worden de kerngegevens van de relatie niet in het bericht opgenomen als de sleutel in het ontvangende systeem gespecificeerd is.

Standaard mag in StUF voor de gerelateerde entiteit alleen verwerkingssoort 'I' of 'T' gedefinieerd worden. De onderstaande tabel gaat hiervan uit. Bij verwerkingssoort 'I' in de gerelateerde entiteit mogen de elementen `begindatumGeldigheid` en `einddatumGeldigheid` niet voorkomen. Zij zijn daarom niet in de tabel opgenomen. De attributen `sleutelOntvangend` en `sleutelGegevensbeheer` zijn ook niet in de tabel opgenomen onder de gerelateerde entiteit. Deze attributen zijn optioneel, terwijl `sleutelVerzendend` verplicht is, als het zendend systeem erover beschikt. Als het attribuut `sleutelOntvangend` niet is opgenomen, dan moeten de kerngegevens die de gerelateerde entiteit identificeren worden opgenomen.
In het sectormodel kan gespecificeerd worden dat een gerelateerde entiteit ook mag voorkomen met verwerkingssoort 'W'. In paragraaf 6.2.3 wordt aangegeven hoe de gerelateerde entiteit dan gevuld dient te worden.

Soort kennisgeving	Mut. soort	Oud/ Nieuw	Entiteit uit header	Relatie-entiteit						Gerelateerde entiteit		
			verwerkings-soort	begindatum-Geldigheid	einddatum-Geldigheid	verwerkings-soort	begindatum-Relatie	einddatum-Relatie	Rest element-inhoud	sleutel-Verzendend	verwerkings-soort	Element-inhoud
				Deze twee datums zijn van toepassing indien voor minstens één element in de relatie-entiteit historie gedefinieerd is in het sectormodel. Bij de verwerkingssoorten T en R worden deze datums dan altijd gevuld. In geval van verwerkingssoort W zijn de datums alleen gevuld als de waarde wijzigt van een element waarvoor in het sectormodel historie is gedefinieerd.			Deze twee datums zijn van toepassing als voor de relatie-entiteit historie gedefinieerd is. Ze worden opgenomen bij de verwerkingssoorten T, R, E, V. Bij verwerkingssoort W worden ze alleen opgenomen als de mutatiesoort C is of als de `begindatumRelatie` een kerngegeven is.					
Toevoegen relatie bij toevoegen object	T	Nvt	T	N	N	T	N	N	Alles	V	I	K/sleutel
Toevoegen relatie bij wijzigen object	W/C	Oud	W/I	-	-	T	-	-	Leeg	-	-	-
		Nieuw	W/I	N	N	T	N	N	Alles	V	I	K/Sleutel
Wijzigen relatie	W	Oud	W/I	O	N	W	K, O	K, Leeg	`K/Sleutel` + te wijzigen geg.	V	I	K/Sleutel
		Nieuw	W/I	N	N	W	K, O	K, Leeg	`K/Sleutel` + gewijzigde geg.	V	I	K/Sleutel
Corrigeren relatie	C	Oud	W/I	O	O	W	C, O	C, O	`K/Sleutel` + te corrig. geg.	V	I	K/Sleutel
		Nieuw	W/I	N	N	W	C, N	C, N	`K/Sleutel` + gecorrig. geg.	V	I	K/Sleutel
Verwijderen relatie	W[6]	Oud	W/I	-	-	V	K, O	K, Leeg	`K/Sleutel`	V	I	K/Sleutel
		Nieuw	W/I	-	-	V	-	-	Leeg	-	-	-
Beëindigen relatie	W	Oud	W/I	-	-	E	O	N	`K/Sleutel`	V	I	K/Sleutel
		Nieuw	W/I	-	-	E	-	-	Leeg	-	-	-
Vervangen relatie	W/C	Oud	W/I	-	-	R	O	N	`K/Sleutel`	V	I	K/Sleutel
		Nieuw	W/I	N	N	R	N	N	Alles	V	I	K/Sleutel
Correctie van het toevoegen van een relatie	C	Oud	W/I	-	-	V	K, O	K, O	`K/Sleutel`	V	I	K/Sleutel
		Nieuw	W/I	-	-	V	-	-	Leeg	-	-	-
Identificatie	W, C, V	Oud	W/I/V/S	-	-	I	K, O	K, O	`K/Sleutel`	V	I	K/Sleutel
		Nieuw	W/I/V/S	-	-	I	K, O	K, O	`K/Sleutel`	V	I	K/Sleutel

Tabel 5 Regels voor het vullen van de attributen en elementen binnen een relatie-entiteit

6 Een verwijdering zit in een kennisgeving met mutatiesoort “W” om hem te kunnen onderscheiden van een correctie van het toevoegen van een relatie

Onderstaande tabel geeft de legenda voor deze tabel:

Mutatiesoort	De codes voor het veld mutatiesoort gedefinieerd in paragraaf 4.2	
Verwerkingssoort entiteit uit header bericht	T	Een toevoeging van de entiteit uit de header van het bericht
	W/I	Wijziging of identificatie van de entiteit uit de header van het bericht
	C/I	Correctie of identificatie van de entiteit uit de header van het bericht
	W/I/V/S	Wijziging, identificatie, verwijdering of sleutelwijziging van de entiteit uit de header van het bericht
`begindatumGeldigheid` en `einddatumGeldigheid` NB: Als de één voorkomt is de ander verplicht!	-	Mag niet voorkomen
	O	Oorspronkelijke waarde, als het attribuut wordt ondersteund
	N	Nieuwe waarde, als het attribuut wordt ondersteund
Verwerkingssoort	De codes voor het veld verwerkingssoort gedefinieerd in paragraaf 5.2.2	
`begindatumRelatie` en `einddatumRelatie` NB: Als de één voorkomt is de ander verplicht!	-	Mag niet voorkomen
	O	Oorspronkelijke waarde
	N	Nieuwe waarde
	K	Alleen opnemen indien het element een kerngegeven is
	Leeg	Opnemen met lege elementinhoud en met als attribuut `StUF:noValue="geenWaarde"`
	C	Opnemen indien kerngegeven of indien `begindatumRelatie` of `einddatumRelatie` wordt gecorrigeerd
Relatie-entiteit Rest elementinhoud	Leeg	Neem de relatie-entiteit op met als attribuut `StUF:noValue="geenWaarde"` en met een lege elementinhoud. Van de gerelateerde entiteit wordt dan dus niets opgenomen.
	K/Sleutel	Opnemen zonder kerngegevens als `sleutelOntvangend` voorkomt en met de kerngegevens als `sleutelOntvangend` ontbreekt.
`sleutelVerzendend`	-	Mag niet voorkomen
	V	Verplicht, als het verzendend systeem over deze sleutel beschikt
Elementinhoud gerelateerde entiteit	K/sleutel	Opnemen zonder kerngegevens als `sleutelOntvangend` voorkomt en met de kerngegevens als `sleutelOntvangend` ontbreekt.
	-	Het element komt niet voor

Tabel 6 Legenda Tabel: Regels voor het vullen van de attributen en elementen binnen een relatie-entiteit

6.2.3 Toevoegen/wijzigen gerelateerde entiteit

Een gerelateerd object mag als onderdeel van een relatie worden toegevoegd. Het gaat hier om de volgende gevallen:

1. Toevoegen van een relatie tijdens het toevoegen van een object
2. Toevoegen van een relatie tijdens het wijzigen van een object
3. Vervangen van een relatie tijdens het wijzigen van een object

Denk bijvoorbeeld aan het door de GBA in het bericht opnemen van partnergegevens bij het specificeren van een huwelijk bij een persoon. De GBA kent de partner niet als een onafhankelijke persoon. In deze gevallen dient in het gerelateerde object ‘T’ als verwerkingssoort te worden gespecificeerd en gelden voor de gerelateerde entiteit dezelfde regels als voor een fundamentele entiteit uit de header van het bericht met inachtneming van de eventuele beperkingen gedefinieerd in het sectormodel.
Bij het wijzigen of corrigeren van een relatie mag in het sectormodel worden gespecificeerd dat als onderdeel van de wijziging ook gegevens van het gerelateerde object mogen worden gewijzigd. Dit gebeurt bijvoorbeeld als bij een huwelijk de naam van de echtgenoot wordt gewijzigd. Het huwelijk (PRSPRSHUW) blijft hetzelfde, maar de gegevens van de gerelateerde persoon wijzigen. Een dergelijke wijziging in de gegevens van de partner mag alleen worden doorgegeven als dit expliciet in het sectormodel is gedefinieerd. Zo niet, dan moet een wijziging in de gerelateerde entiteit worden doorgegeven als een wijziging in het fundamentele entiteittype zelf. Bij het wijzigen of corrigeren van gegevens in het gerelateerde object dient als verwerkingssoort ‘W’ te worden gespecificeerd en gelden verder dezelfde regels als voor het wijzigen van een fundamentele entiteit met inachtneming van de beperkingen gedefinieerd in het sectormodel.

Tabel 5 specificeert het vullen van de verwerkingssoort in de gerelateerde entiteit.

Soort kennisgeving	Mutatie-soort	Oud/ Nieuw	Verwerkingssoort		
			Entiteit uit header	Relatie-entiteit	Gerelateerde entiteit
Toevoegen relatie bij toevoegen object	T	Nvt	T	T	T
Toevoegen relatie bij wijzigen object	W	Oud	W/I	T	T
		Nieuw	W/I	T	T
Wijzigen gerelateerd object	W	Oud	W/I	I/W	W/I
		Nieuw	W/I	I/W	W/I
Corrigeren gerelateerd object	C	Oud	W/I	I/W	W/I
		Nieuw	W/I	I/W	W/I
Vervangen relatie	W	Oud	W/I	R	I
		Nieuw	W/I	R	T

Tabel 7 Invullen attribuut `verwerkingssoort`

6.2.4 Overige regels

- In elementen corresponderend met fundamentele entiteittypen dient in een kennisgevingbericht het attribuut `sleutelVerzendend` te worden opgenomen, als het verzendende systeem over deze sleutel beschikt. Indien het verzendende systeem geen sleutel heeft voor zo'n entiteittype, dan wordt het attribuut `sleutelVerzendend` niet opgenomen.
- De `sleutelVerzendend` van een object moet in zowel het oude als het nieuwe voorkomen in een kennisgevingbericht voor een wijziging of een correctie gelijk zijn.
- De elementen met de oude en nieuwe waarden dienen in exact dezelfde volgorde in het bericht te worden opgenomen.
- Indien in een kennisgevingbericht van een wijziging een element meer dan één keer mag voorkomen, dan worden bij het toevoegen, wijzigen of verwijderen van de waarde van dat element alle waarden voor dat element in het bericht opgenomen. Bij het toevoegen van een tweede telefoonnummer wordt het huidige telefoonnummer dus zowel in de 'oude' als de 'nieuwe' occurrence opgenomen.
- Indien een kennisgevingbericht van een wijziging of correctie meerdere relatie-entiteiten bevat, dan dienen deze relatie-entiteiten in exact dezelfde volgorde in de occurrence met de 'oude' waarden en in de occurrence met de 'nieuwe' waarden te worden opgenomen.
- Indien een kennisgevingbericht van een wijziging of correctie meerdere voorkomens van een bepaalde relatie-entiteit bevat, dan dienen deze voorkomens in dezelfde volgorde te staan in de occurrences met de oude en de nieuwe waarden. Het is dus toegestaan om de gegevens van twee verschillende huwelijken in één kennisgevingbericht te wijzigen.
- Eén en hetzelfde voorkomen van een relatie-entiteit mag maar één keer in een kennisgevingbericht worden opgenomen. Het is dus niet toegestaan om in een kennisgevingbericht bijvoorbeeld voor een huwelijk eerst de sluiting op te nemen en vervolgens in een andere relatie-entiteit ook nog eens de ontbinding. Deze gegevens dienen ofwel te worden opgenomen in één occurrence `PRSPRSHUW` ofwel in twee verschillende kennisgevingberichten.

6.3 Regels voor vraagberichten

Bij het stellen van een vraag aan een ander systeem kan de vraagsteller aangeven van welke objecten, in welke volgorde hij welke gegevens wil ontvangen. In de header van het bericht wordt door middel van het stuurgegeven *entiteittype* gespecificeerd om welk type objecten het gaat en door middel van het stuurgegeven *sortering* in welke volgorde de occurrences worden verwacht. In de berichtbody kan worden gedefinieerd welke objecten precies moeten worden teruggegeven. Een vraagbericht over personen kan bijvoorbeeld vragen om personen met een bepaalde achternaam, om personen die op een bepaald adres wonen of om personen die in een bepaald jaar geboren zijn. In de berichtbody kan ook gespecificeerd worden welke gegevens moeten worden teruggegeven.

Ter illustratie en verduidelijking is in Bijlage 2: voorbeeldberichten een voorbeeld gegeven van een StUF-vraagbericht.

6.3.1 Het specificeren van selectiecriteria

Voor het definiëren van selecties is een simpel mechanisme gekozen. Selectiecriteria kunnen gedefinieerd worden op willekeurige velden in het antwoordbericht. Een object voldoet alleen, als tegelijkertijd aan alle selectiecriteria wordt voldaan. In StUF kan dus alleen met de boolean operator 'en' gewerkt worden en niet met een combinatie van de boolean operators 'en' en 'of'. StUF schrijft niet voor dat iedere applicatie willekeurige selecties moet ondersteunen. De velden uit de in de header gedefinieerde sortering mogen in elk geval als selectiecriterium worden gebruikt. In het sectormodel wordt per systeem dat via StUF te benaderen is gedefinieerd welke velden nog meer als selectiecriterium gebruikt mogen worden. Indien een selectiecriterium wordt gespecificeerd dat niet wordt ondersteund door het systeem, dan wordt dat selectiecriterium genegeerd en wordt de foutmelding StUF008 teruggestuurd (zie Tabel 8: StUF-foutberichten).

Er is voor een simpel mechanisme gekozen om de implementatie van selecties niet te complex te maken en om de responstijd bij selecties te kunnen garanderen. StUF heeft niet de ambitie om een nieuwe syntax voor het definiëren van selecties te introduceren. Omwille van het uitgangspunt dat introductie van StUF niet mag leiden tot grote aanpassingen in de bestaande systemen is ervan afgezien een bestaand mechanisme als SQL binnen StUF op te nemen. Als de door StUF ondersteunde selectie niet verfijnd genoeg is, dan is het aan de vragende partij om extra selecties uit te voeren op de ontvangen occurrences. Deze keuze maakt de implementatie van de standaard simpel en is afdoende voor synchrone vragen bedoeld om interactief een aantal occurrences te tonen in een lijst of om precies één occurrence te selecteren. Voor complexe vragen (meestal asynchrone vragen) biedt StUF nu mogelijk niet voldoende functionaliteit. De praktijk zal dit uitwijzen.

Selectiecriteria worden gespecificeerd door in de body van het vraagbericht twee occurrences op te nemen van het in de header gespecificeerde entiteittype. De eerste occurrence bevat de 'vanaf'-selectiecriteria en de tweede de 'tot-en-met'-selectiecriteria. Een 'is gelijk aan'-selectie wordt gespecificeerd door in de 'tot-en-met'-occurrence dezelfde waarden op te nemen als in de 'vanaf'-occurrence. Het is uiteraard ook mogelijk om helemaal geen selectie te definiëren. In dat geval wordt in de berichtbody alleen twee keer een element voor het entiteittype in de header van het bericht opgenomen met een lege elementinhoud en de attributen `xsi:nil="true"` en `StUF:noValue="geenWaarde"`.

Een occurrence met selectiecriteria kan zijn opgebouwd uit verschillende entiteittypen. Bij het zoeken van personen op een bepaald adres bestaat de berichtbody bijvoorbeeld uit een element voor het entiteittype persoon met als enige inhoud een element voor het relatie-entiteittype PERSOON.verblijft op.ADRES met daarin als enige inhoud een element voor het entiteittype adres. In de 'tot en met'-occurrence wordt dit alles nog eens met dezelfde inhoud in het bericht opgenomen om een 'is gelijk aan'-selectie te specificeren.

Voor selectiecriteria waarvoor een *vanaf*- of *tot-en-met*-selectie is gedefinieerd kunnen low of high values gespecificeerd worden door het element met lege inhoud op te nemen. Wanneer zowel in de 'vanaf'- als de 'tot-en-met'-selectie een veld met lege inhoud en en de attributen `xsi:nil="true"` en `StUF:noValue="geenWaarde"` worden opgenomen, dan heeft dit niet als betekenis dat alle waarden voor dat veld zijn toegestaan, maar dat het veld leeg moet zijn. Het toegestaan zijn van alle waarden kan namelijk gespecificeerd worden door het selectiecriterium helemaal niet op te nemen. Dit is bijvoorbeeld zinnig om bij een adres te kunnen specificeren dat wordt gezocht op een huisnummer zonder huisletter, dat wil zeggen alleen Wal 9 en niet ook Wal 9a en Wal 9b.

Bij het definiëren van selecties is het gemakkelijk te kunnen aangeven dat alleen een gedeelte van de waarde overeen hoeft te stemmen. Dit is bijvoorbeeld nuttig als je een persoon zoekt maar niet weet of hij Jansen heet of Janssen. Door te zoeken op Jans waarbij alle personen worden teruggegeven met een naam die begint met Jans worden zowel de Jansen's als de Janssen's teruggeven. Bij het definiëren van de selectiecriteria wordt aangegeven dat niet exact op Jans wordt gezocht door bij het element voor het selectiecriterium als attribuut op te nemen `StUF:exact="false"`. Wanneer het attribuut `StUF:exact` niet voorkomt of de waarde '`true`' heeft, dan wordt ervan uitgegaan dat het element exact de opgegeven waarde moet hebben. Bij het definiëren van het vraagbericht in het sectormodel dient het attribuut `<attribute ref="StUF:exact"/>` te worden opgenomen bij alle selectiecriteria waarop met niet-exacte waarden geselecteerd mag worden. Het attribuut `StUF:exact` is gedefinieerd in Bijlage 1: Het Generieke XML Schema voor StUF-berichten.

6.3.2 Het specifieren van de sortering

In de header van het bericht kan in het stuurgegeven *sortering* de gewenste sortering worden ingevuld. In het sectormodel is gespecificeerd welke sorteringen mogelijk zijn. Als niets of '0' wordt ingevuld, dan geeft het antwoordende systeem de objecten terug in zijn standaard sortering.

Sorteringen voor een fundamenteel entiteittype worden in het sectormodel gespecificeerd door een getal tussen de 1 en de 99 (de code van de sortering). Voor elke sorteringscode worden de elementen en relaties van het fundamentele entiteittype opgesomd waarop achtereenvolgens oplopend of aflopend gesorteerd moet worden. Indien op een relatie gesorteerd moet worden, dan dient daarvoor ook de sorteervolgorde gespecificeerd te worden door aan te geven op welke elementen en relaties van de relatie en de daarvan gerelateerde entiteiten er oplopend of aflopend gesorteerd moet worden

6.3.3 Het bevragen op sleutel

Soms zal het vragende systeem de sleutel van een gevraagde occurrence in het antwoordende systeem registreren of verwacht het vragende systeem dat het antwoordende systeem de sleutels in het vragende systeem registreert. Om hiervan gebruik te maken kan een systeem een ander systeem ook op sleutel bevragen: één of meer objecten worden gespecificeerd door in het vraagbericht een sleutel of een range van sleutels mee te geven in plaats van een

selectiecriteria. Bij bevragen op sleutel wordt het attribuut `sleutelVerzendend` of `sleutelOntvangend` in de 'vanaf'- en 'tot-en-met'-occurrence meegegeven. Het element heeft verder geen inhoud. Bij een selectie op sleutelwaarde wordt een eventueel in de header gespecificeerde sortering genegeerd. De occurrences die voldoen aan de selectiecriteria mogen in een willekeurige volgorde worden teruggeven.

Bij het selecteren op sleutel kunnen zich de volgende foutsituaties voordoen:
1. Indien zowel de sleutel in het ontvangende systeem als de sleutel in het verzendende systeem worden meegegeven, dan wordt als antwoord het foutbericht StUF006 gegeven (zie Tabel 8).
2. Wanneer het ontvangende systeem ondervraagd wordt op sleutel in het verzendende systeem en de sleutel in het verzendende systeem niet registreert bij het gevraagde entiteittype, dan wordt als antwoord het foutbericht StUF 007 gegeven (zie Tabel 8) .

6.3.4 Het specificeren van de gevraagde gegevens

In de berichtbody kan na de selectiecriteria worden gespecificeerd welke gegevens moeten worden teruggegeven. Hiertoe wordt in de berichtbody na de twee occurrences ter specificatie van de selectiecriteria een derde occurrence van het in de header gespecificeerde entiteittype opgenomen. De inhoud van deze occurrence bevat precies de velden die gevraagd worden in de vorm van elementen met een lege inhoud en de attributen `xsi:nil="true"` en `StUF:noValue="geenWaarde"`. Naar attributen kan niet gevraagd worden. Ook naar het element `tijdvakGeldigheid` kan niet gevraagd worden. Naar de elementen `begindatumRelatie` en `einddatumRelatie` in relatie-entiteittypen hoeft niet gevraagd te worden als het stuurgegeven indicator historisch op '`true`' staat. In dat geval worden ze sowieso geleverd. Als de indicator historisch op '`false`' staat, dan worden ze alleen geleverd als ze expliciet gevraagd worden.

Ook een dergelijk occurrence met de gevraagde gegevens kan zijn opgebouwd uit verschillende entiteittypes. Wanneer van een persoon de naamsgegevens, de geboortedatum en de adresgegevens worden gevraagd, ziet de occurrence met de gevraagde gegevens er als volgt uit: het element corresponderend met het entiteittype *persoon* bevat de elementen voor de geslachtsnaam, de voorvoegsels, de voorletters en de geboortedatum met een lege inhoud. Daarnaast bevat het entiteittype *persoon* het element voor het relatie-entiteittype PERSOON.verblijft op.ADRES. Het element voor PRSADR bevat alleen het element voor het entiteittype *adres*, dat de elementen voor de gevraagde adresgegevens bevat met een lege inhoud.

Als het vragende systeem de gevraagde gegevens niet kan of wil specificeren, dan wordt in het bericht een element voor het entiteittype uit de header van het bericht opgenomen met een lege inhoud. In dat geval mag het antwoordende systeem alleen de kerngegevens teruggeven.

6.3.5 Het stellen van een vervolgvraag

Tot nu toe is er stilzwijgend vanuit gegaan dat het ging om een vraag die voor het eerst werd gesteld. Bij de beschrijving van de functionaliteit in de header hebben we gezien dat het kan gebeuren dat niet alle occurrences die aan de selectiecriteria voldoen worden teruggegeven. Door het sturen van een nieuw vraagbericht met de waarde '`true`' in het stuurgegeven *indicator vervolgvraag* kan om meer occurrences gevraagd worden.

Omdat uitgangspunt van StUF is dat berichten onafhankelijk van elkaar verwerkt moeten kunnen worden, hoeft het antwoordende systeem geen informatie meer te hebben over het eerder verzonden antwoord. In een vervolgvraag dient daarom te worden gespecificeerd wat de laatste ontvangen occurrence was. Dit wordt gedaan door na de specificatie van de gevraagde gegevens in de body van de vervolgvraag de occurence in het laatste ontvangen antwoordbericht op te nemen. De body van een vervolgvraag is dus de body van het eerste vraagbericht gevolgd door een element met de laatst teruggegeven occurrence.

6.4 Regels voor antwoordberichten

In een antwoordbericht dat wordt verzonden naar aanleiding van een vraagbericht moeten altijd alle gevraagde gegevens worden teruggegeven die worden ondersteund en waarvoor het vragende systeem geautoriseerd is. Bij het niet ondersteunen van een gevraagd gegeven wordt het element teruggegeven met een lege inhoud en het attribuut `StUF:noValue="nietOndersteund"`. Als een waarde niet bekend is, dan wordt een element zonder inhoud teruggegeven met het attribuut `StUF:noValue="waardeOnbekend"`. Als het antwoordende systeem zeker weet dat het element niet voorkomt, dan wordt het element zonder inhoud teruggegeven en met het attribuut `StUF:noValue="geenWaarde"`.

Bij het niet ondersteunen van een vraag zijn de volgende specifieke foutberichten gedefinieerd: StUF002, StUF004, StUF006, StUF007, StUF008, StUF009, StUF012 en StUF013 (zie Tabel 8). Afhankelijk van de reden kan er dan één van deze meldingen gebruikt worden om de fout te identificeren. Als geen van bovengenoemde foutmeldingen van toepassing is, kan er besloten worden om het meer algemene foutbericht StUF003 terug te gegeven.

Als het gaat om autorisatie voor het opvragen van gegevens zijn er twee niveau's te onderscheiden:

- *Objectniveau*. Er kan geen antwoord gegeven worden omdat het vragende systeem niet geautoriseerd is om het object te bevragen.
- *Attribuut/relatie-niveau*. Het vragende systeem is weliswaar geautoriseerd voor het object, maar is niet geautoriseerd voor alle attributen en relaties van het object.

Wanneer een vragend systeem niet op objectniveau geautoriseerd is dient het foutbericht StUF010 te worden teruggezonden door het antwoordende systeem. In het geval een vragend systeem niet geautoriseerd is voor bepaalde attributen/relaties van een wel geautoriseerd object, dan dient `StUF:noValue="nietGeautoriseerd"` te worden opgenomen bij die attributen/relaties in het antwoordbericht.

Dezelfde regels als bij kennisgevingen gelden voor het opnemen van relatie-entiteiten in antwoordberichten. In een antwoordbericht dient in het element corresponderend met het fundamentele entiteittype of tabelentiteittype waarop de vraag betrekking heeft altijd het attribuut `sleutelVerzendend` gevuld te worden met de sleutel in het antwoordende (=verzendende) systeem. Als de gevraagde gegevens niet zijn gespecificeerd, dan worden de kerngegevens teruggegeven. Voor asynchrone antwoordberichten zonder direct bijbehorend vraagbericht dient in het sectormodel gespecificeerd te worden welke gegevens in het bericht moeten worden opgenomen.

Synchrone antwoordberichten bevatten alle gevraagde occurrences in één bericht. Asynchrone antwoordberichten mogen slechts één occurrence per bericht bevatten. Meerdere occurrences moeten dus worden verzonden in meerdere asynchrone antwoordberichten. Het voordeel hiervan is dat bij een fout in de verwerking van één van de occurrences duidelijk is welke berichten wel correct verwerkt zijn en welke niet.

Als om meerdere objecten is gevraagd, dan dienen deze objecten terug te worden gegeven in de volgorde gespecificeerd in het stuurgegeven *sortering*. Als het stuurgegeven *sortering* een sortering bevat die het ontvangende systeem niet ondersteunt, dan wordt het foutbericht StUF004 teruggezonden. Het is namelijk niet noodzakelijk dat een systeem alle in het sectormodel gespecificeerde sorteringen ondersteunt.

6.4.1 Het afhandelen van bijzondere situaties

Bij het afhandelen van vraagberichten zijn de volgende bijzondere situaties onderkend:

- Indien het ontvangende systeem het afhandelen van asynchrone vragen niet ondersteunt, dan wordt foutbericht StUF012 teruggezonden.
- Indien de syntax van de body van het vraagbericht onjuist is of de stuurgegevens in de header van het bericht niet correct zijn gevuld, dan wordt het foutbericht StUF001 teruggezonden.
- Indien het ontvangende systeem onbekend is of de gebruiker binnen het ontvangende systeem (ontvangende organisatie, ontvangende applicatie, ontvangende administratie en ontvangende gebruiker uit de stuurgegevens), dan wordt het foutbericht StUF009 teruggezonden.
- Indien in geval van een asynchroon vraagbericht het vragende systeem onbekend is of de gebruiker binnen het vragende systeem (vragende organisatie, vragende applicatie, vragende administratie en vragende gebruiker uit stuurgegevens), dan kan geen foutbericht worden teruggezonden. In dit geval dient het asynchrone vraagbericht op een foutenlog gemeld te worden. Bij een synchroon vraagbericht wordt in geval van een onbekend vragend systeem een foutbericht StUF013 gestuurd.
- Indien het vragende systeem niet geautoriseerd is voor de gevraagde gegevens, dan wordt foutbericht StUF010 gegeven.
- Indien het interactieve proces voor het afhandelen van een synchrone vraag niet beschikbaar is, dan wordt het foutbericht StUF002 teruggezonden.
- Indien de gevraagde gegevens niet beschikbaar zijn, dan wordt het foutbericht StUF003 teruggezonden. Denk hierbij aan bijvoorbeeld het vragen naar een entiteittype dat het antwoordende systeem niet kent.
- Indien voor een occurrence voor geen van de gevraagde elementen een waarde beschikbaar is, dan wordt deze occurrence niet opgenomen in het antwoordbericht. Bij een adres kan bijvoorbeeld alleen om de locatieomschrijving worden gevraagd. In het antwoordbericht worden nu alleen adressen opgenomen die daadwerkelijk een locatieomschrijving hebben. Er worden geen occurrences opgenomen met als attribuut `sleutelVerzendend` en als elementinhoud een element `locatieOmschrijving` met een lege inhoud.
- Indien er geen occurrences zijn die aan de gevraagde selectie voldoen, dan wordt een antwoordbericht teruggezonden met een lege berichtbody.
- Indien beantwoording van een vraag teveel systeemresources vraagt, dan zendt het antwoordende systeem het foutbericht StUF008. Door het zenden van een StUF008 kan het antwoordende systeem voorkomen dat de

responstijd van het systeem nadelig wordt beïnvloed. Tevens geeft het antwoordende systeem hiermee aan het vragende systeem te kennen, dat de vraag waarschijnlijk beter op een andere manier gesteld kan worden.

- Per entiteittype en per soort vraagbericht (synchroon of asynchroon) kan het antwoordende systeem een maximum aantal terug te zenden objecten kennen. Indien bij de beantwoording van een vraag meer dan dat aantal objecten moet worden teruggegeven, dan zendt het antwoordende systeem het maximale aantal objecten, het veld *foutmelding* wordt gevuld met StUF002 en de indicator vervolgvraag wordt op '`true`' gezet.

6.4.2 Het omgaan met historische gegevens

Indien in de stuurgegevens het element `indicatorHistorisch` de waarde '`false`' heeft of niet voorkomt, dan mogen alleen de actuele gegevens in het bericht worden opgenomen. Dat wil zeggen in het bericht wordt één occurrence van het in de header gevraagde entiteittype opgenomen met de nu geldende gevraagde gegevens. Verder worden alle gevraagde relaties opgenomen waarvoor het element `einddatumRelatie` geen waarde heeft, of een waarde die in de toekomst ligt of waarbij het element `einddatumRelatie` niet voorkomt in het relatie-entiteittype. In geen van de entiteiten in het bericht wordt een `tijdvakGeldigheid` opgenomen. Bij relaties wordt het `tijdvakRelatie` opgenomen, als hierom gevraagd is. Bij een persoon worden bijvoorbeeld alle huwelijken meegezonden als de ontbindingsdatum van een huwelijk niet wordt gecodeerd met het element `einddatumRelatie`.

Indien in de stuurgegevens het element `indicatorHistorisch` de waarde '`true`' heeft, dan worden niet alleen de actuele gegevens in het bericht opgenomen, maar ook de historische gegevens. Het `tijdvakGeldigheid` is verplicht in die fundamentele en relatie-entiteiten waarbinnen gevraagd wordt om minstens één element waarvoor in het sectormodel historie is gedefinieerd. In relatie-entiteiten waarvoor in het sectormodel historie is gedefinieerd, is het `tijdvakRelatie` verplicht. Indien het antwoordende systeem hun waarde niet kent, dienen de elementen zonder inhoud en met het attribuut `StUF:noValue="waardeOnbekend"` te worden opgenomen. Wanneer het antwoordende systeem zeker weet dat de gegevens voor een entiteit op het moment van verzenden geldig zijn, dan dient het element `einddatumGeldigheid` met het attribuut `StUF:noValue="geenWaarde"` en met een lege inhoud in het bericht te worden opgenomen. Wanneer het zendende systeem zeker weeet dat een relatie niet beëindigd is, dan dient in deze relatie het element `einddatumRelatie` met het attribuut `StUF:noValue="geenWaarde"` en met een lege inhoud te worden opgenomen

Historische gegevens van een fundamenteel entiteittype (bijv. een persoon) worden in het antwoordbericht opgenomen door in de fundamentele entiteit met de meest recente gegevens na de eigen elementen meerdere elementen voor dat fundamentele entiteittype op te nemen met de historische gegevens. De historische occurrences worden aflopend gesorteerd naar het element `begindatumGeldigheid` in het eerste element van zo'n occurrence. Na de elementen met historische gegevens voor het fundamentele entiteittype volgen de relaties. Hier wordt hetzelfde mechanisme gehanteerd. Wanneer een relatie-entiteit zelf historie heeft (bijv. een huwelijk) voor een of meer van zijn elementen, dan bevat de relatie-entiteit naast de eigen elementen (ook het element voor de gerelateerde entiteit is hier een eigen element!) met de meest recente gegevens nul of meer relatie-entiteiten met historische gegevens. Relaties die historisch zijn doordat de `einddatumRelatie` is gevuld worden aflopend gesorteerd naar `einddatumRelatie` opgenomen na de actuele relaties. Voor de gerelateerde entiteiten worden alleen de actuele waarden teruggegeven.

6.5 Regels voor ontvangstbevestigingsberichten

Het ontvangstbevestigingbericht heeft geen body en bestaat uitsluitend uit element `<StUF:stuurgegevens/>`. Het ontvangstbevestigingbericht is gedefinieerd in Bijlage 1: Het Generieke XML Schema voor StUF-berichten

6.6 Regels voor foutberichten

Het foutbericht bevat in de stuurgegevens naast de standaard elementen ook het element `crossRefNummer` dat een belangrijke rol speelt omdat de waarde van dit element verwijst naar het bericht dat de foutmelding heeft veroorzaakt. Het element berichtsoort bevat als waarde 'Fo01. De elementen `entiteittype, sectormodel, versieStUF, versieSectorModel, zender` en `ontvanger` worden gevuld op basis van de waarden in het bericht naar aanleiding waarvan het foutbericht wordt aangemaakt. De elementen `referentienummer` en `tijdstipBericht` worden gevuld conform de regels in paragraaf 4.1.4 en 4.1.5. Het `crossRefNummer` wordt gevuld met het referentienummer van het bericht naar aanleiding waarvan het foutbericht wordt aangemaakt.

Het foutbericht heeft een body met als elementen `code, plek` en `omschrijving`. De `code` geeft een nummer ter identificatie van het foutbericht, de `plek` geeft met 'client' en 'server' waar de oorzaak van de fout gezocht moet worden, op de client respectievelijk de server. De `omschrijving` geeft een een nadere omschrijving van de fout. Onderstaande tabel geeft de in de standaard onderkende waarden voor code, plek en omschrijving.

Code	Plek	Omschrijving
StUF001	client	De stuurgegevens zijn onjuist gevuld
StUF002	server	Het interactieve proces voor het afhandelen van een synchrone vraag is niet actief
StUF003	server	De gevraagde gegevens zijn niet beschikbaar
StUF004	server	De gevraagde sortering wordt niet ondersteund
StUF005	server	Er heeft zich in de StUF-communicatie een time-out voorgedaan
StUF006	server	Het vraagbericht bevat als selectiecriterium zowel de sleutel in het vragende systeem als in het ontvangende systeem
StUF007	server	Het ontvangende systeem ondersteunt niet het bevraagd worden op sleutel in het vragende systeem
StUF008	server	De beantwoording van een vraagbericht vergt meer systeemresources dan het antwoordende systeem beschikbaar heeft
StUF009	client	Het vraagbericht is gericht aan een niet bekend systeem
StUF010	client	Het vragende systeem is niet geautoriseerd voor de gevraagde gegevens
StUF011	client	De syntax van StUF-vraagbericht is onjuist
StUF012	server	Het ontvangende systeem ondersteunt niet de afhandeling van asynchrone vraagberichten
StUF013[7]	client	Het vragende systeem is bij het ontvangende systeem niet bekend

Tabel 8: StUF-foutberichten

In een sectormodel kunnen extra codes, plekken en omschrijvingen worden gedefinieerd. De codes dienen te starten met de code voor het sectormodel.

[7] Nieuwe StUF 2.04-foutberichtdefinities kunnen worden toegevoegd vanaf StUF013

7 COMMUNICATIE

In de voorafgaande hoofdstukken is beschreven hoe berichten worden gemaakt en wat verwacht mag worden van een StUF-compliant berichtverwerkend systeem. Hierbij is op een hoog functioneel niveau gesproken over berichtcycli.

Dit hoofdstuk beschrijft een mogelijk aanpak hoe het transport van StUF-berichten daadwerkelijk kan worden geïmplementeerd. Er is hier gekozen voor WSDL 1.1 (Web Services Description Language) vanwege haar laagdrempeligheid en wijde verspreidheid. WSDL is gebaseerd op HTTP en SOAP 1.1 technologie en bouwt daarop verder.

StUF heeft als ambitie de inhoud ("payload") van berichten te standaardiseren en laat de definitie van de envelop (het communicatieprotocol) zo vrij mogelijk. Daarom kan StUF ook aan andere communicatieprotocollen (dan SOAP en WSDL) gebonden worden zoals bijvoorbeeld ebMS (zie [EBMS] in de bijlage) uit de ebXML-familie.

Voor de wijze van berichtuitwisseling is gekozen voor de Conversational Message Exchange en niet voor het concept van Remote Procedure Calls (RPC) met in- en uitvoerparameters.

7.1 Uitwisseling via een bestand

Om de volgorde van de StUF -berichten bij communicatie via een bestand te waarborgen worden ze in de volgorde van de gewenste verwerking weggeschreven in een XML-bestand met als root-tag `<StUF-berichtenSet>`). Binnen de `<StUF-berichtenSet>`-tag worden de in een sectormodel gedefinieerde berichten opgenomen die asynchroon verzonden kunnen worden en het StUF:foutBericht. Het bevestigingsbericht hoeft niet te worden opgenomen, omdat de ontvangst van via een bestand geleverde berichten niet bevestigd hoeft te worden.

Onderstaand de definitie van het element `<StUF-berichtenSet>` afkomstig uit het XML schema `stuf204.xsd`. Dit element wordt opgebouwd met subelementen van het type `anyType` om alle berichten te kunnen meenemen die voor kunnen komen in een berichtenbestand. De desbetreffende berichtenschema's definieren deze in meer detail. Een correcte validatie is mogelijk door in het bericht expliciet de namespace van het sector-specieke berichtenschema op te nemen waarin het bericht gevalideerd dient te worden.

```
<element name="StUF-berichtenSet">
      <complexType>
         <sequence minOccurs="0" maxOccurs="unbounded">
            <choice>
               <element name="kennisgevingsBericht" type="anyType"/>
               <element name="vraagBericht" type="anyType"/>
               <element name="asynchroonAntwoordBericht" type="anyType"/>
               <element ref="StUF:foutBericht"/>
            </choice>
         </sequence>
      </complexType>
</element>
```

De defnitie van het element StUF-`berichtenSet` in het StUF-schema heeft tot gevolg dat de mogelijk voorkomende berichttypen hier met het type "anyType" gedefinieerd worden. Op het niveau van de StUF-standaard kan de inhoud van de elementen `kennisgevingsBericht`, `vraagBericht` en `asynchroonAntwoordBericht` niet worden gevalideerd. De inhoud ervan wordt per slot van rekening in de sectormodellen gedefinieerd. De beoogde inhoud kan door de zender expliciet worden gemaakt door te verwijzen naar de sectormodel namespace in de verschillende elementen binnen het element `StUF-berichtenSet`.

Op deze manier kunnen grote hoeveelheden StUF -berichten in een groot bestand worden opgeslagen en vervolgens op een medium worden geplaatst dat kan worden uitgewisseld tussen de zender en ontvanger van de StUF -berichten. Technieken als TAR, ZIP en RAR bieden functionaliteit om een bestand te splitsen mocht dit noodzakelijk zijn vanwege de opslagcapaciteit van het gebruikte mediatype. De ontvanger van het medium zal de berichten uitpakken en deze aan de geadresseerde applicatie aanbieden. Deze manier van uitwisselen zal vooral gebruikt worden wanneer een datanetwerk niet voorhanden is, het economisch onverantwoord is of de performance van dit netwerk het niet aantrekkelijke maakt om grote hoeveelheden data te verwerken. We zullen in dit hoofdstuk verder geen aandacht besteden aan deze vorm van uitwisseling.

7.2 Berichtenuitwisseling op basis van HTTP, SOAP en WSDL

De uitwisseling van StUF-berichten via de HTTP/SOAP binding wordt gedefinieerd met behulp van WSDL (versie

1.1). Zie voor meer details over WSDL 1.1: http://www.w3.org/TR/wsdl.

Onderstaand staat een WSDL schema voor het sectormodel Basisgegevens. Dit schema definieert de webservices voor het sectormodel basisgegevens. Dit schema gaat uit van twee web services:

1. voor het ontvangen van asynchrone berichten (kennisgevingen, asynchrone antwoorden, asynchrone vragen en foutberichten),
2. voor het ontvangen van synchrone vragen.

Dit wordt gedaan door per sectorberichttype in het sectormodel Basisgegevens (kennisgeving, vraag, synchroon antwoord en asynchroon antwoord) en per generiek StUF-berichttype (bevestiging en fout) een message te definieren. Voor sector-specifieke berichten wordt gebruikt gemaakt van het berichtenschema uit het desbetreffende domein.

```
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:tns="http://www.egem.nl/StUF/wsdl/bg0204.wsdl"
xmlns:StUF="http://www.egem.nl/StUF/StUF0204"
xmlns:BG="http://www.egem.nl/StUF/sector/bg/0204"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.egem.nl/StUF/wsdl/bg0204.wsdl" name="StUFBG">
   <import namespace="http://www.egem.nl/StUF/stuf0204" location="stuf0204.xsd"/>
   <import namespace="http://www.egem.nl/StUF/sector/bg/0204" location="bg0204.xsd"/>
   <message name="StUFBG_Kennisgeving">
      <part name="kennisgeving" element="BG:kennisgevingsBericht"/>
   </message>
   <message name="StUFBG_Vraag">
      <part name="vraag" element="BG:vraagBericht"/>
   </message>
   <message name="StUFBG_AsynchroonAntwoord">
      <part name="asynchroonAntwoord" element="BG:asynchroonAntwoordBericht"/>
   </message>
   <message name="StUFBG_SynchroonAntwoord">
      <part name="synchroonAntwoord" element="BG:synchroonAntwoordBericht"/>
   </message>
   <message name="StUF_Bevestiging">
      <part name="bevestiging" element="StUF:bevestigingsBericht"/>
   </message>
   <message name="StUF_Fout">
      <part name="fout" element="StUF:foutBericht"/>
   </message>
   <portType name="StUFBG_AsynchroonPortType">
      <operation name="ontvangKennisgeving">
         <input message="tns:StUFBG_Kennisgeving"/>
         <output message="tns:StUF_Bevestiging"/>
      </operation>
      <operation name="ontvangAsynchroneVraag">
         <input message="tns:StUFBG_Vraag"/>
         <output message="tns:StUF_Bevestiging"/>
      </operation>
      <operation name="ontvangAsynchroonAntwoord">
         <input message="tns:StUFBG_AsynchroonAntwoord"/>
         <output message="tns:StUF_Bevestiging"/>
      </operation>
      <operation name="ontvangFout">
         <input message="tns:StUF_Fout"/>
         <output message="tns:StUF_Bevestiging"/>
      </operation>
   </portType>
   <portType name="StUFBG_SynchroonPortType">
      <operation name="beantwoordSynchroneVraag">
         <input message="tns:StUFBG_Vraag"/>
         <output message="tns:StUFBG_SynchroonAntwoord"/>
         <fault name="StUF_Fout" message="tns:StUF_Fout"/>
      </operation>
   </portType>
   <binding name="StUFSOAPBindingAsynchroon" type="tns:StUFBG_AsynchroonPortType">
      <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="ontvangKennisgeving">
         <soap:operation soapAction="http://www.egem.nl/StUF"/>
         <input>
```

```
            <soap:body use="literal"/>
         </input>
         <output>
            <soap:body use="literal"/>
         </output>
      </operation>
      <operation name="ontvangAsynchroneVraag">
         <soap:operation soapAction="http://www.egem.nl/StUF"/>
         <input>
            <soap:body use="literal"/>
         </input>
         <output>
            <soap:body use="literal"/>
         </output>
      </operation>
      <operation name="ontvangAsynchroonAntwoord">
         <soap:operation soapAction="http://www.egem.nl/StUF"/>
         <input>
            <soap:body use="literal"/>
         </input>
         <output>
            <soap:body use="literal"/>
         </output>
      </operation>
      <operation name="ontvangFout">
         <soap:operation soapAction="http://www.egem.nl/StUF"/>
         <input>
            <soap:body use="literal"/>
         </input>
         <output>
            <soap:body use="literal"/>
         </output>
      </operation>
   </binding>
   <binding name="StUFSOAPBindingSynchroon" type="tns:StUFBG_SynchroonPortType">
      <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="beantwoordSynchroneVraag">
         <soap:operation soapAction="http://www.egem.nl/StUF"/>
         <input>
            <soap:body use="literal"/>
         </input>
         <output>
            <soap:body use="literal"/>
         </output>
         <fault name="StUF_Fout">
            <soap:fault name="StUF_Fout" use="literal"/>
         </fault>
      </operation>
   </binding>
   <service name="StUFBGAsynchroon">
      <port name="StUFBGAsynchronePort" binding="tns:StUFSOAPBindingAsynchroon">
         <soap:address location="http://example.com/StUFBGAsynchroon"/>
      </port>
   </service>
   <service name="StUFBGSynchroon">
      <port name="StUFBGSynchronePort" binding="tns:StUFSOAPBindingSynchroon">
         <soap:address location="http://example.com/StUFBGSynchroon"/>
      </port>
   </service>
</definitions>
```

Met behulp van `portTypes` worden op abstract niveau de verschillende varianten voor het uitwisselen van StUF-berichten gegroepeerd naar de synchrone en de asynchrone uitwisseling. Asynchroon zijn er de varianten:

- `ontvangKennisgeving`
- `ontvangAsynchroneVraag`
- `ontvangAsynchroonAntwoord`
- `ontvangFout`

Synchroon is er alleen `beantwoordVraag` met als response hetzij het antwoord, hetzij een foutbericht.

We definieren als `soapAction http://www.egem.nl/StUF` om aan te geven dat de service uitsluitend door StUF gedefineerde berichten mag verwachten. Omdat de uit te wisselen berichten volledig gedefinieerd zijn in het sectormodel cq de StUF-standaard, kunnen we ze zonder verdere encoding (“literal”) opnemen in de body.

Er is gekozen voor twee verschillende services, omdat een asynchrone service feitelijk als enige taak heeft het ontvangen bericht te persisteren en dit persisteren te bevestigen naar de zender van het bericht. Bij de synchrone service dient onmiddellijk een antwoord geformuleerd te worden op de gestelde vraag.
Het beantwoorden van een vraag kan veel meer resources vergen en dient op een andere wijze getuned te kunnen worden, dan het opslaan van een inkomend bericht. In het endpoint dient de url nog correct te worden ingevuld.

Voor elk sectormodel dient op basis van dit voorbeeld een specifiek WSDL-schema gemaakt en gepubliceerd te worden.

Binnen de connection timeout voor de HTTP-server moet er in principe antwoord komen van de ontvanger, zo niet dan wordt een HTTP-fout gegenereerd. Zolang een zender geen bevestigingsbericht heeft ontvangen, is het zijn verantwoordelijkheid ervoor te zorgen dat het bericht bij de ontvanger komt. Een bevestigingsbericht mag door de ontvanger pas gestuurd worden als hij heeft gewaarborgd dat het bericht niet meer verloren kan gaan. StUF definieert geen additionele foutafhandeling boven op de standaard foutafhandeling rond time-outs voor HTTP. Er hoeft bijvoorbeeld geen speciaal foutbericht gestuurd te worden, indien een synchrone vraag niet binnen de time-out tijd voor HTTP-communicatie beantwoord kan worden.

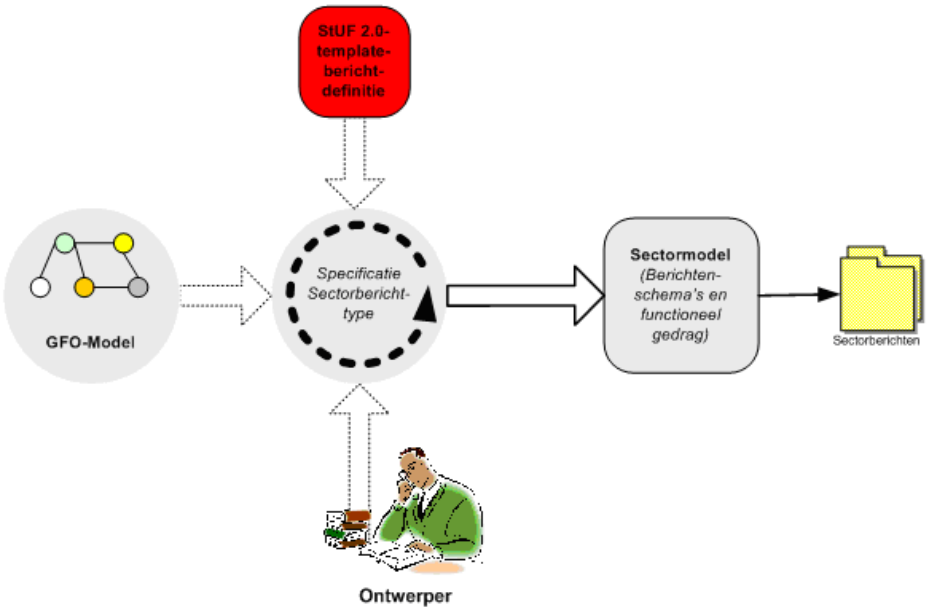
StUF kent synchroon en asynchroon berichtenverkeer. Bij synchroon berichtenverkeer bevat de SOAP-Response het antwoord- of foutbericht, mits dit binnen de connection time out voor http beschikbaar is en bij asynchroon berichtenverkeer het bevestigingsbericht.

Het antwoord op een asynchroon StUF-vraagbericht wordt in een gescheiden SOAP- cyclus teruggestuurd, waarbij zender en ontvanger van rol wisselen. De zender van het StUF-vraagbericht ontvangt een SOAP-request met daarin een StUF-antwoordbericht. De zender van dit request ontvangt een SOAP-Response met daarin een bevestigingsbericht om aan te geven dat het bericht goed is aangekomen. Aan de hand van het element crossreferentienummer in de stuurgegevens kan de StUF-zender bepalen bij welke vraag dit antwoord hoort. Op deze manier wordt de asynchrone StUF-communicatie via SOAP opgelost.

8 HET MAKEN VAN EEN SECTORMODEL

8.1 Het modelleren van het domein in het sectormodel

Een sectormodel bevat altijd een entiteit-relatie-diagram of ERD dat het domein beschrijft waarover gegevens uitgewisseld kunnen worden. In dit ERD worden in ieder geval de fundamentele entiteittypen en de tabelentiteittypen opgenomen.



Figuur 10 Assembleren specificaties voor sector-berichttypen

De tabelentiteittypen worden in het ERD opgenomen om te expliciteren dat bij de gegevensuitwisseling tabellen gebruikt worden die een bepaalde partij definieert en onderhoudt. Het definiëren van een waardebereik voor een bepaald element kan een alternatief zijn voor het definiëren van een tabelentiteittype. Er kan gekozen worden voor het definiëren van een waardebereik als niet te verwachten is dat het waardebereik op gezette tijden wijzigt en als het aantal waarden niet te groot is (zeker geen honderden). Het grote voordeel van tabelentiteittypen is dat StUF de functionaliteit bevat waarmee wijzigingen in een tabelentiteit gedistribueerd kunnen worden naar alle geïnteresseerde systemen en dat een wijziging in de tabel niet leidt tot een wijziging van de software. Bij een wijziging in een waardebereik is meestal een aanpassing in de software noodzakelijk.

Bij de eigenschappen van een attribuut wordt in het sectormodel gespecificeerd of een object die eigenschap altijd heeft (optionaliteit=`false`) of dat een object die eigenschap al dan niet kan bezitten (optionaliteit=`true`). De geboortedatum is een eigenschap die een persoon altijd heeft, want een persoon moet geboren zijn om te kunnen bestaan. Hetzelfde geldt voor eigenschappen als zijn burgerlijke staat en geslachtsnaam. Een eigenschap als voorvoegsels geslachtsnaam hoeft een persoon niet altijd te hebben. Binnen een adres is de huisnummertoevoeging een voorbeeld van een eigenschap die al dan niet kan voorkomen.

Een associatie is een relatie tussen entiteittypen. Indien deze relatie zelf eigenschappen kent wordt deze relatie als een apart relatie-entiteittype in het ERD opgenomen. De relatie tussen een persoon en een adres heeft bijvoorbeeld zelf geen eigenschappen en wordt daarom gemodelleerd als een directe associatie tussen de entiteittypen. Tijdens de uitwisseling worden alle associaties als relatie-entiteittype in het bericht opgenomen, dus ook de associaties die geen eigenschappen bevatten.

De huwelijksrelatie tussen twee personen heeft wel eigenschappen en wordt daarom gemodelleerd als een relatie-entiteittype. Bij de relatie tussen een ouder en een kind is beide mogelijk. Als relevant is hoe de relatie tussen ouder en kind tot stand gekomen is (geboorte, adoptie, erkenning, e.d.), dan wordt een relatie-entiteittype gebruikt met als eigenschap de familierechtelijke betrekking. Wanneer dit niet relevant is, dan kan volstaan worden met een verbindingslijn van persoon naar persoon.

In het ERD van een sectormodel wordt een momentopname van het beschouwde domein gemodelleerd en wordt tijdsafhankelijkheid genegeerd. Van relaties die zelf geen eigenschappen hebben, worden dus de ingangsdatum en de einddatum van de relatie niet als attribuut opgenomen. Het verblijfsadres van een persoon is een voorbeeld van een

dergelijke relatie. In de praktijk is het wel degelijk relevant te weten op welke adressen een persoon in de loop van de tijd heeft verbleven. In het GFO wordt daarom aangegeven of de historie van relaties relevant is. Hetzelfde geldt voor attributen. Ook bij een attribuut wordt aangegeven of de historie van dat attribuut relevant is. Het grote voordeel van dit uitgangspunt is, dat in het sectormodel expliciet kan worden aangegeven dat op een bepaald moment in de tijd een persoon maar één verblijfsadres kan hebben of dat een attribuut maar één waarde kan hebben.

Het ERD heeft in principe de derde normaalvorm, maar er zijn enkele uitzonderingen:

- Meervoudig voorkomende attributen als een telefoonnummer of een gironummer bij een persoon worden niet uitgenormaliseerd naar een zelfstandig entiteittype. Dergelijke attributen worden in het sectormodel opgenomen bij het fundamentele entiteittype. Verder wordt er aangegeven dat ze meervoudig kunnen voorkomen door te specificeren dat de kardinaliteit groter is dan één. Er is voor deze werkwijze gekozen, omdat het sectormodel de werkelijkheid modelleert en niet de gegevensstructuren in een database. In de werkelijkheid is een telefoon- of gironummer meestal geen zelfstandig voorkomend object (een model voor KPN Telecom c.q. de Postbank is de bekende uitzondering die de regel bevestigt).
- Een gegevensgroep die afhankelijk is van een occurrence van een fundamenteel entiteittype, maar die in zijn geheel al dan niet voorkomt bij een bepaalde occurrence, mag gemodelleerd worden als een onafhankelijk apart entiteittype. Het voordeel hiervan is dat de berichtverwerking opgedeeld kan worden in welomschreven functionele eenheden, waardoor hergebruik van bepaalde functionele eenheden eenvoudiger te realiseren is en de complexiteit van de berichtverwerking verminderd wordt. Bij het definiëren van berichten kunnen dit soort afhankelijke gegevensgroepen worden opgenomen binnen de fundamentele entiteit waarvan ze afhankelijk zijn.

Attributen die binnen een entiteittype zijn gedefinieerd kunnen in meerdere sectormodellen voorkomen en verwijzen naar dezelfde eigenschappen van objecten in de werkelijkheid. Deze attributen moeten over de sectormodellen heen dezelfde elementnaam, dezelfde kardinaliteit en hetzelfde waardebereik hebben om eenduidige uitwisseling van gegevens mogelijk te maken. Het sectormodel Basisgegevens, dat alle gegevens dient te bevatten die in twee of meer andere sectormodellen voorkomen, krijgt daarmee een speciale status. Wanneer een ander sectormodel over een object gegevens wil uitwisselen die reeds voorkomen in het sectormodel Basisgegevens, dan dient de definitie uit het sectormodel Basisgegevens te worden overgenomen. In elementen die voorkomen in vraagberichten dient het attribuut `StUF:exact` te worden opgenomen.

8.2 Het definiëren van de berichtenschema's met behulp van XML-schema

Deze paragraaf bevat een beschrijving van het maken van berichtenschema's op basis van de eerder genoemde specificaties. De methode die hier wordt beschreven is als voorbeeld bedoeld en is daarom niet voorschrijvend. Er is gestreefd naar zoveel mogelijk hergebruik van definities, zodat de berichtenschema's goed onderhoudbaar zijn.

De regels gedefinieerd in paragraaf 6.1.2 hebben gevolgen voor de inrichting van de constructies `complexType` en `element`: deze constructies dienen altijd uitgerust te worden met de attributen `nillable="true"` en `StUF:noValue`. In elementen die voorkomen in vraagberichten dient het attribuut `StUF:exact` te worden opgenomen. Alleen datumelementen zijn een uitzondering op deze regel, daar wordt volstaan met het attribuut `StUF:indOnvolledigeDatum`.

Kennisgevingberichten en antwoordberichten kunnen dezelfde of een verschillende structuur hebben. Dit is een keuze voor de ontwerper van het sectormodel. Het eenvoudigste is om ze dezelfde structuur te geven. In dat geval kan volstaan worden met één `complexType` per entiteittype. In het onderstaande gaan we uit van dezelfde structuur voor kennisgeving- en antwoordberichten.

De basis van de berichtenschema's vormen de attributen uit het ERD in het GFO. Per attribuut wordt er een `simpleType` in het XML Schema gedefinieerd. Hieronder zijn een aantal voorbeelden opgenomen.

```
<simpleType name="AanduidingBijzonderNederlanderschap">
     <restriction base="string">
        <maxLength value="1"/>
     </restriction>
</simpleType>

<simpleType name="AanduidingGegevensInOnderzoek">
     <restriction base="string">
        <maxLength value="1"/>
     </restriction>
</simpleType>

<simpleType name="AanduidingBijHuisnummer">
```

```
      <restriction base="string">
         <maxLength value="2"/>
      </restriction>
</simpleType>
```

Voor elk fundamenteel en tabelentiteittype uit het ERD wordt een `complexType` gedefinieerd, dat de eigen attributen bevat, maar geen relaties naar andere entiteittypen. Een element mag meerdere keren voorkomen binnen een entiteittype. Dit is bijvoorbeeld nuttig voor een element als een bank/gironummer, waar een persoon er verschillende van kan hebben. Omdat een element meerdere malen mag voorkomen, hoeft geen relatie-entiteittype geïntroduceerd te worden om meerdere voorkomens van een gegeven in een bericht op te nemen. Het sectormodel bepaalt of er meerdere voorkomens van een gegeven zijn toegestaan. De structuur van de `complexTypes` ten behoeve van de berichten hoeft niet één-op-één identiek te zijn aan het ERD uit het GFO.

Ook voor relatie-entiteittypen wordt een `complexType` gedefinieerd. Een complicatie hierbij is dat relaties een richting hebben. De relatie vanuit een persoon naar een adres geeft het adres van een persoon en is een andere relatie dan de relatie vanuit adres naar persoon, die aangeeft welke personen er op een adres wonen. Voor een associatie of een relatie-entiteittype in het ERD worden dus in het schema in het sectormodel één of twee complex types voor relatie-entiteiten gedefinieerd.

In plaats van de relatie naar een tabelentiteittype kan in een aantal gevallen de logische sleutel van het tabelentiteittype als element bij het fundamentele entiteittype worden opgenomen. Het is bijvoorbeeld niet nodig om de relatie van een adres naar het land in het bericht op te nemen als een relatie-entiteittype naar het tabelentiteittype *Land*. Eenvoudiger is het om de landcode te definiëren als een element van het entiteittype *Adres*. De logische sleutel van een tabelentiteittype kan worden opgenomen in een ander entiteittype als aan twee voorwaarden wordt voldaan:

1. De relatie heeft zelf geen eigenschappen
2. De begin- en einddatum van de relatie zijn niet relevant.

De relatie tussen adres en land voldoet aan beide criteria. Bij de relatie tussen een persoon en zijn nationaliteit zijn de begin- en einddatum wel relevant. De code nationaliteit kan daardoor niet als element bij persoon worden opgenomen.

Het complexType voor een fundamentele, een relatie en een tabel entiteittype dient altijd ook het element `extraElementen` te bevatten met het type `ExtraElementen` uit het basis schema (zie Bijlage 1: Het Generieke XML Schema voor StUF-berichten). Het `complexType` voor een fundamenteel entiteittype dient de attribute group `fundamenteel` uit het bovengenoemd basis schema te bevatten, voor een relatie entiteittype de attribute group `relatie` en voor een tabel entiteittype de attribute group `tabel`. Als historie relevant is voor een fundamenteel of een relatie entiteittype, dient ook het element `tijdvakGeldigheid` te worden opgenomen. In een relatie entiteittype kan desgewenst het element `tijdvakRelatie` worden opgenomen. Het `complexType` voor een relatie-entiteit bevat naast de elementen van het relatie-entiteittype zelf ook een element met als type de fundamentele of de tabelentiteit waarnaar de relatie ligt. Het `complexType` voor een relatie-entiteit bevat dus ook altijd het fundamentele entiteittype waarnaar de relatie ligt.

Hieronder zijn als voorbeeld (een deel van) de `complexTypes` voor een tabel entiteittype, een fundamenteel entiteittype en een relatie entiteittype opgenomen. In het voorbeeld van de relatie wordt als gerelateerde persoon verwezen naar een apart complex type `PRS-gerelateerde`, dat minder elementen en relaties bevat dan het standaard complex type voor een persoon. Op deze manier wordt ervoor gezorgd dat deze berichtenschema's alleen op twee niveaus personen kunnen bevatten en dat er in dit geval geen diepere nesting ten gevolge van recursie kan ontstaan.

8.2.1 Tabel entiteittype

```
<complexType name="LND-tabel">
      <annotation>
         <documentation>tabel entiteit land</documentation>
      </annotation>
      <sequence>
         <element name="landcode" nillable="true" minOccurs="0">
            <complexType>
               <simpleContent>
                  <extension base="BG:Landcode">
                     <attributeGroup ref="StUF:element"/>
                     <attribute name="kerngegeven" type="boolean" fixed="true"/>
                  </extension>
               </simpleContent>
            </complexType>
         </element>
         <element name="landnaam" nillable="true" minOccurs="0">
```

```
            <complexType>
               <simpleContent>
                  <extension base="BG:Landnaam">
                     <attributeGroup ref="StUF:element"/>
                  </extension>
               </simpleContent>
            </complexType>
         </element>
         <element name="ingangsdatum" type="StUF:DatumMetIndicator" nillable="true"
                  minOccurs="0"/>
         <element name="einddatum" type="StUF:DatumMetIndicator" nillable="true"
                  minOccurs="0"/>
         <element name="extraElementen" type="StUF:ExtraElementen" minOccurs="0"/>
      </sequence>
      <attributeGroup ref="StUF:tabel"/>
</complexType>
```

8.2.2 Fundamenteel entiteittype

```
<complexType name="PRS-fund">
   <annotation>
      <documentation>fundamentele entiteit PRS</documentation>
   </annotation>
   <sequence>
         <element name="a-nummer" nillable="true" minOccurs="0">
            <complexType>
               <simpleContent>
                  <extension base="BG:A-nummer">
                     <attribute name="kerngegeven" type="boolean" fixed="true"/>
                     <attributeGroup ref="StUF:element"/>
                  </extension>
               </simpleContent>
            </complexType>
         </element>
         ..............
         <element name="academischeTitel" nillable="true" minOccurs="0"
                  maxOccurs="unbounded">
            <complexType>
               <simpleContent>
                  <extension base="BG:AcademischeTitelCode">
                     <attributeGroup ref="StUF:element"/>
                  </extension>
               </simpleContent>
            </complexType>
         </element>
         ..............
         <element name="tijdvakGeldigheid" type="StUF:TijdvakGeldigheid"
                                                                  minOccurs="0"/>
         <element name="extraElementen" type="StUF:extraElementen" minOccurs="0"/>
   </sequence>
   <attributeGroup ref="StUF:fundamenteel"/>
</complexType>
```

8.2.3 Relatie entiteittype

```
<complexType name="PRSPRSKND-rel">
      <annotation>
         <documentation>relatie entiteit kind</documentation>
      </annotation>
      <sequence>
            <element name="ingangsdatum" type="StUF:DatumMetIndicator" nillable="true"
                     minOccurs="0"/>
            <element name="einddatum" nillable="true" minOccurs="0">
               <complexType>
                  <simpleContent>
                     <extension base="StUF:Datum">
                        <attributeGroup ref="StUF:element"/>
                     </extension>
                  </simpleContent>
               </complexType>
            </element>
            <element name="tijdvakRelatie" type="StUF:TijdvakRelatie" minOccurs="0"/>
```

```
            <element name="tijdvakGeldigheid" type="StUF:TijdvakGeldigheid"
                                                                    minOccurs="0"/>
            <element name="extraElementen" type="StUF:ExtraElementen" minOccurs="0"/>
            <element name="PRS" type="BG:PRS-gerelateerde"/>
      </sequence>
      <attributeGroup ref="StUF:relatie"/>
</complexType>
```

Als basis voor het maken van het vraag-, antwoord- en kennisgevingbericht worden de `complexTypes` voor de fundamentele, relatie en tabel entiteittypen gebruikt. Om het aantal berichten op WSDL-niveau te minimaliseren onderkennen we slechts drie berichttypen:

- kennisgeving
- vraag
- antwoord

De verschillende entiteiten worden binnen een choice opgenomen. Als een entiteit relaties heeft, dan worden deze in het bericht opgenomen door middel van een extensie van de hierboven behandelde complexTypes. Hieronder staan delen van de definities van deze drie berichten.

8.2.3.1 Kennisgevingsbericht

```
<element name="kennisgevingsBericht">
      <annotation>
         <documentation>body voor kennisgevingsberichten BG</documentation>
      </annotation>
      <complexType>
         <sequence>
            <element ref="StUF:stuurgegevens"/>
            <element name="body">
               <complexType>
                  <choice>
                     <element name="LND" type="BG:LND-tabel" maxOccurs="2"/>
                     <element name="PRS" maxOccurs="2">
                        <complexType>
                           <complexContent>
                              <extension base="BG:PRS-fund">
                                 <sequence>
                                    <element  name="PRSADRVBL" type="BG:XXXADRVBL-rel"
                                              nillable="true"/>
                                    <element  name="PRSPRSHUW" type="BG:PRSPRSHUW-rel"
                                              nillable="true" minOccurs="0"
                                              maxOccurs="unbounded"/>
                                 </sequence>
                              </extension>
                           </complexContent>
                        </complexType>
                     </element>
                  </choice>
               </complexType>
            </element>
         </sequence>
      </complexType>
</element>
```

8.2.3.2 Vraagbericht

```
<element name="vraagBericht">
      <annotation>
         <documentation>body voor vraag-berichten BG</documentation>
      </annotation>
      <complexType>
         <sequence>
            <element ref="StUF:stuurgegevens"/>
            <element name="body">
               <complexType>
                  <choice>
                     <element   name="LND" type="BG:LND-tabel" nillable="true"
                                minOccurs="3" maxOccurs="4"/>
                     <element name="PRS" nillable="true" minOccurs="3" maxOccurs="4">
```

```
                        <complexType>
                           <complexContent>
                              <extension base="BG:PRS-fund">
                                 <sequence minOccurs="0">
                                    <element   name="PRSADRVBL" type="BG:XXXADRVBL-rel"
                                               nillable="true"/>
                                    <element   name="PRSPRSHUW" type="BG:PRSPRSHUW-rel"
                                               nillable="true" minOccurs="0"
                                               maxOccurs="unbounded"/>
                                 </sequence>
                              </extension>
                           </complexContent>
                        </complexType>
                     </element>
                  </choice>
               </complexType>
            </element>
         </sequence>
      </complexType>
</element>
```

8.2.3.3 Antwoordbericht

```
<element name="antwoordBericht">
      <annotation>
         <documentation>body voor antwoordberichten BG</documentation>
      </annotation>
      <complexType>
         <sequence>
            <element ref="StUF:stuurgegevens"/>
            <element name="body">
               <complexType>
                  <choice>
                     <element   name="LND" type="BG:LND-tabel" nillable="true"
                                minOccurs="0" maxOccurs="unbounded"/>
                     <element   name="PRS" nillable="true" minOccurs="0"
                                maxOccurs="unbounded">
                        <complexType>
                           <complexContent>
                              <extension base="BG:PRS-fund">
                                 <sequence>
                                    <element   name="PRS" type="BG:PRS-fund" minOccurs="0"
                                               maxOccurs="unbounded">
                                          <annotation>
                                               <documentation>
                                                    Historische   gegevens
                                               </documentation>
                                          </annotation>
                                    </element>
                                    <element   name="PRSADRVBL" type="BG:XXXADRVBL-rel
                                               nillable="true" maxOccurs="unbounded"/>
                                    <element   name="PRSPRSHUW" type="BG:PRSPRSHUW-rel"
                                               nillable="true" minOccurs="0"
                                               maxOccurs="unbounded"/>
                                 </sequence>
                              </extension>
                           </complexContent>
                        </complexType>
                     </element>
                  </choice>
               </complexType>
            </element>
         </sequence>
      </complexType>
</element>
```

In het antwoordbericht zien we de voorzieningen voor het opnemen van historische gegevens. Het complexe type `PRS-Fund` wordt uitgebreid (door middel van een `extension`) met een onbeperkt aantal historische personen voor de relaties en in de relatie voor het verblijfsadres is maxOccurs nu unbounded om ook de historische adressen in het bericht te kunnen opnemen.

OPEN ISSUES:

Issues voor nieuwe versies van de StUF standaard:

- **Wensen voor extra functionaliteit:**
  - Foutherstelbericht
  - Onderscheid tussen ontdubbeling en sleutelwijziging
  - Event-informatie, het communiceren van gebeurtenissen uit de werkelijkheid
  - Communiceren van correcties historische gegevens
  - XQuery-syntax en -semantiek voor vraagberichten

- **Aandachtspunten voor bestaande functionaliteit:**
  - Er is momenteel onvoldoende functionaliteit om 100% correctheid te garanderen bij correcties in historische gegevens of bij het niet per gebeurtenis doorgeven van wijzigingen.
  - Er is momenteel geen functionaliteit voor het corrigeren van begin- en einddatum geldigheid.
  - Het tevens opnemen van de toekomstige situatie(s) in de “historische” gegevens. Vervangen van het begrip “historisch” voor het begrip “periodegebonden”.
  - Invoering van expliciete XML-syntax voor gegevensgroepen en kerngegevens.
  - Het verbeteren van de uitbreidbaarheid van de StUF XML Schema’s.
  - Formele conformiteit aan de WS-Interoperability Basic Profile.
  - Namen zoveel mogelijk uitschrijven (b.v. in plaats van T, V, W en C, respectievelijk Toevoeging, Verwijdering, Wijziging en Correctie) om de berichten beter leesbaar te maken.

REFERENTIES

[HTTP] RFC 2616 - Hypertext Transfer Protocol HTTP/1.1
http://www.w3.org/Protocols/rfc2616/rfc2616.html

[SOAP] Versie 1.1
http://www.w3.org/TR/2000/NOTE-SOAP-20000508

[URL/URI] Naming and Addressing: URIs, URLs, ...
http://www.w3.org/Addressing/

[WSDL] Versie 1.1
http://www.w3.org/TR/wsdl

[XML] Extensible Markup Language (XML) 1.0 (Second Edition)
http://www.w3.org/TR/2000/REC-xml-20001006

[XML Schema]
http://www.w3.org/TR/2004/REC-xmlschema-0-20041028 (Primer)
http://www.w3.org/TR/2004/REC-xmlschema-1-20041028 (Structures)
http://www.w3.org/TR/2004/REC-xmlschema-2-20041028 (Datatypes)

[EBMS] ISO 15000-2 ebXML Message Service Specification.
http://www.oasis-open.org/specs/index.php#ebxmlmsgv2

BEGRIPPENLIJST

Activiteit	Een begrensde op zichzelf bestaanbare werkzaamheid of verrichting
Applicatie	Een computerprogramma, of verzameling programma's, die een bepaalde taak verrichten.
Architectuur	Een systematisch gestructureerd overzicht van de onderdelen van het beschouwde probleem domein (i.c. een informatiesysteem of een onderdeel daarvan) en hun onderlinge relatie en in relatie met de omgeving.
Attribuut	(Essentieel) kenmerk van een object
Backoffice	Dat deel van de werkorganisatie waarin geen directe communicatie over en weer met klantenplaatsvindt. Het is het deel waarin de organisatiespecifieke bedrijfsfuncties worden uitgevoerd.
Bericht	Vorm/methode waarin uit te wisselen informatie wordt verpakt
Definitie	Omschrijving, betekenis van een begrip
Element	XML-element te gebruiken om een informatie-element mee te onderscheiden in een XML-document.
Entiteit	In de werkelijk bestaand ding of eenheid
Fundamentele entiteit	Een fundamentele entiteit is een entiteit die autonoom kan bestaan
Relatie-entiteit	Een relatie-entiteit beschrijft de relatie tussen fundamentele entiteiten
Tabelentiteit	Een tabelentiteit bevat tabel met waarden voor een bepaald type attribuut. Tabelentiteiten kunnen als entiteit worden uitgewisseld tussen systemen.
Entiteittype	Type van het werkelijke bestaand ding of eenheid
Firewall	Een set applicaties, die het eigen netwerk tegen aanvallen vanuit andere netwerken beschermt
Frontoffice	Het deel van de werkorganisatie waarin directe communicatie over en weer met de klanten plaatsvindt. De functies van de frontoffice zijn primair gericht op het verzamelen van de juiste en correcte informatie die nodig is voor de functies in de backoffice en op de terugmelding aan de klanten van de resultaten van de handeling(en) in de backoffice..
Gebeurtenis	Iets wat gebeurt of gebeurd is, en relevant is voor een (informatie-) systeem.
Gemeentelijke Functioneel Ontwerp (GFO)	Een beschrijving van de relevante objecten en hun relaties voor een bepaald gemeentelijk beleidsdomein.
Gemeentelijke module	Een software-implementatie van een specifieke gemeentelijke taak.
HTTP	HTTP (HyperText Transfer Protocol) is een standaard protocol voor verzending van webpagina's op het Internet
Informatiesysteem	Systeem met functionaliteit dat gericht is op het beheren van informatie en bijbehorende diensten kan leveren aan gebruikers.
Master-Slave-configuratie	Dit begrip wordt gebruikt om aan te duiden dat in de gegevensuitwisseling sprake is van een situatie waarbij de master bepaalt wat de slave aan informatie krijgt.
Midoffice	Een verzameling van functionaliteiten, die de processen en de daarbij behorende applicaties en gegevens in de frontoffice en de backoffice met elkaar verbindt.
Object	Een voorwerp (fysiek of imaginair) in de werkelijke wereld
Occurrence	Voorval, instantie. Dhr Janssen is een occurence van het entiteittype Persoon.
Proces	Een serie verrichtingen, handel- of werkwijze.
Protocol	Een verzameling regels en afspraken (semantisch en syntactisch) voor het uitwisselen van informatie tussen entiteiten (bijv. programma's, computers)
Referentiearchitectuur	Een abstracte beschrijving van een relevant probleemdomein in de werkelijkheid
Relatie	Een logische verbinding tussen twee objecten. Een relatie kan mogelijk ook zelf weer een object zijn indien het relatie object een eigen leven kent dat relevant is om te beschouwen. Een relatie representeert iets gemeenschappelijks tussen twee objecten.
Sectormodel	In deze context zijn dit alle ontwerpproducten die nodig zijn om StUF-koppelingen te realiseren tussen systemen. Een sectormodel bestaat uit vier delen: 1) Een datamodel (b.v. het ERD: GFO Basis Gegegevens). 2) De berichtendefinities (b.v. in de vorm van een XML Schema). 3) Semantische en functionele afspraken die volgens de StUF-standaard in het sectormodel moeten worden geregeld. 4) Details over de implementatie door de verschillende leveranciers (bijv. sorteringen bij antwoordberichten)
Service	Het aan bieden van een functie, capaciteit, door een aanbieder aan een servicevrager. Primair hierin is de beschrijving van de dienst en (een referentie naar) de te gebruiken informatie. Een andere definitie: een gedefinieerde (o.a pre- en postcondities) en zelf omvattende functieaanroep-specificatie, welke niet afhankelijk is van de context of state van andere services. Een service is meer gebaseerd op het uitvoeren van een taak dan het

	aanbieden van informatie.
Sleutel	(Attribute sleutel): unieke identificerende waarde voor een entiteit
SOAP	Simple Object Access Protocol; SOAP definieert een standaard protocol dat door iedere applicatie gebruikt kan worden om te communiceren en gegevens uit te wisselen met andere applicaties die SOAP ondersteunen
Specificatie	Formele omschrijving van een begrip
Syntax	Leer van de opbouw en structuur van berichten
Systeem	Het geheel dat volgens een beginsel geordend, gerangschikt of ingedeeld is met betrekking tot een bepaald aspect (bijv. informatie).
TCP/IP	Transmission Control Protocol/Internet Procotol: een communicatietaal voor netwerken
Template	Formaat (sjabloon)
WSDL	Web Services Description Language. Een op XML gebaseerde specificatietaal om webdiensten mee te beschrijven.
XML	Extended mark up Language: een volgens het World Wide Web consortium aanbevolen standaard voor het beschrijven van gestructureerde data op het internet
XML-document	Een in XML-notatie weergegeven informatiebericht
XML-Schema	Een metabeschrijving van een XML-document

BIJLAGE 1: HET GENERIEKE XML SCHEMA VOOR STUF-BERICHTEN

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns:StUF="http://www.egem.nl/StUF/StUF0204" xmlns="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.egem.nl/StUF/StUF0204" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="0204">
   <annotation>
      <documentation xml:lang="nl">
               Schema met de in StUF gebruikte elementen en attributen
         </documentation>
   </annotation>
   <!--  definitie van het element stuurgegevens in de SOAP header -->
   <element name="stuurgegevens">
      <complexType>
         <sequence>
            <element name="berichtsoort">
               <simpleType>
                  <restriction base="string">
                     <minLength value="4"/>
                     <maxLength value="4"/>
                     <pattern value="Fo01"/>
                     <pattern value="Lv01"/>
                     <pattern value="Lv02"/>
                     <pattern value="La01"/>
                     <pattern value="La02"/>
                     <pattern value="Lk01"/>
                     <pattern value="Bv01"/>
                  </restriction>
               </simpleType>
            </element>
            <element name="entiteittype">
               <simpleType>
                  <restriction base="string">
                     <minLength value="3"/>
                     <maxLength value="3"/>
                  </restriction>
               </simpleType>
            </element>
            <element name="sectormodel" type="string"/>
            <element name="versieStUF" type="StUF:Versienr"/>
            <element name="versieSectormodel" type="StUF:Versienr"/>
            <element name="zender" type="StUF:Systeem"/>
            <element name="ontvanger" type="StUF:Systeem"/>
            <element name="referentienummer" type="StUF:RefNummer"/>
            <element name="tijdstipBericht" type="StUF:Tijdstip"/>
            <choice>
               <element name="kennisgeving">
                  <complexType>
                     <sequence>
                        <element name="mutatiesoort" type="StUF:Mutatiesoort"/>
                        <element name="indicatorOvername" type="StUF:IndicatorOvername"
                                 default="I"/>
                        <element name="tijdstipMutatie" type="StUF:Tijdstip" minOccurs="0"/>
                     </sequence>
                  </complexType>
               </element>
               <element name="vraag">
                  <complexType>
                     <sequence>
                        <element name="sortering" type="StUF:Sortering" default="0"
                                 minOccurs="0"/>
                        <element name="maximumAantal" type="StUF:MaximumAantal" default="15"
                                 minOccurs="0"/>
                        <element name="indicatorHistorisch" type="boolean" default="false"
                                 minOccurs="0"/>
                        <element name="indicatorVervolgvraag" type="boolean" default="false"
                                 minOccurs="0"/>
                        <element name="indicatorAfnemerIndicatie" type="boolean" default="false"
                                 minOccurs="0"/>
                     </sequence>
                  </complexType>
               </element>
               <element name="antwoord">
                  <complexType>
                     <sequence>
                        <element name="indicatorVervolgvraag" type="boolean" default="false"
                                 minOccurs="0"/>
                        <element name="indicatorAfnemerIndicatie" type="boolean" default="false"
                                 minOccurs="0"/>
                        <element name="crossRefNummer" type="StUF:RefNummer"/>
                        <element name="foutmelding" type="string" minOccurs="0"/>
                     </sequence>
```

```
            </complexType>
         </element>
         <element name="bevestiging">
            <complexType>
               <sequence>
                  <element name="crossRefNummer" type="StUF:RefNummer"/>
               </sequence>
            </complexType>
         </element>
         <element name="fout">
            <complexType>
               <sequence>
                  <element name="crossRefNummer" type="StUF:RefNummer"/>
               </sequence>
            </complexType>
         </element>
      </choice>
   </sequence>
</complexType>
</element>
<!-- definitie van het foutbericht. -->
<element name="foutBericht">
   <complexType>
      <sequence>
         <element ref="StUF:stuurgegevens"/>
         <element name="body">
            <complexType>
               <sequence>
                  <element name="code" type="StUF:FoutCode"/>
                  <element name="plek" type="StUF:FoutPlek"/>
                  <element name="omschrijving" type="StUF:FoutOmschrijving"/>
               </sequence>
            </complexType>
         </element>
      </sequence>
   </complexType>
</element>
<!-- definitie van het bevestigingsbericht. -->
<element name="bevestigingsBericht">
   <complexType>
      <sequence>
         <element ref="StUF:stuurgegevens"/>
      </sequence>
   </complexType>
</element>
<!-- definitie van het omhullende element voor uitwisseling via een bestand -->
<element name="StUF-berichtenSet">
   <annotation>
      <documentation xml:lang="nl">
         het schema van de StUF-standaard kan dit element slechts gedefinieerd worden met als
         types anyType voor de die voor kunnen komen in een berichtenbestand. De verschillende
         sectormodellen definieren deze in meer detail. Een correcte validatie is mogelijk
         door in het bericht expliciet het sectormodel te specificeren
         het bericht gevalideerd dient te worden.
      </documentation>
   </annotation>
   <complexType>
      <sequence minOccurs="0" maxOccurs="unbounded">
         <choice>
            <element name="kennisgevingsBericht" type="anyType"/>
            <element name="vraagBericht" type="anyType"/>
            <element name="asynchroonAntwoordBericht" type="anyType"/>
            <element ref="StUF:foutBericht"/>
         </choice>
      </sequence>
   </complexType>
</element>
<!-- definitie van het transmissionError element tbv WSDL beschrijvingen -->
<element name="transmissionError" type="string"/>
<!-- definitie van door StUF gedefinieerde elementen -->
<element name="extraElement" nillable="true">
   <complexType>
      <simpleContent>
         <extension base="string">
            <attributeGroup ref="StUF:element"/>
            <attribute name="naam" type="string" use="required"/>
         </extension>
      </simpleContent>
   </complexType>
</element>
<!-- attribute groups met de attributen van fundamentele, relatie en tabel entiteiten -->
<attributeGroup name="fundamenteel">
   <attribute name="soortEntiteit" type="string" use="required" fixed="F"/>
```

```
      <attribute ref="StUF:sleutelVerzendend"/>
      <attribute ref="StUF:sleutelOntvangend"/>
      <attribute ref="StUF:sleutelGegevensbeheer"/>
      <attribute ref="StUF:verwerkingssoort"/>
      <attribute ref="StUF:noValue"/>
   </attributeGroup>
   <attributeGroup name="relatie">
      <attribute name="soortEntiteit" type="string" use="required" fixed="R"/>
      <attribute ref="StUF:sleutelVerzendend"/>
      <attribute ref="StUF:sleutelOntvangend"/>
      <attribute ref="StUF:sleutelGegevensbeheer"/>
      <attribute ref="StUF:verwerkingssoort"/>
      <attribute ref="StUF:noValue"/>
   </attributeGroup>
   <attributeGroup name="tabel">
      <attribute name="soortEntiteit" type="string" use="required" fixed="T"/>
      <attribute ref="StUF:verwerkingssoort"/>
      <attribute ref="StUF:noValue"/>
   </attributeGroup>
   <attributeGroup name="element">
      <attribute ref="StUF:noValue"/>
      <attribute ref="StUF:exact"/>
   </attributeGroup>
   <attribute name="sleutelVerzendend" type="StUF:Sleutel"/>
   <attribute name="sleutelOntvangend" type="StUF:Sleutel"/>
   <attribute name="sleutelGegevensbeheer" type="StUF:Sleutel"/>
   <attribute name="verwerkingssoort" type="StUF:Verwerkingssoort"/>
   <attribute name="indOnvolledigeDatum" type="StUF:IndOnvolledigeDatum" default="V"/>
   <attribute name="noValue" type="StUF:NoValue"/>
   <attribute name="exact" type="boolean" default="true"/>
   <!--  definitie van simpleTypes en complexTypes -->
   <complexType name="ExtraElementen">
      <sequence maxOccurs="unbounded">
         <element ref="StUF:extraElement" minOccurs="0"/>
      </sequence>
   </complexType>
   <complexType name="Systeem">
      <sequence>
         <element name="organisatie" minOccurs="0">
            <simpleType>
               <restriction base="string">
                  <maxLength value="10"/>
               </restriction>
            </simpleType>
         </element>
         <element name="applicatie">
            <simpleType>
               <restriction base="string">
                  <minLength value="3"/>
                  <maxLength value="20"/>
               </restriction>
            </simpleType>
         </element>
         <element name="administratie" minOccurs="0">
            <simpleType>
               <restriction base="string">
                  <maxLength value="1"/>
               </restriction>
            </simpleType>
         </element>
         <element name="gebruiker" minOccurs="0">
            <simpleType>
               <restriction base="string">
                  <maxLength value="20"/>
               </restriction>
            </simpleType>
         </element>
      </sequence>
   </complexType>
   <complexType name="DatumMetIndicator">
      <simpleContent>
         <extension base="StUF:Datum">
            <attributeGroup ref="StUF:element"/>
            <attribute ref="StUF:indOnvolledigeDatum"/>
         </extension>
      </simpleContent>
   </complexType>
   <complexType name="TijdvakGeldigheid">
      <sequence>
         <element name="begindatumTijdvakGeldigheid" type="StUF:DatumMetIndicator"
                  nillable="true"/>
         <element name="einddatumTijdvakGeldigheid" type="StUF:DatumMetIndicator"
                  nillable="true"/>
```

```
      </sequence>
   </complexType>
   <complexType name="TijdvakRelatie">
      <sequence>
         <element name="begindatumRelatie" type="StUF:DatumMetIndicator" nillable="true"/>
         <element name="einddatumRelatie" type="StUF:DatumMetIndicator" nillable="true"/>
      </sequence>
   </complexType>
   <simpleType name="Datum">
      <restriction base="decimal">
         <totalDigits value="8"/>
         <fractionDigits value="0"/>
      </restriction>
   </simpleType>
   <simpleType name="FoutCode">
      <restriction base="string">
         <length value="7"/>
      </restriction>
   </simpleType>
   <simpleType name="FoutPlek">
      <restriction base="string">
         <enumeration value="client"/>
         <enumeration value="server"/>
      </restriction>
   </simpleType>
   <simpleType name="FoutOmschrijving">
      <restriction base="string">
         <length value="200"/>
      </restriction>
   </simpleType>
   <simpleType name="IndicatorOvername">
      <restriction base="string">
         <enumeration value="I">
            <annotation>
               <documentation xml:lang="nl">Informatief</documentation>
            </annotation>
         </enumeration>
         <enumeration value="V">
            <annotation>
               <documentation xml:lang="nl">Verplicht</documentation>
            </annotation>
         </enumeration>
      </restriction>
   </simpleType>
   <simpleType name="IndOnvolledigeDatum">
      <restriction base="string">
         <enumeration value="M">
            <annotation>
               <documentation xml:lang="nl">maand en dag onbekend</documentation>
            </annotation>
         </enumeration>
         <enumeration value="D">
            <annotation>
               <documentation xml:lang="nl">dag onbekend</documentation>
            </annotation>
         </enumeration>
         <enumeration value="V">
            <annotation>
               <documentation xml:lang="nl">datum is volledig</documentation>
            </annotation>
         </enumeration>
      </restriction>
   </simpleType>
   <simpleType name="MaximumAantal">
      <restriction base="nonNegativeInteger">
         <totalDigits value="8"/>
      </restriction>
   </simpleType>
   <simpleType name="Mutatiesoort">
      <restriction base="string">
         <enumeration value="T">
            <annotation>
               <documentation xml:lang="nl">Toevoeging</documentation>
            </annotation>
         </enumeration>
         <enumeration value="W">
            <annotation>
               <documentation xml:lang="nl">Wijziging</documentation>
            </annotation>
         </enumeration>
         <enumeration value="V">
            <annotation>
               <documentation xml:lang="nl">Verwijdering</documentation>
```

```
            </annotation>
         </enumeration>
         <enumeration value="C">
            <annotation>
               <documentation xml:lang="nl">Correctie</documentation>
            </annotation>
         </enumeration>
      </restriction>
   </simpleType>
   <simpleType name="NoValue">
      <restriction base="string">
         <enumeration value="nietOndersteund"/>
         <enumeration value="nietGeautoriseerd"/>
         <enumeration value="geenWaarde"/>
         <enumeration value="waardeOnbekend"/>
      </restriction>
   </simpleType>
   <simpleType name="RefNummer">
      <restriction base="string">
         <maxLength value="12"/>
      </restriction>
   </simpleType>
   <simpleType name="Sleutel">
      <restriction base="string">
         <maxLength value="40"/>
      </restriction>
   </simpleType>
   <simpleType name="Sortering">
      <restriction base="nonNegativeInteger">
         <totalDigits value="2"/>
      </restriction>
   </simpleType>
   <simpleType name="Tijdstip">
      <restriction base="string">
         <pattern value="[0-9]{16}"/>
      </restriction>
   </simpleType>
   <simpleType name="Versienr">
      <restriction base="string">
         <minLength value="4"/>
         <maxLength value="4"/>
         <pattern value="[0-9][1-9][0-9][0-9]"/>
      </restriction>
   </simpleType>
   <simpleType name="Verwerkingssoort">
      <restriction base="string">
         <maxLength value="1"/>
         <enumeration value="T">
            <annotation>
               <documentation xml:lang="nl">Toevoeging</documentation>
            </annotation>
         </enumeration>
         <enumeration value="W">
            <annotation>
               <documentation xml:lang="nl">Wijziging of correctie</documentation>
            </annotation>
         </enumeration>
         <enumeration value="V">
            <annotation>
               <documentation xml:lang="nl">Verwijdering</documentation>
            </annotation>
         </enumeration>
         <enumeration value="E">
            <annotation>
               <documentation xml:lang="nl">Een relatie entiteit wordtbeeindigd.</documentation>
            </annotation>
         </enumeration>
         <enumeration value="I">
            <annotation>
               <documentation xml:lang="nl">
                  Entiteit bevat alleen identificerende gegevens.
               </documentation>
            </annotation>
         </enumeration>
         <enumeration value="R">
            <annotation>
               <documentation xml:lang="nl">
                  Een relatie entiteit wordt vervangen door een nieuwe relatie entiteit.
               </documentation>
            </annotation>
         </enumeration>
         <enumeration value="S">
            <annotation>
```

```
                <documentation xml:lang="nl">
                   De sleutel van een fundamentele entiteit wordt gewijzigd.
                </documentation>
             </annotation>
          </enumeration>
       </restriction>
    </simpleType>
</schema>
```

BIJLAGE 2: VOORBEELDBERICHTEN

Vraagbericht

In dit vraagbericht worden een aantal gegevens opgevraagd van personen die tussen 1985 en 1995 zijn geboren.

```
<?xml version="1.0" encoding="UTF-8"?>
<vraagBericht xmlns="http://www.egem.nl/StUF/sector/bg/0204"
xmlns:StUF="http://www.egem.nl/StUF/StUF0204"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.egem.nl/StUF/sector/bg/0204 bg0204.xsd">
   <StUF:stuurgegevens>
      <StUF:berichtsoort>Lv01</StUF:berichtsoort>
      <StUF:entiteittype>PRS</StUF:entiteittype>
      <StUF:sectormodel>BG</StUF:sectormodel>
      <StUF:versieStUF>0204</StUF:versieStUF>
      <StUF:versieSectormodel>0204</StUF:versieSectormodel>
      <StUF:zender>
         <StUF:organisatie/>
         <StUF:applicatie>app1</StUF:applicatie>
         <StUF:administratie/>
      </StUF:zender>
      <StUF:ontvanger>
         <StUF:organisatie/>
         <StUF:applicatie>app2</StUF:applicatie>
         <StUF:administratie/>
      </StUF:ontvanger>
      <StUF:referentienummer>123456</StUF:referentienummer>
      <StUF:tijdstipBericht>1234567890123456</StUF:tijdstipBericht>
      <StUF:vraag/>
   </StUF:stuurgegevens>
   <body>
      <!-- VANAF gedeelte van de vraag-->
      <PRS soortEntiteit="F">
         <geboortedatum>19850101</geboortedatum>
      </PRS>
      <!-- TOT EN MET gedeelte van de vraag -->
      <PRS soortEntiteit="F">
         <geboortedatum>19950101</geboortedatum>
      </PRS>
      <!-- De velden waarnaar gevraagd wordt -->
      <PRS soortEntiteit="F">
         <a-nummer       xsi:nil="true" StUF:noValue="geenWaarde"/>
         <bsn-nummer     xsi:nil="true" StUF:noValue="geenWaarde"/>
         <voornamen      xsi:nil="true" StUF:noValue="geenWaarde"/>
         <geslachtsnaam  xsi:nil="true" StUF:noValue="geenWaarde"/>
         <geboortedatum  xsi:nil="true" StUF:noValue="geenWaarde"/>
      </PRS>
   </body>
</vraagBericht>
```

Antwoordbericht

```
<?xml version="1.0" encoding="UTF-8"?>
<asynchroonAntwoordBericht xmlns="http://www.egem.nl/StUF/sector/bg/0204"
xmlns:StUF="http://www.egem.nl/StUF/StUF0204"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.egem.nl/StUF/sector/bg/0204 bg0204.xsd">
   <StUF:stuurgegevens>
      <StUF:berichtsoort>La01</StUF:berichtsoort>
      <StUF:entiteittype>PRS</StUF:entiteittype>
      <StUF:sectormodel>BG</StUF:sectormodel>
      <StUF:versieStUF>0204</StUF:versieStUF>
      <StUF:versieSectormodel>0204</StUF:versieSectormodel>
      <StUF:zender>
         <StUF:organisatie/>
         <StUF:applicatie>app2</StUF:applicatie>
         <StUF:administratie/>
      </StUF:zender>
      <StUF:ontvanger>
         <StUF:organisatie/>
         <StUF:applicatie>app1</StUF:applicatie>
         <StUF:administratie/>
      </StUF:ontvanger>
      <StUF:referentienummer>1234567</StUF:referentienummer>
      <StUF:tijdstipBericht>1334567890123456</StUF:tijdstipBericht>
      <StUF:antwoord>
         <StUF:crossRefNummer>123456</StUF:crossRefNummer>
      </StUF:antwoord>
   </StUF:stuurgegevens>
   <body>
      <PRS soortEntiteit="F">
         <a-nummer>1123456789</a-nummer>
         <bsn-nummer>100000001</bsn-nummer>
         <voornamen>Henri Peter</voornamen>
         <geslachtsnaam>Korver</geslachtsnaam>
         <geboortedatum>19680815</geboortedatum>
      </PRS>
   </body>
</asynchroonAntwoordBericht>
```