Standaard Uitwisseling Formaat voor applicaties

StUF 03.00: Kandidaat Aanbeveling

Inhoudsopgave

1. INLEIDING 5

1.1 Conventies 6

2. GLOBALE FUNCTIONALITEIT EN OPZET VAN STUF 7

2.1 Inleiding 7
2.2 Relatie berichten tot de werkelijkheid 8
2.3 Historische gegevens 10
2.4 Gegevensmanagement 11
2.5 Transacties 12
2.6 Vrije berichten 13
2.7 Berichtenlogistiek 13

3. CONTENTMODEL 14

3.1 Gewenste functionaliteit en semantiek 14
3.1.1 Soorten entiteittypen en hun plaats in een bericht 14
3.1.2 De structuur van een object in een bericht 15
3.1.3 Identificatie: kerngegevens en systeemsleutels 15
3.1.4 Gegevensgroepen 16
3.1.5 Historische gegevens 16
3.1.6 Robuuste berichtverwerking 17
3.2 De syntax voor een object in een bericht 17
3.2.1 Metagegevens voor een fundamenteel, een relatie- en een tabelentiteittype 17
3.2.2 De structuur van een object 18
3.2.3 Het opnemen van metagegevens ten behoeve van historie 19
3.2.4 Samengestelde elementen in een entiteittype 20
3.2.5 Het opnemen van niet in het sectormodel gedefinieerde elementen 21
3.3 Het opnemen van elementen en relatie-entiteiten in een entiteit 21
3.3.1 Het opnemen van elementen in een entiteit 21
3.3.2 Het opnemen van relatie-entiteit in een entiteit 23

4. BERICHTVERWERKING: STURING, LOGISTIEK EN FOUTAFHANDELING 24

4.1 Codering van het type bericht 24
4.1.1 Versie StUF en sectormodel 24
4.1.2 Berichtcode 24
4.1.3 Entiteittype en functie 25
4.2 Adressering zender en ontvanger 25
4.3 Identificatie en volgorde 26
4.3.1 Identificatie van berichten 26
4.3.2 De volgorde waarin de berichten worden verwerkt 26
4.4 Berichtenlogistiek en foutafhandeling 26
4.4.1 Regels voor bevestigingsberichten 28
4.4.2 Regels voor triggerbericht 29
4.4.3 Regels voor foutberichten 29

5. KENNISGEVING- EN SYNCHRONISATIEBERICHTEN 32

5.1 Stuurgegevens voor kennisgeving- en synchronisatieberichten 33
5.2 Respons en foutafhandeling voor kennisgeving- en synchronisatieberichten 35
5.3 Regels voor enkelvoudige kennisgevingberichten 35
5.3.1 De structuur van objecten in enkelvoudige kennisgevingberichten 35
5.3.2 Het attribute verwerkingssoort 36
5.3.3 Het vullen van de <object> elementen 36
5.3.4 Het vullen van relatie-entiteiten en gerelateerde entiteiten 38
5.3.5 Toevoegen/wijzigen gerelateerde entiteit 42
5.3.6 Aanvullende regels 42

5.4 Regels voor samengestelde kennisgevingberichten 43
5.5 Regels voor synchronisatieberichten 43
5.5.1 Synchronisatiebericht actueel 43
5.5.2 Synchronisatiebericht historisch 44
5.5.3 Foutafhandeling 45

6. VRAAG- EN ANTWOORDBERICHTEN 46

6.1 Stuurgegevens voor vraagberichten 46
6.2 Stuurgegevens voor antwoordberichten 48
6.3 Regels voor vraagberichten 50
6.3.1 Het specificeren van selectiecriteria 50
6.3.2 Het bevragen op sleutel 52
6.3.3 Het specificeren van de gevraagde gegevens 52
6.3.4 Het stellen van een vervolgvraag 53
6.4 Regels voor antwoordberichten 54
6.4.1 Het opnemen van objecten in een antwoordbericht 54
6.4.2 Het vullen van objecten in een antwoordbericht 55
6.4.3 Het omgaan met historische gegevens 56
6.4.4 Foutafhandeling 59

7. VRIJE BERICHTEN 60

7.1 Interactiepatronen en berichtcodes 60
7.2 Gereserveerde elementen en structuur van het vrije bericht 60
7.2.1 Het opnemen van losse gegevens en meldingen 61
7.2.2 Elementen voor een entiteittype uit het sectormodel 61
7.2.3 Het wijzigen van gegevens 62
7.2.4 Het opvragen/selecteren van gegevens 62
7.2.5 Zaakinformatie 62

8. PROTOCOLBINDINGEN 63

8.1 Uitwisseling via een bestand 63
8.2 Berichtuitwisseling op basis van WSDL, SOAP en http 63
8.2.1 Het gebruik van het <message> element binnen StUF 64
8.2.2 Het gebruik van het <portType> element binnen StUF 64
8.2.3 Het gebruik van het <binding> element binnen StUF 65
8.2.4 Het gebruik van het <service> element binnen StUF 67

TABEL MET MOGELIJKE FOUTBERICHTEN 68

OPEN ISSUES: 70

REFERENTIES 71

Versie StUF030002: Wijzigingen vergeleken met StUF030001

Hoofdstuk 7	Een inleiding toegevoegd analoog aan de inleiding bij de hoofdstukken 5 en 6, plus het hoofdstuk anders ingedeeld zonder functionele wijzigingen aan te brengen
Figuur 2	Mnemonics vervangen door volledige elementnamen
Figuur 3	Vervangen door een eenvoudiger figuur
Figuur 4	Wegvallen tekst gecorrigeerd
Hoofdstuk 2	Subhoofdstuk 2.2 opgenomen in de 2.1 Inleiding en tekstueel gewijzigd.
Tabel met foutberichten	Hoofdstuk met een tabel met alle foutberichten toegevoegd
Referenties	Hyperlinks voor referenties naar StUF-standaard aangepast. Referentie naar STUFBGWSDL verwijderd.

Versie StUF030001: Wijzigingen vergeleken met StUF030000

Titelpagina:	StUF 02.10 --> StUF 03.00
p. 50 in tabel 6.2:	*VoorgaandRefNummer --> SequenceNumber*
Voetnoot `10	Deze voetnoot is opgenomen bij het eerste voorkomen van notificatie in de tekst en nog wat verduidelijkt.
Synchronisatiebericht	Het element `<gerelateerde>` heeft niet onmiddellijk een complexType maar wordt gevuld met elementen voor de verschillende gerelateerden die mogen voorkomen. De namen van deze elementen worden gedefinieerd in het sectormodel. Verduidelijkt dat synchrone synchronisatieberichten dezelfde body hebben als asynchrone. Nog wat verduidelijkingen doorgevoerd.
Historie in antwoordberichten	`<historie>` element niet langer voor de elementen voor relaties maar als laatste element binnen het `<object>` element in de body van een antwoordbericht (In overeenstemming met de vrijheid die een berichtontwerper heeft bij het plaatsen van elementen voor attributen en relaties en handiger bij het definiëren van de schema's).
Kennisgeving-, synchronisatie-, vraag- en antwoordbericht	Duidelijker gespecificeerd hoe de berichtstuurgegevens entiteittype en functie gevuld dienen te worden.
5.5.2 eerste alinea	Expliciet gemaakt dat bij de verwerking van een synchronisatiebericht historisch alleen de geleverde (historische) gegevens vervangen dienen te worden. De verwerking kan complex zijn, als vanuit meerdere bronnen historische gegevens worden geleverd.
7.2 een-na-laatste alinea	Voorbeeld over SQL operatie uit tekst verwijderd, omdat deze tekst verkeerde suggesties wekt.
Voetnoot 11	Verwijderd, want voor elke `<operation>` in een `<portType>` dient ook een `<operation>` met de binding in het `<binding>` element gedefinieerd te worden.
Vulling attribute soapAction (blz. 68)	Gewijzigd van “” naar “http://www.egem.nl/StUF”, omdat XMLSPY “” niet accepteert.

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 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 tussen 1996 en 1998 onder auspiciën van de VNG het Standaard Uitwisseling Formaat (StUF) gedefinieerd. Deze standaard is bekend onder de naam StUF 01.05. StUF 01.05 berichten maken gebruik van TCP/IP en sluiten qua structuur aan op GBA-berichten conform het Logisch Ontwerp versie 3.

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 2005 is daarom een vernieuwde versie van de StUF-standaard uitgebracht met ruwweg dezelfde functionaliteit als StUF 01.05, maar nu gebruikmakend van XML, SOAP en WSDL[1]. Deze versie van de StUF-standaard kreeg als versienummer 02.04. De VNG heeft bij het uitbrengen van StUF 02.04 het beheer overgedragen aan EGEM. StUF 02.04 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.

Snel na het uitbrengen van StUF 02.04 bleek de landelijke voozieningen voor de basisregistratie Adressen en Gebouwen behoefte te hebben aan wat extra functionaliteit. Deze functionaliteit is gedefinieerd in StUF 02.05. Uit discussies met bijvoorbeeld de gemeente 's-Gravenhage bleek, dat StUF 02.04 onvoldoende voldeed aan een aantal uitgangspunten van een service geöriënteerde architectuur. Ervaringen in de gemeente Amsterdam wezen uit dat de StUF 02.04 onvoldoende functionaliteit bood voor het synchroniseren van gegevens. Daarnaast was er de behoefte om naast de door de StUF-standaard voorgedefinieerde semantiek voor berichten ook berichten te kunnen definiëren die wel gebruik maken van het StUF-contentmodel, maar een eigen semantiek hebben. Deze wensen en een aantal kleinere verbeteringen zijn opgenomen in deze nieuwe versie van de StUF-standaard: versie 03.00. Het is versie 03.00 geworden, omdat deze versie een breaking change is ten opzichte van 02.04 en 02.05. Met deze versie voldoet StUF nu aan de uitgangspunten van een service geöriënteerde architectuur. De verwachting is dat deze versie een aantal jaren stabiel zal blijven en zonder breaking changes uitgebreid kan worden met nieuwe functionaliteit.

Bij het definiëren van StUF 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 een 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 de StUF-standaard.

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 en om een aantal uitgangspunten die aan het ontwerp van StUF ten grondslag liggen over het voetlicht te brengen. Dit hoofdstuk is bedoeld voor mensen die in StUF geïnteresseerd zijn, maar geen technische achtergrond hebben.

In hoofdstuk 3 wordt beschreven hoe StUF objecten in een bericht opneemt. Dit hoofdstuk legt de basis voor een grote mate van herbruikbaarheid van elementen in StUF-berichten.

Hoofdstuk 4 beschrijft de gegevens ten behoeve van de adressering en de aansturing van de berichtverwerking. Dit hoofdstuk gaat ook in de verschillende door StUF ondersteunde interactiepatronen en de foutafhandeling.

De hoofdstukken 5 t/m 7 gaan in op de belangrijkste functionaliteiten van StUF: Hoofdstuk 5 beschrijft de berichten ten behoeve van het notificeren van events, het synchroniseren van databases en het aanbieden van databasetransacties. Dit hoofdstuk behandelt de kennisgeving- en synchronisatieberichten. Hoofdstuk 6 beschrijft de

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

berichten ten behoeve van het opvragen van gegevens in een database, de zogenaamde vraag- en antwoordberichten. Hoofdstuk 7 beschrijft hoe met behulp van StUF berichten gedefinieerd kunnen worden met een niet door de StUF-standaard voorgeschreven semantiek, de zogenaamde vrije berichten.

Hoofdstuk 8 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 3 tot en met 8 bevatten de volledige specificatie voor de berichtverwerkende software. Deze hoofdstukken richten zich primair op de ontwerpers en bouwers van software.

1.1 Conventies

In dit document wordt de term attribuut gebruikt in de zin van een attribuut van een entiteittype in een Entiteit Relatie Diagram. De term attribute wordt gebruikt om een attribute van een XML-element aan te duiden.

Dit document hanteert de volgende conventies voor de formattering:

Italic: in *italic* geformatteerde tekst duidt een begrip aan dat gebruikt wordt voor de aansturing van de berichtverwerking via de berichtstuurgegevens of via de parameters voor een vraag/antwoord of kennisgevingbericht

`Courier`: in `Courier` geformatteerde tekst bevat een in de StUF-standaard of onderliggende standaard gespecificeerd XML-attribute of XML-element (een element is altijd omgeven door < en >: `<element>`).
Ook alle voorbeelden van XML-berichten en stukjes schema zijn geformatteerd in Courier.

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 heeft deze problematiek ook onderkend 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. Hier wordt de term Master-Data Management gebruikt.

Een bericht heeft altijd betrekking op een concreet object. Denk bijvoorbeeld 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. Deze twee berichten hebben betrekking op objecten uit heel verschillende sectoren: de GBA en het kentekenregister van de RDW. Zonder standaardisatie is de kans groot dat een persoon binnen een bericht van de RDW er heel anders uitziet dan een persoon binnen een GBA-bericht, terwijl functioneel soortgelijke eisen gelden.

Een template-berichtdefinitie die de functionaliteit van berichten voor het uitwisselen van gegevens over objecten en voor het opvragen van gegevens standaardiseert kan hier goede diensten bewijzen. Een template-berichtdefinitie beschrijft domeinonafhankelijk de syntax en semantiek van de berichten. Om met deze template-berichtdefinitie een berichtenschema voor een sector te definiëren, heb je nog 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.

De StUF-standaard is zo'n template-berichtdefinitie. In deze standaard worden alleen voor heel algemene situaties als het bevestigen van de ontvangst van het bericht of de foutafhandeling concrete berichten gedefinieerd. Alle andere berichten worden op basis van de voorschriften in de StUF-standaard gedefinieerd in de sectormodellen. 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 StUF is om zoveel functionaliteit te definiëren dat voldaan kan worden aan de eisen van gemeenten, de authentieke registraties van de rijksoverheid en het Master-Data Management binnen grote bedrijven.

De twee eenvoudigste berichtuitwisselingspatronen zijn: notificatie[2] en verzoek-respons. Bij notificatie verwacht de zender van het bericht geen respons van de ontvanger. Bij verzoek-respons verwacht de zender wel een respons. Voor wat betreft de koppeling tussen een verzoek en een respons is er een onderscheid tussen synchroon en asynchroon berichtenverkeer. Bij synchroon berichtenverkeer wordt een antwoord verwacht op de verbinding waarover het bericht is verzonden. De verzender wacht, totdat het antwoord over die verbinding is ontvangen of oordeelt dat er sprake is van een fout (time-out of niet verwacht antwoord). Bij asynchroon berichtenverkeer wordt het bericht verzonden, maar wordt er geen respons verwacht op de verbinding waarover het bericht is verstuurd. De verzender dient te wachten, totdat de ontvanger van het bericht zelf verbinding zoekt om een respons te geven.

StUF ondersteunt alleen deze twee uitwisselingspatronen in de synchrone en de asynchrone variant. Patronen, waarbij de interactie bestaat uit meer dan twee berichten dienen te worden opgebouwd uit de één- ot twee-

[2]Het begrip notificatie wordt hier anders gebruikt dan binnen de wsdl-definitie. Daar staat een notification voor een van de service uitgaand bericht waarop geen antwoord wordt verwacht. Een op de service inkomend bericht, waarop geen antwoord wordt verwacht is in wsdl-termen one-way. Omdat het hier gaat om uitwisselingspatronen op business niveau en one-way een weinig zeggende term is die in de wsdl specificatie op een technisch niveau wordt gedefinieerd, wordt in de StUF-standaard de voorkeur gegeven aan het gebruik van de term notificatie voor een bericht waarop geen antwoord wordt verwacht ongeacht of dit bericht op de service binnenkomt of naar buiten gaat.

berichtspatronen van StUF. StUF geeft hier echter geen voorschriften of richtlijnen voor. Hulpmiddelen voor procesorkestratie als BPEL[3] bieden hier ondersteuning voor.

Voor het doorgeven van wijzigingen worden in StUF kennisgevingberichten gebruikt. Een kennisgevingbericht kan direct verwerkt moeten worden, opdat de zender weet of de verwerking geslaagd is. Dit wordt ook wel aangeduid als een transactie. In dat geval wordt een synchroon kennisgevingbericht gebruikt. In andere gevallen zal een zender het voldoende vinden een afnemer op de hoogte te stellen van een wijziging en is onmiddellijke verwerking niet nodig. In dat geval wordt een asynchroon kennisgevingbericht gebruikt. Voor het opvragen van de toestand in de database worden in StUF vraag/antwoordberichten gebruikt. De gevraagde gegevens kunnen onmiddellijk worden geleverd (synchroon vraag/antwoord) of ze worden pas na verloop van tijd geleverd (asynchroon vraag/antwoord).

De opkomst van de Service Geöriënteerde Architectuur (SGA) heeft geleid tot de behoefte aan een nieuw type berichten naast de hierboven reeds onderkende berichten ten behoeve van het uitwisselen van gegevens over objecten en voor het opvragen van gegevens over objecten. De StUF-standaard heeft voor deze berichten niet als functie het in detail definiëren van de semantiek. Dit is de verantwoordelijkheid van de aanbieder van de service. Vanuit het oogpunt van standaardisatie en hergebruik is het wel belangrijk dat waar mogelijk de modelgedreven berichtstructuur gebruikt voor kennisgevingen en vraag/antwoord ook wordt gebruikt binnen deze berichten. In het vervolg zullen we de berichten met een door de berichtontwerper gedefinieerde semantiek aanduiden als vrije berichten. Paragraaf 2.6 gaat wat dieper in op dit type berichten. De paragrafen 2.2 t/m 2.5 gaan dieper in op de benodigde functionaliteit voor het uitwisselen en het opvragen van gegevens over objecten. Paragraaf 2.7 gaat kort in op de functionaliteit nodig voor het starten en stoppen van de verzending van berichten.

2.2 Relatie berichten tot de werkelijkheid

Kennisgevingberichten hebben betrekking op de werkelijkheid
Berichten worden normaliter uitgewisseld tussen geautomatiseerde systemen met een database. Een organisatie heeft die database vaak ingericht, omdat een afbeelding van de werkelijkheid nodig is. Toch is de werkelijkheid en niet de database is het semantisch relevante niveau, want aan een wijziging van de database zal een gebeurtenis in de werkelijkheid vooraf gaan. Die gebeurtenis in de werkelijkheid is de aanleiding voor het verzenden van een kennisgeving. Voor de betekenis van kennisgevingberichten wordt daarom niet uitgegaan van inserts, updates en deletes in de databases maar van gebeurtenissen in de werkelijkheid.

Typen kennisgevingberichten
Bij kennisgevingberichten is de relatie tot de werkelijkheid complex. Een eerste vraag is op hoeveel objecten een kennisgevingbericht betrekking heeft. In StUF heeft met het oog op reductie van complexiteit een kennisgevingbericht betrekking op precies één object in de werkelijkheid. De volgende gebeurtenissen in de werkelijkheid kunnen de aanleiding zijn 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.

[3]BPEL = Business Process Execution Language

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.

Vraag/antwoord berichten
Bij vraag/antwoord berichten worden gegevens in de database van het ontvangende systeem opgevraagd. Een antwoordbericht geeft dus de toestand in de database van het antwoordende systeem. Als deze database bij is, dan is dit tevens de toestand in de werkelijkheid. Het verschil met kennisgevingen is dat een kennisgeving meestal wordt verzonden naar aanleiding van een gebeurtenis in de werkelijkheid. Een vraag/antwoord bericht wordt uitgewisseld op het moment dat de vragende partij iets wil weten. Bij vraag/antwoordberichten is er dus geen link met een gebeurtenis in de werkelijkheid.

Speciale waarden van eigenschappen
Bij het definiëren van een waardebereik voor een eigenschap is het van belang er rekening mee te houden 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 onder andere 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 dergelijke waarden buiten het normale waardebereik 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. Volgens de datamodelleringstheorie zijn er twee soorten eigenschappen:
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 een relatie legt naar een ander object, bijvoorbeeld de ouder van een persoon.

Of een eigenschap wordt gezien als een relatie is enigszins arbitrair. De ontwerper van een objectmodel hoeft namelijk niet het objecttype te onderkennen waarnaar een relatie ligt. Het adres van een persoon kan bijvoorbeeld 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 tot en met Logisch Ontwerp 3 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.

2.3 Historische gegevens

De eigenschappen van een object kunnen in de tijd variëren, bijvoorbeeld doordat een persoon een aantal keren verhuist. Een object kan ook niet langer bestaan, bijvoorbeeld een overleden persoon. In het eerste geval spreken we over een historisch gegeven en in het tweede geval over een historisch object. Een historisch gegeven is een gegeven van een object dat niet meer overeenkomt met de werkelijkheid. Een historisch object is een object dat in de werkelijkheid niet meer bestaat maar nog wel van belang is. Een historisch object heeft dus actuele gegevens, als hun waarden overeenstemmen met de laatste (actuele) waarden voordat het object ophield te bestaan.

Bij het uitwisselen van historische gegevens spelen voor een standaard twee problemen:
1. Is de semantiek van de standaard voldoende rijk?
2. Zijn de gebruikers van de standaard bereid om de extra complexiteit voor historie te implementeren in hun systemen?

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 is er daarom voor gekozen om het wijzigen van historische gegevens slechts te ondersteunen door middel van het opnieuw aanbieden van het object inclusief al zijn historie. StUF ondersteunt dus niet het gericht corrigeren van een enkel foutief historisch gegeven.

Wanneer een bericht het gevolg is van een gebeurtenis, dan bepaalt die gebeurtenis de eindgeldigheid van de oude waarde en de begingeldigheid 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 eindgeldigheid voor een waarde. Dit kan op drie manieren worden geïmplementeerd:

1. Door per waarde een tijdvak geldigheid in de berichten op te nemen
   Een attribuut kan meervoudig voorkomen in een object met per voorkomen een tijdvak geldigheid.
2. Door per groep van attributen een tijdvak geldigheid in de berichten op te nemen
   De groepen worden gedefinieerd in het sectormodel. Meerdere voorkomens van een groep worden binnen een object opgenomen met verschillende tijdvakken geldigheid. Alle attribuutwaarden binnen de groep zijn geldig gedurende dat tijdvak geldigheid.
3. Door voor alle attribuutwaarden in een object op objectniveau een tijdvak geldigheid op te nemen
   Meerdere voorkomens van een object worden in een bericht opgenomen met elk een tijdvak geldigheid. Er is precies één voorkomen met de actuele waarden. De andere voorkomens bevatten historische waarden.

De derde keuze leidt tot de simpelste berichtverwerking, omdat de actuele gegevens eenvoudig te vinden zijn en sluit het beste aan bij de wijze waarop in veel databases met historische gegevens wordt omgegaan. De tweede keuze sluit aan bij de manier waarop in de GBA met historische gegevens wordt omgegaan. De eerste keuze maakt het werken met attributen met kardinaliteit groter dan één ingewikkelder, omdat moet worden nagegaan of een attribuut meervoudig voorkomt vanwege de kardinaliteit of vanwege het historisch zijn van de waarde. Alleen de tweede en derde keuze worden in de StUF-standaard uitgewerkt.

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 de objecten waartussen de relatie ligt. Het tijdstip van het begin en van het eind van de relatie markeren een tijdvak geldigheid voor het object. De eenvoudigste oplossing is om het ontstaan en beëindigen van relaties onafhankelijk van het tijdvak geldigheid voor de attribuutwaarden van het object waaruit de relatie ontspringt in berichten op te nemen. Om de omgang met de historie van relaties in de standaard te kunnen definiëren is het noodzakelijk in de standaard het bestaanstijdvak van een relatie en de gewenste functionaliteit eromheen te definiëren.

Historische gegevens worden via antwoordberichten gecommuniceerd door in één bericht meerdere voorkomens van een object op te nemen met elk een tijdvak geldigheid. De vraagsteller kan in het vraagbericht 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 werkt een andere keuze beter. In kennisgevingberichten voor een wijziging of correctie worden twee voorkomens van het gewijzigde object opgenomen: het eerste met de waarden geldig tot de wijzigingsdatum (het ‘oude’ voorkomen) en het tweede 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 de ‘nieuwe’ voorkomen de ‘nieuwe’ situatie. Het bestaanstijdvak en het tijdvak geldigheid relatie specificeren samen de benodigde informatie ten behoeve van de opbouw van historie voor relaties. Het tijdvak geldigheid op objectniveau is niet relevant voor de historie van een relatie. Dit heeft uitsluitend betrekking op de attribuutwaarden.

Bij het opbouwen van historie via kennisgevingen wordt de oudste situatie het eerst gezonden, daarna de één-na-oudste situatie enzovoorts, totdat de actuele situatie is bereikt. Een systeem dat historie negeert, kan al deze berichten gewoon verwerken en eindigt met de actuele gegevens. Voorwaarde is wel dat er nooit kennisgevingen worden verzonden met wijzigingen in historische gegevens. Een systeem dat geen historische gegevens kent, zal zo'n kennisgeving namelijk gewoon verwerken en eindigt met foutieve actuele gegevens. Voor het corrigeren van historische gegevens heeft de StUF-standaard een ander mechanisme: het synchronisatiebericht.

2.4 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.

2.5 Transacties
Niet altijd kan in één kennisgevingbericht een volledige logische transactie worden ondergebracht. Er is dus behoefte aan de mogelijkheid om meerdere kennisgevingberichten te groeperen tot één transactie. Hierbij kunnen een paar keuzes gemaakt worden:
1. Wordt een transactie samengesteld uit kennisgevingberichten of definiëren we hier een nieuw mechanisme voor?
2. Mag een transactie betrekking hebben op verschillende entiteittypen en mutatiesoorten?
3. Mogen binnen een asynchrone transactie verschillende overname indicatoren voorkomen?

Het definiëren van een nieuw mechanisme impliceert dat bij het implementeren van samengestelde transacties geen gebruik gemaakt kan worden van reeds bestaande verwerking voor de eenvoudiger en vaak voorkomende atomaire kennisgevingen. Om die reden is gekozen voor het opbouwen van samengestelde transacties uit atomaire kennisgevingen.

Een kennisgevingbericht heeft altijd betrekking op één object van een bepaald entiteittype met één mutatiesoort. Deze beperking hoeft niet opgelegd te worden aan een samengestelde transactie, omdat de kennisgevingen in een samengestelde transactie sowieso atomair verwerkt moeten kunnen worden. Het groeperen van kennisgevingberichten voor verschillende entiteittypen stelt dus geen extra eisen aan de implementaties behalve de onontkoombare eis dat ontvangende systemen de samengestelde kennisgevingberichten als één transactie dienen te verwerken. Hetzelfde geldt voor het groeperen van kennisgevingen met verschillende mutatiesoorten. Binnen een samengesteld kennisgevingbericht mogen daarom atomaire kennisgevingen met verschillende mutatiesoorten voorkomen. Het ligt anders met de overname indicatoren. Binnen een samengestelde kennisgeving wordt verondersteld dat alle atomaire kennisgevingen verwerkt worden. De overname indicatoren van de atomaire kennisgevingen dienen derhalve gelijk te zijn aan de overname indicator van de samengestelde kennisgeving.

2.6 Vrije berichten

De StUF-standaard is ontstaan, omdat er in gemeenten behoefte was aan het uitwisselen van gegevens (kennisgevingen) en het opvragen van gegevens (vraag/antwoord). Dit zie je terug in de tot nu toe beschreven functionaliteit. Binnen servicegeoriënteerde architecturen is er behoefte aan meer functionaliteit. De StUF-standaard speelt hier op in met zogenaamde vrije berichten. Een vrij bericht is een bericht waarvan de berichtontwerper zelf de semantiek kan definiëren.

Op het niveau van een template-berichtdefinitie hoeft er voor vrije berichten veel minder gespecificeerd te worden dan voor kennisgeving- en vraag/antwoordberichten. Het definiëren van de semantiek is per slot van rekening de verantwoordelijkheid van de ontwerper van het vrije bericht. Welk respons er moet volgen op de aanroep van een 'vrij bericht' service wordt dus niet beschreven in de StUF-standaard. Zo'n respons is wel altijd een vrij bericht. De StUF-standaard stelt wel een aantal eisen aan een vrij bericht. Deze eisen worden besproken in hoofdstuk 7.

2.7 Berichtenlogistiek

StUF maakt veel gebruik van asynchrone berichten. Hierdoor zijn de verwerkingsprocessen bij de zender en de ontvanger van elkaar ontkoppeld. De zender zet berichten voor een ontvanger klaar. Het is niet noodzakelijk dat de ontvanger op het moment van het aanmaken de berichten kan ontvangen. Zender en ontvanger kunnen procedurele afspraken maken over het verzenden/ontvangen van de berichten, bijvoorbeeld tussen 20u00 en 22u00 is de webservice voor het ontvangen van berichten actief. Of de webservice is actief van 7u00 tot 19u00, zodat alle berichten die overdag worden aangemaakt, onmiddellijk kunnen worden verzonden.

In de praktijk verschilt de beschikbaarheid van het zendende en ontvangende systeem vaak. Een broker zal bijvoorbeeld vrijwel 7x24 uur beschikbaar zijn en een ontvangend systeem slechts gedurende een paar uur per dag. In dat soort gevallen is het gemakkelijk, wanneer het ontvangende systeem zelf het initiatief kan nemen om de klaarstaande berichten te laten versturen. StUF ondersteunt dit door middel van het zogenaamde triggerbericht. Na ontvangst van een triggerbericht dient het ontvangende systeem binnen vijf minuten te starten met het verzenden van de voor de verzender van het triggerbericht klaarstaande berichten. De berichtverzending stopt pas als er geen te verzenden berichten meer zijn. Ook tijdens het verzenden aangemaakte berichten dienen dus verzonden te worden.

Als de ontvanger van het triggerbericht verwacht binnen vijf minuten te kunnen starten met de berichtverzending, dan wordt op het triggerbericht gereageerd met een bevestigingsbericht, anders wordt er gereageerd met een foutbericht.

3. Contentmodel

StUF is bedoeld om gegevens eenvoudig te kunnen uitwisselen tussen systemen. Waar mogelijk baseert StUF zich op een model van de uit te wisselen gegevens in de vorm van entiteittypen, hun relaties en hun attributen, een zogenaamd entiteitrelatiediagram of ERD met de bijbehorende beschrijvingen van attributen en relaties. Dit hoofdstuk beschrijft hoe de gegevens van een object dat voorkomt in het ERD in de body van een bericht worden opgenomen. Dit hoofdstuk doet dit in termen onafhankelijk van een concreet objecttype als persoon of adres. Dit hoofdstuk gaat dus in op de algemene structuur van een object binnen een bericht. Het definiëren concrete structuur van een object in een bericht in de vorm van elementen voor de attributen en relaties van een objecttype. Dit doet de ontwerper van het sectormodel. De richtlijnen voor het maken van een sectormodel zijn opgenomen in een aparte document, omdat de StUF-standaard bewust de berichtontwerpers nog veel vrijheid laat.

De eerste paragraaf gaat in op de gewenste functionaliteit en de bijbehorende semantische aspecten. De tweede paragraaf gaat dieper in op de syntax voor een object in het bericht. De derde paragraaf beschrijft hoe en wanneer de waarden van gegevens of een relatie in een bericht worden opgenomen. De hierna hoofdstukken vullen de hier gegevens regels nog aan met regels specifiek voor een bepaalde berichtsoort.

3.1 Gewenste functionaliteit en semantiek

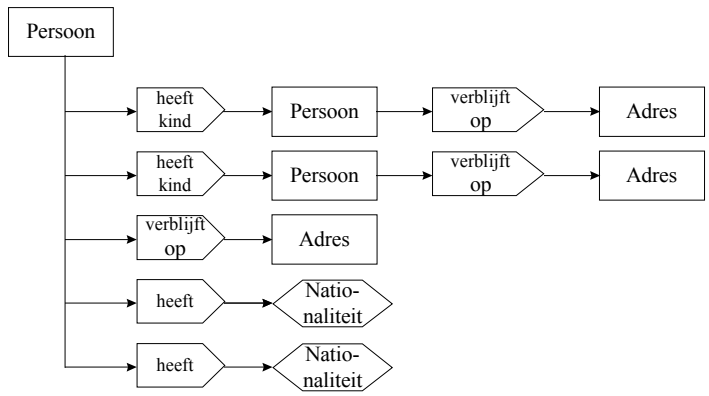
3.1.1 Soorten entiteittypen en hun plaats in een bericht

StUF onderscheidt binnen een entiteitrelatiediagram drie 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.
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.
3. *relatie-entiteittypen*
   Een relatie-entiteittype representeert een relatie tussen twee entiteittypen, bijvoorbeeld tussen een persoon en een adres. Tussen persoon en adres bestaan verschillende relaties, bijvoorbeeld PERSOON.verblijft op.ADRES[4], 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 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 in het ERD 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 worden geen berichten gedefinieerd, omdat een relatie-entiteit afhankelijk is van het object van waaruit de relatie ligt. De keuze of iets een relatie-entiteittype is of een fundamenteel entiteittype is overigens enigszins arbitrair. Desgewenst kan een huwelijk bijvoorbeeld ook als een fundamenteel entiteittype gedefinieerd worden. De relaties naar de twee huwelijkspartners worden dan gedefinieerd als een relatie-entiteittype met kardinaliteit twee van huwelijk.

[4] 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.

3.1.2 De structuur van een object in een bericht



Figuur 1: 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 relaties en de eventuele attributen van de relatie
- 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:
- Fundamentele en tabelentiteiten mogen in alle berichten met entiteiten in de body als kind van het `<body>`-element voorkomen
- Relatie-entiteiten mogen alleen in vrije berichten als kind van het `<body>`-element 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.

3.1.3 Identificatie: kerngegevens en systeemsleutels

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 niet altijd voldoende om een persoon te identificeren. Denk aan twee gezinsleden met dezelfde voorletters die 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 kent StUF het begrip kerngegevens. Dit is een deelverzameling van de attributen en relaties van een entiteittype aan de hand waarvan een object kan worden geïdentificeerd. Bij personen kan bijvoorbeeld voor de volgende set identificerende gegevens gekozen worden:
- SoFi-nummer
- A-nummer
- Naamgegevens (Geslachtsnaam, voorvoegsels en voorletters)
- Verblijfsadres
- Geboortedatum
- Geslacht

Bij een aantal berichtsoorten zal in de volgende hoofdstukken het al dan niet verplicht zijn van de kerngegevens gespecificeerd worden. In het sectormodel dienen voor elk entiteittype de verzameling kerngegevens gedefinieerd te worden. Als bij een relatie entiteittype geen kerngegevens zijn gespecificeerd, dan mag ervan uitgegaan worden dat het

fundamentele entiteittype vanwaaruit de relatie ligt en de gerelateerde de kerngegevens zijn. Er wordt geen gereserveerd element gedefinieerd voor de kerngegevens, omdat dit leidt tot inflexibiliteit bij het definieren van de berichten in een sectormodel.

Systemen identificeren de voorkomens 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 voorkomens, worden deze sleutels in StUF als afzonderlijk attribuut onderkend. In het sectormodel hoeven deze sleutels niet opgenomen te worden. De attributen voor de systeemsleutels worden beschreven in paragraaf 3.2.2

3.1.4 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. De standaard 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.

3.1.5 Historische gegevens

Onder een historisch gegeven verstaan we een gegeven met een waarde die vroeger geldig was maar nu niet meer overeenkomt met de waarde in de werkelijkheid. Als historie relevant is, dient het tijdvak te kunnen 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 voor het object als geheel of voor een groep van elementen binnen een object, zie paragraaf 2.4.

Het is niet verplicht om een tijdvak geldigheid bij een entiteit of een groep van elementen binnen een entiteit op te nemen. Als het tijdvak geldigheid ontbreekt, bevat de entiteit of de groep actuele gegevens.

Een tijdvak geldigheid is van toepassing op alle in de entiteit of in de groep voorkomende attributen van een entiteittype. Het geldt niet voor gekoppelde relatie-entiteiten en de daarachter liggende entiteiten. In het voorbeeld in Figuur 1 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 entiteiten. 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 entiteit gespecificeerd te kunnen worden.

3.1.6 Robuuste berichtverwerking

Soms zal een ontvanger van een bericht entiteittypen of elementen binnen een entiteittype aantreffen die hij niet kent. Deze elementen worden bij de verwerking van het bericht genegeerd. Het is dus niet noodzakelijk dat zender en ontvanger met precies dezelfde implementatie van een sectormodel 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 is het dus niet altijd noodzakelijk bij alle gebruikers tegelijkertijd een nieuwe versie van de berichtverwerkende software te implementeren. Een berichtverwerker dient alle versies van een sectormodel te kunnen verwerken met hetzelfde majorversienummer (de eerste twee cijfers van het versienummer), voorzover deze versies werken met dezelfde versie van de StUF-standaard.

3.2 De syntax voor een object in een bericht

3.2.1 Metagegevens voor een fundamenteel, een relatie- en een tabelentiteittype

Bij een fundamenteel, een relatie- of een tabelentiteittype kunnen de volgende metagegevens als attributes in het bericht worden opgenomen:

- *Type entiteit*
  Het attribute `entiteittype` geeft aan wat het entiteittype is van het object. Dit attribute is verplicht op elk element voor een entiteittype uit het sectormodel.
- *Sleutel in het verzendende systeem*
  Het attribute `sleutelVerzendend` bevat de sleutel in het verzendende systeem. Het attribute `sleutelVerzendend` is optioneel behalve in kennisgevingberichten en asynchrone antwoordberichten (zie Tabel 3.1).
- *Sleutel in het ontvangende systeem*
  Het attribute `sleutelOntvangend` bevat de sleutel in het ontvangende systeem. Het attribute `sleutelOntvangend` is optioneel.
- *Sleutel in het gegevensbeheersysteem*
  Het attribute `sleutelGegevensbeheer` bevat de sleutel van een object in het gegevensbeheersysteem. Deze sleutel zorgt voor een systeemoverschrijdende identificatie van een object. Het attribute `sleutelGegevensbeheer` is optioneel.

Tabel 3.1 geeft een specificatie van deze attributes.

Naam	Type	Optioneel/Verplicht
`entiteittype`	String[30]	Verplicht
`sleutelVerzendend`	String[40]	Verplicht in kennisgevingberichten en asynchrone antwoordberichten, als in het sectormodel dit attribuut in de berichtdefinitie is opgenomen.[5] Aan deze verplichting hoeft niet voldaan te worden als het zendende systeem niet beschikt over de sleutel.
`sleutelOntvangend`	String[40]	Optioneel, als in het sectormodel dit attribuut in de berichtdefinitie is opgenomen.
`sleutelGegevensbeheer`	String[40]	Optioneel, als in het sectormodel dit attribuut in de berichtdefinitie is opgenomen.

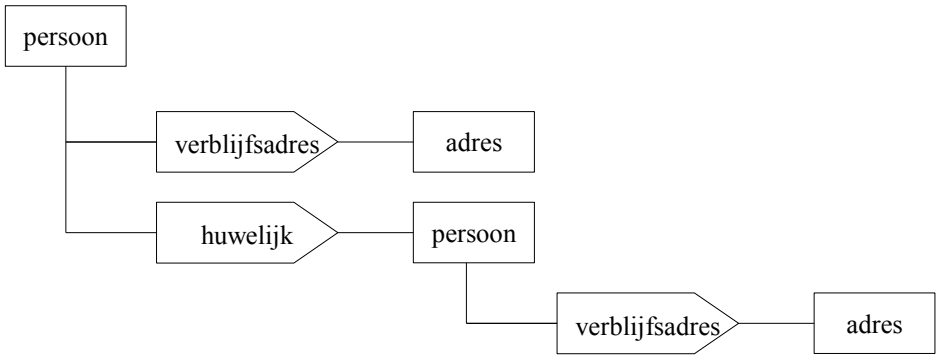
Tabel 3.1 Overzicht attributes voor een entiteitelement

In het generieke XML-schema (zie [StUFXSD]) voor StUF-berichten zijn de attributen van een fundamenteel, relatie- en tabelentiteittype met en zonder de attributes voor de sleutels gedefinieerd in de attribute groups `entiteit` en `entiteitZonderSleutels` (inclusief de in de volgende hoofdstukken nog te definiëren attributes).

[5] Deze sleutel is dan verplicht, omdat brokers dan objecten op eenvoudige wijze kunnen identificeren.

3.2.2 De structuur van een object

Figuur 1 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 Figuur 2 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, omdat het verblijfsadres een kerngegeven is. 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.



Figuur 2: Bericht over het huwelijk en de partner van een persoon

Figuur 2 geeft schematisch de opbouw van een berichtbody weer. Figuur 3 toont de corresponderende structuur in XML. Met elk blok in Figuur 2 correspondeert in Figuur 3 een begin- en een eindtag. Voor PRS bijvoorbeeld is de begintag `<persoon entiteittype=”PRS”>` en de eindtag `</persoon>` en voor PRSADR resp. `<verblijfsadres entiteittype=”PRSADR”>` en `</verblijfsadres>`. Tussen de begin- en eindtag staan alle gegevens die behoren tot het entiteittype. Zo begint het bericht met de tag `<persoon>` en eindigt het met `</persoon>`, want alle gegevens in het bericht behoren tot de persoon corresponderend met het blokje linksboven. De tags `<verblijfsadres>` en `</verblijfsadres>` omsluiten zowel de gegevens over de relatie tussen een persoon en zijn verblijfsadres als de gegevens over het adres zelf in het gereserveerde element `<gerelateerde>`. De tags `<gerelateerde>` en `</gerelateerde>` omsluiten alleen de adresgegevens. De figuur legt verder zichzelf uit. Bij het verblijfsadres is de enige functie van de relatie entiteit aan te geven in welk tijdvak de persoon op het adres verbleef. De relatie-entiteit `<huwelijk>` bevat na het element voor de gerelateerde nog gegevens over het huwelijk. De huwelijkspartner bevat zelf ook een relatie naar het verblijfsadres.

```
<persoon entiteittype=”PRS”>
    ...
    <verblijfsadres entiteittype=”PRSADRVBL”>
        <gerelateerde entiteittype=”ADR”>
            ...
        </gerelateerde>
        ...
    </verblijfsadres>
    <huwelijk entiteittype=”PRSPRSHUW”>
        <gerelateerde entiteittype=”PRS”>
            ...
            <verblijfsadres entiteittype=”PRSADRVBL”>
                    <gerelateerde entiteittype=”PRS”>
                            ...
                    </gerelateerde>
                    ...
            </verblijfsadres>
        </gerelateerde>
        ...
    </huwelijk>
</persoon>
```

Figuur 3: De opbouw van een bericht uit elementen

Een relatie-entiteit bevat altijd als eerste het gereserveerde element <gerelateerde>. Het element <gerelateerde> mag als kinderen bevatten de elementen van een fundamentele of tabelentiteit. Welke fundamentele of tabelentiteit dient gespecificeerd te worden in het attribute entiteittype: <gerelateerde entiteittype="XXX">, met XXX de code uit het sectormodel voor de fundamentele of tabelentiteit. Het element <gerelateerde> mag ook als kind een <choice> bevatten met twee of meer elementen met een attribute entiteittype en daarbinnen de elementen voor dat entiteittype. Het element <gerelateerde> heeft dan geen attributes. Deze constructie is van belang, als het entiteittype van de gerelateerde niet vaststaat. Een voorbeeld hiervan is de relatie naar de gebruiker van een verblijfsobject. Dit kan zowel een natuurlijk als een niet-natuurlijk persoon zijn.

De StUF-standaard schrijft niets voor over de volgorde van elementen voor attributen en relaties van een entiteittype. Het staat de ontwerper van een sectormodel dus vrij om eerst alle elementen voor de attributen op te nemen en daarna de elementen voor de relaties of elementen voor een relatie met ervoor en erna elementen voor attributen van het entiteittype. Relaties mogen ook in een groepselement worden opgenomen samen met elementen voor attributen van het entiteittype. De StUF-standaard schrijft niets voor om het gebruik van het extension- en restriction-mechanisme bij het definiëren van schema's zo min mogelijk te belemmeren.

In alle voorbeelden worden fundamentele en tabel entiteittypen aangeduid met drieletterige mnemonics en relatie entiteittypen met twee of drie drieletterige mnemonics, omdat de figuren en de tekst hiermee gemakkelijk leesbaar zijn. Bij het definiëren van sectormodellen mogen als elementnamen voor de entiteittypen lange en meer begrijpelijke namen worden gebruikt. In het onderstaande voorbeeld zijn voor de elementen langere elementnamen gebruikt.

3.2.3 Het opnemen van metagegevens ten behoeve van historie

Voor het specificeren van de historie zijn de volgende metagegevens relevant:

- *Begin van geldigheid*
  Dit metagegeven geeft aan vanaf welk tijdstip[6] 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 het begin van het 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. Het begin van 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. Het begin van geldigheid wordt opgenomen als het element <beginGeldigheid> binnen het element <tijdvakGeldigheid> met als type <TijdstipMetIndicator>. <tijdvakGeldigheid> is weer een element binnen een fundamentele of een relatie-entiteit. Het tijdstip wordt gecodeerd met minimaal 8 en maximaal 17 cijfers in het formaat EEJJMMDDhhmmssddd: eeuw (EE:00-99), jaar binnen eeuw (JJ:00-99), maand (MM:01-12), dag (DD:01-31), uur (hh:00-23), minuten (mm:00-59), seconden (ss:00-59) en één tot en met drie cijfers (d:0-9) voor de decimale nauwkeurigheid. Wanneer het tijdstip binnen de dag niet relevant of bekend is, dan wordt het tijdstip gecodeerd als datum (EEJJMMDD). Wanneer het tijdstip niet op de dag nauwkeurig bekend is, dan wordt de datum gevuld met een geldige datum en wordt een attribuut indOnvolledigeDatum bij het tijdvak-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'. Als de datum wel een waarde heeft, maar ook het jaar niet bekend is, dan krijgt dit attribuut de waarde 'J'. De default waarde voor dit attribuut is 'V', dat wil zeggen de datum is volledig. Als het tijdstip niet tot op het uur, de minuut, seconde, tiende van seconde, honderdste van seconde of duizendste van seconde nauwkeurig is, dan kunnen respectievelijk de laatste 9, 7, 5, 4, 3, 2 of 1 cijfer(s) uit de reeks EEJJMMDDhhmmssddd worden weggelaten.
- *Eind van geldigheid*
  Dit metagegeven geeft aan tot welk tijdstip 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 het begin tijdvak geldigheid tot het eind tijdvak geldigheid allemaal de gespecificeerde waarde hebben. Het eind 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. Het

[6] Met tijdstip wordt hier de combinatie van datum en tijd bedoeld. De exacte definitie van het begrip tijdstip komt verderop in de tekst aan de orde.

eindtijdstip van geldigheid wordt opgenomen als het element <eindGeldigheid> binnen het element <tijdvakGeldigheid> met als met als type <TijdstipMetIndicator>.

- *Begin van een relatie*
  Dit metagegeven komt alleen voor bij relatie-entiteiten en geeft aan op welk tijdstip de relatie ontstaan is. Begin en eind relatie worden als metagegeven gedefinieerd, omdat de standaard voor het ondersteunen van historische gegevens specificaties geeft voor het werken met deze twee metagegevens. Het *begin relatie* wordt opgenomen als het element <beginRelatie> binnen het element <tijdvakRelatie> met als type <TijdstipMetIndicator>.
- *Eind van een relatie*
  Dit metagegeven komt alleen voor bij relatie-entiteiten en geeft aan vanaf welk tijdstip de relatie niet meer bestaat (In geval van datum is het dus een tot-datum: de datum volgend op de dag waarop de relatie voor het laatst bestond). Het *eind relatie* wordt opgenomen als het element <eindRelatie> binnen het element <tijdvakRelatie met als met als type <TijdstipMetIndicator>.

Deze metagegevens kunnen niet worden opgenomen als attributes, omdat datums binnen StUF deels onbekend kunnen zijn en daarom een complexType hebben. Een attribute mag alleen een simpleType hebben. Het is aan de ontwerper van een sectormodel om de elementen <tijdvakGeldigheid> en <tijdvakRelatie> al dan niet voor een entiteittype te definiëren.

In het schema voor StUF-berichten (zie [StUFXSD]) zijn gedefinieerd het complexType voor het element <tijdvakGeldigheid> met de elementen <beginGeldigheid> en <eindGeldigheid> en het complexType voor het element <tijdvakRelatie> met de elementen <beginRelatie> en <eindRelatie>.

3.2.4 Samengestelde elementen in een entiteittype

De waarden van attributen van een object worden binnen een fundamentele entiteit, relatie-entiteit of tabelentiteit opgenomen als elementen met simpleContent. Relaties van een object worden binnen een fundamentele entiteit of relatie-entiteit opgenomen als elementen die weer elementen bevatten ofwel als elementen met complexContent.

De StUF-standaard staat toe dat binnen een entiteittype meerdere elementen voor de waarde van een attribuut en/of voor een relatie worden gegroepeerd tot een samengesteld element. Een samengesteld element mag dus zowel elementen voor de waarde van een attribuut als elementen voor een relatie bevatten. Samengestelde elementen mogen samen met andere samengestelde elementen en/of elementen voor de waarde van een attribuut en/of elementen voor een relatie weer in een samengesteld element worden gegroepeerd. De ontwerper van een sectormodel kan zelf specificeren of elementen binnen zo'n groep verplicht zijn of niet, bijvoorbeeld om een gegevensgroep rond huisnummer of centroïdcoördinaten te definiëren. Een samengesteld element mag alleen in een bericht worden opgenomen, als het minstens één element voor een waarde van een eigenschap of voor een relatie bevat.

Indien een samengesteld element uitsluitend elementen voor een waarde van een attribuut bevat en niet voorkomt binnen een samengesteld element dat ook een element voor een relatie bevat, dan mag in het samengestelde element ook een <tijdvakGeldigheid> worden opgenomen. Historie kan dan op het niveau van de groep gedefinieerd door het samengestelde element worden bijgehouden in plaats van op het niveau van het entiteittype. Zie voor meer details de specificaties voor het omgaan met historische gegevens in paragraaf 6.4.3.

Een <tijdvakGeldigheid> mag alleen binnen een entiteittype gedefinieerd worden, als het entiteittype geen samengestelde elementen bevat, waarbinnen zowel elementen voor de waarde van een eigenschap als elementen voor een relatie voorkomen. De reden hiervoor is dat met historische relaties op een andere manier wordt omgegaan als met historische attributen.

Voor samengestelde elementen definieert de StUF-standaard verder geen enkele functionaliteit. Door StUF gedefinieerde attributen voor de waarde van een eigenschap (StUF-element) of voor een relatie (StUF-relatie) mogen niet voorkomen op samengestelde elementen. De samengestelde elementen zijn slechts een middel om elementen met te groeperen. Voorschriften in de StUF-standaard voor elementen binnen een fundamentele entiteit, een relatie-entiteit of een tabelentiteit hebben altijd uitsluitend betrekking op de elementen op het laagste niveau in een samenstelling.

3.2.5 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 is het element `<extraElement>` en het complexType `<ExtraElementen>` gedefinieerd. Hieronder staan de definities uit het StUF-schema [StUFXSD].

```
<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>

<complexType name="ExtraElementen">
   <sequence maxOccurs="unbounded">
      <element ref="StUF:extraElement" minOccurs="0"/>
   </sequence>
</complexType>
```

Een `<extraElement>` wordt dus gedefinieerd als een element met simpleContent van het type string, waaraan het attribute elementnaam is toegevoegd. De elementnamen voor extra elementen worden buiten het sectormodel om gedefinieerd. Een uitgegeven elementnaam wordt geregistreerd samen met de aanvrager en desgewenst een definitie of omschrijving. Een dergelijk elementnaam mag niet een tweede keer worden uitgegeven.

3.3 Het opnemen van elementen en relatie-entiteiten in een entiteit

Of een attribuut of een relatie van een object gedefinieerd in het sectormodel als element in een entiteit wordt opgenomen en zo ja hoe, is afhankelijk van een aantal factoren. Deze paragraaf gaat daar dieper op in.

3.3.1 Het opnemen van elementen in een entiteit

Er zijn redenen waarom van een element niet altijd met een geldige waarde in een bericht kan worden opgenomen. Deze redenen worden onderscheiden met het attribute `StUF:noValue` en worden besproken aan de hand van de beslisboom in Figuur 4.

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 attribute `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 attribute `StUF:noValue="nietOndersteund"` (a) en een lege elementinhoud[7]. 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).

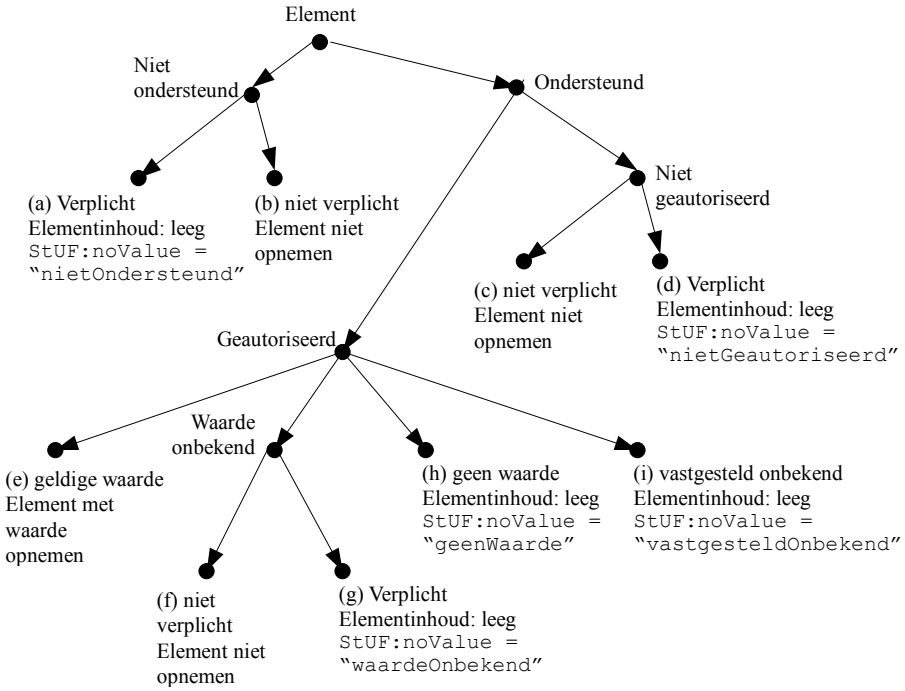
Wanneer het element wel wordt ondersteund, wordt er gekeken of de ontvanger van het bericht geautoriseerd is voor dit element. Als het element niet verplicht is, dan wordt het niet opgenomen in het bericht (c). Als het verplicht is, dan wordt het niet-geautoriseerd zijn expliciet gemeld door het element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

Indien de ontvanger van het bericht is geautoriseerd om het element te ontvangen, zijn er vier mogelijkheden voor de waarde van het element:

[7] Als een lege elementinhoud niet is toegestaan vanuit het contentmodel in het schema, dan dient het attribute `xsi:nil="true"` op het element te worden opgenomen. In het schema dient voor het element dan `nillable="true"` te zijn gespecificeerd.

1. Element heeft geldige waarde
2. Het element heeft een waarde, maar de waarde is bij de zender niet bekend
3. Er is vastgesteld dat de waarde van het element onbekend is
4. Element heeft geen waarde

De derde mogelijkheid is een bijzondere vorm van het onbekend zijn van een waarde. In dat geval is vastgesteld dat de waarde 'echt' onbekend is, bijvoorbeeld omdat het brondocument voor de waarde onleesbaar is geworden of omdat het vaststellen van de waarde onmogelijk is geworden, doordat het object niet meer bestaat.



Figuur 4: Het opnemen van een element in een bericht

Als het element in de werkelijkheid een geldige waarde heeft, wordt het element met zijn waarde als inhoud opgenomen (e). Als er wel een waarde is, maar deze is bij de zender onbekend, dan wordt het element, als het niet verplicht is niet opgenomen (f) en als het wel verplicht opgenomen met een lege inhoud en StUF:noValue="waardeOnbekend" (g). Een kerngegeven als de geboortedatum van een persoon waarvan het zendende systeem niet altijd de waarde kent, wordt bijvoorbeeld met StUF:noValue="waardeOnbekend" opgenomen. Als het element geen waarde heeft, bijvoorbeeld de overlijdensdatum van een niet overleden persoon, dan wordt het element in het bericht opgenomen voorzien met een lege inhoud en StUF:noValue="geenWaarde" (h). Als vastgesteld is dat de waarde onbekend is, dan wordt het element opgenomen met een lege elementinhoud en StUF:noValue="vastgesteldOnbekend" (i).

In datumvelden heeft `StUF:noValue="geenWaarde"` als betekenis dat de datum nog geen waarde heeft. Dit kan alleen voorkomen bij datums die niet altijd een waarde hoeven te hebben (optionaliteit 0). Bij de overlijdensdatum van een persoon wil dit bijvoorbeeld zeggen dat de persoon nog niet is overleden. Bij een `<eindRelatie>` wil dit zeggen dat de relatie nog bestaat en bij een `<eindGeldigheid>` wil dit zeggen dat de gegevens heden geldig zijn.

In datumvelden heeft `StUF:noValue="waardeOnbekend"` als betekenis dat de datum onbekend is. Bij een datum die niet altijd een waarde hoeft te hebben zoals een overlijdensdatum wil dit zeggen dat niet bekend is of de persoon al dan niet is overleden. Bij een `<eindRelatie>` wil dit zeggen dat niet bekend is of de relatie al dan niet beëindigd is. Als in een `<eindGeldigheid>` `StUF:noValue="waardeOnbekend"` staat, dan wil dit zeggen dat niet bekend is of de gegevens heden geldig zijn. Wanneer zeker is dat een persoon is overleden, maar onbekend is wanneer, dan kan de `indVolledigeDatum` op `'J'` gezet worden. Hetzelfde geldt voor de elementen `<eindGeldigheid>` en `<eindRelatie>`.

3.3.2 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 attribute `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 attribute `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 attribute `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 het element `<gerelateerde>` voor het gerelateerde entiteittype gevuld met elementen van de gerelateerde.
- *Het verzendende weet dat vastgsteld is dat het onbekend is of de relatie al dan niet voorkomt*
  Het element voor de relatie-entiteit wordt opgenomen zonder elementinhoud en met het attribute `StUF:noValue="vastgesteldOnbekend"`.
- *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 attribute `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.

4. Berichtverwerking: sturing, logistiek en foutafhandeling

Bij het versturen van een brief wordt al honderden jaren onderscheid gemaakt tussen de envelop relevant voor het transport en de inhoud bestemd voor de geadresseerde. Dit onderscheid wordt ook gemaakt bij de verwerking van digitale berichten. SOAP hanteert bijvoorbeeld het begrip 'envelope' voor het complete bericht. De SOAP-envelope bevat een of meer header-elementen met gegevens ten behoeve van het transport en een body-element bedoeld voor de uiteindelijk geadresseerde.

De StUF-standaard maakt een soortgelijk onderscheid. Een StUF-bericht bevat altijd een element `<stuurgegevens>` met een aantal gegevens ten behoeve van het transport, de berichtenlogistiek en het op hoog niveau aangeven om wat voor soort bericht het gaat. Na het element `<stuurgegevens>` volgt in de meeste berichten het element `<body>` met de 'inhoud' van het bericht. De algemene structuur van een StUF-bericht is dus:

```
<berichtnaam>
     <stuurgegevens>
          ...
     </stuurgegevens>
     <body>
          ...
     </body>
</berichtnaam>
```

met berichtnaam nog vrij te kiezen.

De structuur van een StUF-bericht lijkt slechts op die van een SOAP-envelope. StUF wijkt bewust af van SOAP, want het wil een protocolonafhankelijke standaard zijn. In enkele protocolbindingen maakt StUF gebruikt van SOAP. In het hoofdstuk over protocolbindingen zal worden uitgelegd hoe een StUF-bericht wordt getransporteerd binnen een SOAP-envelope.

Dit hoofdstuk bespreekt de functionaliteit gedefinieerd voor het element `<stuurgegevens>`. Denk hierbij aan het coderen van de verschillende berichttypen, de adressering (zender en ontvanger), identificatie van berichten en de volgorde van verwerking. Het al dan niet verplicht zijn van de elementen binnen de stuurgegevens wordt hier niet besproken, omdat dit varieert per type bericht. De exacte definitie qua formaat is te vinden in het StUF-schema.

Dit hoofdstuk beschrijft ook hoe gereageerd dient te worden op de ontvangst van een bericht. Aan de orde komen de synchrone respons op een asynchroon bericht en de door de StUF-standaard gedefinieerde berichtsoorten bevestingsbericht, triggerbericht en foutbericht. Daarnaast wordt ook de voor alle berichtsoorten geldende foutafhandeling behandeld.

4.1 Codering van het type bericht

4.1.1 Versie StUF en sectormodel

De StUF-standaard ontwikkelt zich in de loop van de tijd en kent daarom verschillende versies. Het stuurgegeven *versieStUF* geeft aan op basis van welke versie van StUF een bericht is aangemaakt.

Met StUF kunnen berichten worden uitgewisseld voor verschillende sectoren die elk een eigen sectormodel hanteren. Een ontvanger moet dus weten op basis van welk sectormodel een bericht is aangemaakt. Ook van sectormodellen kunnen er in de loop van de tijd verschillende versies bestaan. De stuurgegevens *sectormodel* en *versieSectormodel* geven daarom aan uit welk sectormodel en welke versie daarvan een bericht afkomstig is. Een sociale dienst systeem zal in het kader van de binnengemeentelijke gegevensuitwisseling bijvoorbeeld werken met het sectormodel ‘binnen-gemeentelijk’ en voor de uitwisseling binnen de sociale sector met het sectormodel ‘sociale zekerheid’.

4.1.2 Berichtcode

In hoofdstuk 2 is een aantal verschillende berichtsoorten beschreven die de StUF-standaard ondersteunt. De StUF-standaard codeert dit in het stuurgegeven *berichtcode*. Het gebruik en de betekenis van de verschillende berichtcodes wordt verderop in de standaard beschreven. Onderstaande lijst geeft alvast de waarden van het stuurgegeven *berichtcode* en de betekenis van het bericht:

- Bv01: een bevestigingsbericht als functionele asynchrone respons

- Bv02: een bevestigingsbericht als functionele synchrone respons op een synchroon bericht
- Bv03: een bevestigingsbericht als technische synchrone respons op een asynchroon bericht waarbij het bericht op basis van berichtstuurgegevens verwerkbaar wordt geacht
- Bv04: een bevestigingsbericht als technische synchrone respons op een asynchroon bericht, dat een check op verwerkbaarheid op basis van de berichtstuurgegevens ontkent
- Bv05: een bevestigingsbericht op een triggerbericht dat binnen 5 minuten gestart zal worden met het zenden van de klaarstaande berichten.
- Di01: een asynchroon inkomend vrij bericht
- Di02: een synchroon inkomend vrij bericht
- Du01: een asynchroon uitgaand vrij bericht (respons op een Di01)
- Du02: een synchroon uitgaand vrij bericht (respons op een Di02)
- Fo01: een foutbericht als functionele asynchrone respons
- Fo02: een foutbericht als functionele synchrone respons
- Fo03: een foutbericht als technische synchrone respons op een asynchroon bericht
- Fo04: een foutbericht op een triggerbericht dat het ontvangende systeem niet in staat is binnen 5 minuten te starten met het zenden van de berichten die klaarstaan voor de zender van het triggerbericht
- La01: een synchroon antwoordbericht
- La02: een asynchroon antwoordbericht
- Lk01: een asynchroon kennisgevingbericht
- Lk02: een synchroon kennisgevingbericht
- Lk03: een asynchroon samengesteld kennisgevingbericht
- Lk04: een synchroon samengesteld kennisgevingbericht
- Lv01: een synchroon vraagbericht
- Lv02: een asynchroon vraagbericht
- Sa01: een asynchroon synchronisatiebericht voor de actuele gegevens
- Sa02: een synchroon synchronisatiebericht voor de actuele gegevens
- Sh01: een asynchroon synchronisatiebericht voor de actuele en de historische gegevens
- Sh02: een synchroon synchronisatiebericht voor de actuele en de historische gegevens
- Tr01: een triggerbericht

Het aantal berichtcodes is groter dan het aantal berichtsoorten uit hoofdstuk 2, omdat sommige berichten een synchrone en een asynchrone variant hebben.

4.1.3 Entiteittype en functie

Een enkelvoudig kennisgevingbericht, vraag/antwoord berichten en een synchronisatiebericht hebben altijd betrekking op objecten van één entiteittype. Dat entiteittype wordt meegegeven in het stuurgegeven *entiteittype.*

Het sectormodel definieert voor welke entiteittypen er berichten zijn met een bepaalde *berichtcode*. Vrije berichten worden veelal gedefinieerd ten behoeve van een bepaalde functie, bijvoorbeeld het teruggeven van taxatiegegevens voor een WOZ-object of het doorgeven van een gebeurtenis. De functie van het vrije bericht wordt meegegeven in het stuurgegeven *functie*. Het sectormodel definieert de verschillende vrije berichten met hun functie. Een vrij bericht kan betrekking hebben op objecten van één entiteittype. In dat geval wordt dat entiteittype meegegeven in het stuurgegeven *entiteittype*.

4.2 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 o*ntvanger* gedefinieerd. Deze twee stuurgegevens maken gebruik van een gemeenschappelijke adrestype-definitie. Hieronder worden de verschillende onderdelen van het adrestype gespecificeerd.

StUF-berichten worden uitgewisseld tussen geautomatiseerde systemen. Zo'n geautomatiseerd systeem wordt beheerd door een organisatie. Het hoogste niveau in de adrestype is daarom de *organisatie*. Een organisatie heeft over het algemeen een groot aantal verschillende applicaties en 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 zo-

wel 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 stuurgegevens: de *applicatie* en de *administratie*. Met het stuurgegeven *administratie* kan onderscheid worden gemaakt tussen de verschillende gegevensverzamelingen die een applicatie beheert.

StUF biedt de mogelijkheid om berichten te adresseren op het niveau van individuele gebruikers met behulp van het stuurgegeven *gebruiker*. Dit stuurgegeven kan ook gebruikt worden ten behoeve van autorisatie, bijvoorbeeld als een vraagbericht alleen beantwoord mag 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

4.3 Identificatie en volgorde

4.3.1 Identificatie van berichten

Berichten worden geïdentificeerd met een *referentienummer*. StUF schrijft niet voor hoe het referentienummer opgebouwd moet worden. Ook alle berichten die een reactie zijn op een ander bericht (bevestigingsberichten, foutberichten, antwoordberichten en uitgaande vrije berichten) krijgen een eigen referentienummer van het systeem dat het bericht aanmaakt.

Berichten die onafhankelijk van elkaar zijn aangemaakt door verschillende systemen kunnen toevallig hetzelfde referentienummer hebben, omdat StUF geen voorschriften geeft voor de opbouw van het referentienummer. 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.

Voor berichten die een reactie zijn op een ander bericht, is het noodzakelijk te weten op welk bericht wordt gereageerd. Hiervoor wordt in deze berichten het stuurgegeven *crossRefnummer* opgenomen. Het *crossRefnummer* wordt gevuld met de waarde van het referentienummer van het bericht waarop wordt gereageerd.

4.3.2 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 *tijdstipBericht* gedefinieerd waarin tot op een duizendste seconde nauwkeurig het tijdstip van de aanmaak van het bericht kan worden gespecificeerd. Het tijdstip dient minimaal op het niveau van een datum te worden gespecificeerd. Het staat een verzendend systeem vrij te bepalen hoe nauwkeurig het tijdstip binnen de dag wordt opgegeven. Een systeem dat bijvoorbeeld dagelijks één bericht verzendt, zou ervoor kunnen kiezen om het tijdstip te coderen als de datum (EEJJMMDD). StUF stelt wel als randvoorwaarde dat het *tijdstipBericht* van een bericht groter is dan het *tijdstipBericht* van alle eerder door een systeem verzonden berichten.

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

4.4 Berichtenlogistiek en foutafhandeling

Binnen berichtenverkeer worden de varianten synchroon en asynchroon onderscheiden. Synchroon verkeer wil zeggen dat de respons over dezelfde verbinding wordt gegeven als waarover het verzoek gedaan is. Asynchroon wil

zeggen dat de respons over een andere meestal nieuw opgezette verbinding wordt gegeven. Het voordeel van synchroon verkeer is dat de verzoekende partij kan wachten op de respons op de verbinding waarover het verzoek gedaan is. Als de respons er binnen een zekere time-out tijd is, dan is het verzoek geslaagd en anders faalt het verzoek. Synchroon berichtenverkeer stelt dus hoge eisen aan de aanbieder van een service. Service-aanbieders waarbij de belasting van de service sterk kan variëren, geven daarom vaak de voorkeur aan asynchroon berichtenverkeer, want dan kunnen zij conform de eigen capaciteit de binnenkomende berichten verwerken. Asynchroon berichtenverkeer is dus robuuster, maar heeft als nadeel dat de serviceverzoeker herhaaldelijk zal moeten checken of er inmiddels een antwoord is ontvangen (pollen). De service-aanbieder verschuift dus een deel van zijn probleem naar de serviceverzoeker.

Op functioneel niveau kent de StUF-standaard synchroon en asynchroon berichtenverkeer. De respons op een asynchroon StUF-verzoek wordt over een nieuw opgezette verbinding naar de zender van het verzoek gestuurd, Hierbij wisselen zender en ontvanger van rol. De zender van het StUF-verzoek ontvangt een asynchroon bericht met daarin de respons. Aan de hand van het stuurgegeven *crossRefnummer* kan de zender van het verzoek bepalen bij welke verzoek een respons hoort.

StUF wil net zoals SOAP een communicatiemodel kunnen ondersteunen met eventueel intermediaire nodes, via welke het bericht wordt doorgegeven naar de end node die het verwerkt. Een dergelijke end node zullen we in het vervolg een StUF end node noemen. Intermediaire nodes hoeven het bericht niet inhoudelijk te interpreteren. Ze zijn slechts verantwoordelijk voor het transport. Voor synchrone StUF-berichten zullen intermediaire nodes in de praktijk niet voorkomen, omdat de ontvanger synchroon dient te antwoorden. Het is lastig een synchroon proces te implementeren met één of meer intermediaire nodes. Deze versie van de standaard gaat ervan uit dat synchrone StUF-berichten altijd direct verzonden worden naar de end node die het bericht verwerkt.

Bij asynchroon berichtenverkeer worden geregeld intermediaire nodes gebruikt, denk aan een broker die zorgt voor de routering. Voorgaande versies van de StUF-standaard schrijven voor dat de ontvangst van een asynchroon bericht wordt bevestigd via een respons over de verbinding waarover het asynchrone bericht aan de ontvanger is aangeboden. Technisch is er dus ook bij asynchroon berichtenverkeer sprake van een synchrone respons. Het grote voordeel van deze werkwijze is dat de zender van een asynchroon bericht onmiddellijk weet of de verzending geslaagd is of dat hij op een later tijdstip opnieuw het bericht moet aanbieden aan de ontvanger. Deze werkwijze wijkt af van de voorstellen binnen de standaarden WS-ReliableMessaging en WS-Reliability. De WS-Reliability standaard schrijft bijvoorbeeld niet voor dat de ontvanger van een reliable message onmiddellijk een bevestiging van ontvangst zendt.

De StUF-standaard biedt in de stuurgegevens voorzieningen, zodat een StUF end node zelf eventuele dubbelen kan elimineren. Daarnaast stelt de StUF-standaard niet als eis dat onafhankelijke berichten per se in de volgorde van verzending verwerkt moeten worden. Het heeft uiteraard wel de voorkeur om ze met behulp van het stuurgegeven *tijdstipBericht* in de volgorde van verzending te verwerken, omdat dit de kans op foutsituaties verkleint. Als een respons bestaat uit een groep berichten (asynchrone antwoordberichten) biedt de StUF-standaard voorzieningen om na te gaan of er onderweg berichten verloren zijn gegaan. StUF heeft voor deze uitgangspunten gekozen om met zo min mogelijk inspanningen van de implementerende systeem een voldoende mate van betrouwbaarheid te bieden. Dankzij de StUF-voorschriften is de berichtverzending state-less.

In deze versie van de standaard wordt voor asynchrone berichten de bestaande werkwijze gehandhaafd en uitgebreid met foutafhandeling op het niveau van de berichtstuurgegevens bij de ontvangst van een asynchroon bericht door een StUF end node. Tevens wil deze versie van de StUF-standaard het werken met intermediaire nodes die geen kennis hebben van StUF-berichten niet onmogelijk maken. Van een intermediaire node kan niet verwacht worden dat deze controleert of de stuurgegevens aan zekere eisen voldoen. Een intermediaire node kan slechts een ontvangstbevestiging sturen. Voor asynchrone berichten zijn dus twee verschillende synchrone ontvangstbevestigingen nodig:

1. Een bevestiging dat het bericht is ontvangen en dat het op het niveau van de berichtstuurgegevens een eerste check op verwerkbaarheid heeft doorstaan. Zo'n technische synchrone respons is een bevestigingsbericht met berichtcode Bv03. Als er fouten worden geconstateerd, dan wordt als synchrone respons een foutbericht met berichtcode Fo03 gestuurd.
2. Een bevestiging dat het bericht is ontvangen en gegarandeerd zal worden afgeleverd bij de StUF end node, maar dat er op het niveau van de berichtstuurgegevens geen controle is uitgevoerd. De StUF-standaard eist dus dat een intermediaire node die verder geen kennis heeft van StUF in staat is een door de StUF-standaard gedefinieerd

bericht als bevestiging terug te zenden op een inkomend asynchroon StUF-bericht. Deze technisch synchrone respons is een bevestigingsbericht met berichtcode Bv04.

Intermediaire nodes dienen eventuele Fo03-foutberichten van de StUF end node als asynchrone berichten aan te bieden aan de oorspronkelijke verzender van het StUF-bericht. Een Fo03-foutbericht kan dus zowel synchroon als asynchroon ontvangen worden. Synchroon bij het direct aanbieden van een asynchroon StUF-bericht aan een StUF end node en asynchroon na de ontvangst van een synchroon Bv04-bevestigingsbericht bij het verzenden van een asynchroon StUF-bericht via één of meer intermediaire nodes. In het hoofstuk over de protocolbindingen wordt beschreven hoe deze werkwijze voor de verschillende protocollen is geïmplementeerd.

Als het Bv03-bevestigingsbericht, het Fo03-foutbericht of het Bv04-bevestigingsbericht als respons op een asynchroon bericht niet op tijd is ontvangen, dan dient de zender ervan uit te gaan dat het asynchrone bericht niet ontvangen is en het op een later tijdstip opnieuw te verzenden. Op tijd kan het beste gedefinieerd worden aan de hand van de connection time out en bijbehorende foutafhandeling van het onderliggende transportprotocol. Http kent bijvoorbeeld zo'n connection time out en bijbehorende foutafhandeling. Als het onderliggende transportprotocol geen connection time out kent, dan dient in de protocolbinding het begrip op tijd en de bijbehorende foutafhandeling te worden gedefinieerd.

Een StUF end node voor asynchrone berichten mag niet met een foutbericht reageren, indien een reeds eerder ontvangen bericht opnieuw wordt aangeboden. Dit zal gebeuren, als het Bv03-bevestigingsbericht of het Fo03-foutbericht verloren is gegaan. Het opnieuw aangeboden bericht heeft dan het referentienummer van een al eerder opgeslagen bericht. Zodra een referentienummer niet uniek is, dient de ontvanger daarom te checken of het bericht met het dubbele referentienummer identiek is aan het eerder ontvangen bericht met hetzelfde referentienummer. Zo ja, dan wordt een Bv03-bevestigingsbericht gestuurd en wordt het nieuw ontvangen bericht niet opgeslagen. Zo nee, dan wordt het Fo03-foutbericht met code StUF016 voor een dubbel referentienummer gestuurd (zie paragraaf 4.4.3).

Ook een intermdiaire node dient met Bv04-bevestigingsbericht te reageren, indien een bericht voor de tweede keer wordt aangeboden. Het is de verantwoordelijkheid van de StUF end node om eventuele dubbele berichten te elimineren.

Indien een synchroon verzoekbericht niet op tijd (zie hierboven) kan worden verwerkt, dan mag het antwoordende systeem gebruik maken van de foutafhandeling op het niveau van het onderliggende transportprotocol om aan te geven dat het niet in staat is aan het verzoek te voldoen. Het heeft echter de voorkeur dat er in een dergelijk geval wordt gereageerd met een Fo02-foutbericht met code StUF960 (zie paragraaf 4.4.3). Het vragende systeem weet dan dat fout niet ligt op het niveau van het transportprotocol, maar functioneel binnen het antwoordende systeem.

4.4.1 Regels voor bevestigingsberichten

Afhankelijk van het type bevestigingsbericht wordt in de stuurgegevens `<berichtcode>` gevuld met Bv01, Bv02 of Bv03. De berichtcodes zijn gedefinieerd in paragraaf 4.1.2. Het gebruik van de verschillende berichtcodes wordt hieronder nader toegelicht. De elementen `<sectormodel>`, `<versieStUF>`, `<versieSectorModel>`, `<zender>` en `<ontvanger>` in de stuurgegevens worden gevuld op basis van de waarden in het bericht naar aanleiding waarvan het bevestigingsbericht wordt aangemaakt. De elementen `<referentienummer>` en `<tijdstipBericht>` worden gevuld conform de regels in paragraaf 4.3.1 en 4.3.2. Het `<crossRefNummer>` wordt gevuld met het referentienummer van het bericht naar aanleiding waarvan het bevestigingsbericht wordt aangemaakt.

Er zijn twee voorwaarden voor het mogen verzenden door een StUF end node van een Bv03-bevestigingsbericht als respons op een asynchroon verzoek:

1. De ontvanger heeft zeker gesteld dat het bericht veilig is opgeslagen. Veilig wil zeggen dat de zender er na de ontvangst van het bevestigingsbericht vanuit mag gaan dat het bericht bij de ontvanger niet verloren zal gaan, ook niet in geval van calamiteiten als systeemuitval ten gevolge van een softwarefout, stroomuitval en dergelijke. De StUF-standaard schrijft niet voor dat het bericht ook nog terug te vinden moet zijn in geval van het verloren gaan van het medium waarop het bericht is opgeslagen.
2. De ontvanger heeft op basis van de stuurgegevens vastgesteld dat het bericht verwerkt kan worden. De uit te voeren controles staan in de vorm van foutsituaties gedefinieerd in Tabel 4.1 voor soort fout 3. De StUF-standaard schrijft niet voor dat de ontvanger ook moet checken dat het bericht aan het schema voldoet of moet kijken of het bericht gegeven de body van het bericht verwerkbaar is. Dit is de verantwoordelijkheid van de zender.

Als niet aan beide voorwaarden wordt voldaan, dan dient als synchrone respons op een asynchroon bericht een Fo03-foutbericht gestuurd te worden, voor meer details zie hieronder.

Als aan de volgende voorwaarde is voldaan mag een intermediaire node als respons op een asynchroon StUF-verzoek een Bv04-bevestigingsbericht verzenden: De intermediaire node garandeert dat het bericht zal worden afgeleverd bij de StUF end node waarvoor het bestemd is.

Bevestigingsberichten kunnen in StUF ook op functioneel niveau gebruikt worden als respons: de bevestigingsberichten met de berichtcodes Bv01 en Bv02. In een functioneel bevestigingsbericht kan in het optionele element `<melding>` met kardinaliteit ∞ in de body worden aangegeven dat het bericht verwerkt is, maar niet geheel conform de verwachting van de serviceverzoeker. Voor enkele situaties is de inhoud van dit element gedefinieerd in de StUF-standaard. Daarnaast kan de inhoud van dit element in het sectormodel worden gedefinieerd. Het `<melding>` element is gedefinieerd in het StUF-schema ([StUFXSD]).

De StUF-standaard definieert bijvoorbeeld als respons op synchrone kennisgevingen en synchronisatieberichten een Bv02-bevestigingsbericht. Bij een asynchroon vrij bericht kan een Bv01-bevestigingsbericht gebruikt worden om aan te geven dat de verwerking geslaagd is. Wanneer een Bv01-bevestigingsbericht functioneel de respons is op een asynchroon bericht wordt eerst synchroon een Bv03-bevestigingsbericht verstuurd om aan te geven dat het bericht in goede orde ontvangen is en vervolgens asynchroon een Bv01-bevestigingsbericht om aan te geven dat de verwerking van het bericht functioneel geslaagd is.

4.4.2 Regels voor triggerbericht

In de stuurgegevens van het triggerbericht mogen de elementen `<sectormodel>` en `<versieSectormodel>` niet voorkomen. Als ze wel gevuld zijn, dan heeft dit functioneel geen betekenis. Het triggerbericht dient verwerkt te worden alsof ze leeg waren. Als de ontvanger van het triggerbericht verwacht binnen 5 minuten te kunnen starten met de verzending van de voor de zender van het triggerbericht klaarstaande berichten, dan stuurt het ontvangende systeem als synchroon antwoord een bevestigingsbericht met als berichtcode Bv05 en zonder de elementen `<sectormodel>` en `<versieSectormodel>` in de stuurgegevens en zonder `<melding>` in de body. Als de ontvanger de verzending niet kan starten, dan wordt een foutbericht met berichtcode Fo04 verstuurd zonder de elementen `<sectormodel>` en `<versieSectormodel>` in de stuurgegevens en met als code StUF061 (zie hieronder).

Alle voor de zender van het triggerbericht klaarstaande dienen verstuurd te worden ongeacht het sectormodel en de versie. De verzending van de berichten stopt, zodra er niet binnen de in de protocolbinding voorgeschreven connection time out een Bv03-bevestigingsbericht of een Fo03-foutbericht is ontvangen. De verzending van de berichten stopt ook, als 5x na elkaar als respons een Fo03-foutbericht wordt ontvangen of als er meer dan 25 Fo03-foutberichten zijn ontvangen. Eventueel tijdens het verzendproces nog aangemaakte berichten worden ook verzonden. Normaal stopt de verzending dus pas, als er geen klaarstaande berichten meer zijn.

4.4.3 Regels voor foutberichten

Op een synchroon bericht wordt gereageerd met een Fo02-foutbericht of de in StUF-standaard of het sectomodel gedefineerde respons. Voor synchrone kennisgevingberichten, synchronisatieberichten en vraag/antwoord berichten beschrijft de StUF-standaard de respons en de foutafhandeling. Ook voor asynchrone vraagberichten beschrijft de StUF-standaard de respons en de foutafhandeling. Voor synchrone en asynchrone vrije berichten dient de functionele foutafhandeling door middel van een Fo02- cq Fo01-bericht in het sectormodel gedefinieerd te worden. Als de veilige opslag of de verwerking van een asynchroon bericht niet mogelijk is, dan wordt technisch synchroon gereageerd met een Fo03-foutbericht.

De elementen `<sectormodel>`, `<versieStUF>`, `<versieSectorModel>`, `<zender>` en `<ontvanger>` in de stuurgegevens van een foutbericht 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 4.3.1 en 4.3.2. Het `<crossRefNummer>` wordt gevuld met het referentienummer van het bericht naar aanleiding waarvan het foutbericht wordt aangemaakt.

De Fo01-, Fo02 -en Fo03-foutberichten hebben een body met als elementen `<code>`, `<plek>`, `<omschrijving>` en `<details>`. 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. De details over de fout kunnen worden opgenomen in het vrij in te vullen <details> element.

Tabel 4.1 geeft de onderkende waarden voor <code>, <plek>, <omschrijving> en <details> in foutberichten geldig voor alle berichttypen. De kolom Soort fout geeft aan voor welke foutberichten de fout van toepassing is (1 = Fo01, 2 = Fo02 en 3 = Fo03). Foutafhandeling specifiek voor een bepaalde berichttype wordt beschreven in het hoofdstuk over dat betreffende berichttype. Ook voor vrije berichten is de hieronder beschreven foutafhandeling onverkort van toepassing. Additionele foutafhandeling voor vrije berichten kan worden gedefinieerd in het sectormodel. In een sectormodel kunnen extra combinaties van <code>, <plek>, <omschrijving> en <details> voor Fo01- en Fo02-foutberichten worden gedefinieerd. De in een sectormodel gedefinieerde waarden voor <code> dienen te starten met de code voor het sectormodel.

Als er meerdere foutsituaties van toepassing zijn, dan wordt uitsluitend het foutbericht voor de eerst in de tabel voorkomende situatie verzonden. Dit geldt ook voor de in de rest van dit document nog gedefinieerde foutsituaties. De foutsituaties gedefinieerd in een sectormodel gaan voor foutsituaties gedefinieerd binnen de StUF-standaard.

Foutsituatie (<omschrijving>)	**Soort fout**	**<code>**	**<plek>**	**<details>**
Versie StUF niet ondersteund	2,3	StUF001	Server	De dichtstbijzijnde[8] wel ondersteunde versie StUF.
Sectormodel niet ondersteund	2,3	StUF004	Server	-
Versie sectormodel niet ondersteund	2,3	StUF007	Server	De dichtstbijzijnde wel ondersteunde versie sectormodel
Combinatie van ontvangende organisatie, applicatie en administratie onbekend	2,3	StUF010	Client	
Combinatie van zendende organisatie, applicatie en administratie onbekend	2,3	StUF013	Client	
Combinatie zender en referentienummer niet uniek	2,3	StUF016	Client	
TijdstipBericht niet groter dan voorgaand TijdstipBericht van zender	2,3	StUF019	Client	
Berichtcode onbekend	2,3	StUF022	Client	
Berichtcode niet ondersteund	2,3	StUF025	Server	
Entiteittype onbekend binnen sectormodel	2,3	StUF028	Client	
Entiteittype niet ondersteund	2,3	StUF031	Server	
Functie onbekend binnen sectomodel	2,3	StUF034	Client	
Functie niet ondersteund	2,3	StUF037	Server	
Combinatie van berichtcode, entiteittype en functie niet ondersteund	2,3	StUF040	Server	
Crossreferentienummer niet bekend	2,3	StUF043	Client	
Opslaan bericht niet mogelijk	3	StUF046	Server	
Proces voor afhandelen synchroon bericht niet beschikbaar	2	StUF049	Server	
Het zendende systeem is niet geautoriseerd	1,2	StUF052	Client	

[8]Onder dichtstbijzijnde wordt hier verstaan de laagst ondersteunde versie, als de versie in het bericht lager is dan de ondersteunde versie en de hoogst ondersteunde versie, als de versie in het bericht hoger is dan de ondersteunde versie.

Foutsituatie (<omschrijving>)	**Soort fout**	**<code>**	**<plek>**	**<details>**
voor de gevraagde combinatie van berichtcode, entiteittype en functie				
Berichtbody is niet conform schema in sectormodel	1,2	StUF055	Client	
Proces voor afhandelen bericht geeft fout	1,2	StUF058	Server	Desgewenst een omschrijving van de fout in het afhandelende proces
Starten berichtverzending niet mogelijk binnen 5 minuten	2	StUF061	Server	
Het antwoordende systeem is niet in staat het verzoek af te handelen binnen de connection time out	2	StUF960	Server	

Tabel 4.1: Onderkende foutsituaties met hun <code>, <plek>, <omschrijving> en <details>

De foutsituaties StUF052 en StUF058 mogen desgewenst vervangen worden door een in het sectormodel gedefinieerde fout.

Alle foutsituaties waarbij in de kolom 'Soort fout' een 3 staat, dienen bij de ontvangst van een asynchroon bericht gecheckt te worden, voordat het Bv03-bericht wordt verstuurd. Als niet aan de check wordt voldaan, dan dient een Fo03-foutbericht te worden teruggestuurd. De fout met code StUF043 specificeert bijvoorbeeld dat voor het zenden van het Bv03-bevestigingsbericht gecheckt moet worden of een respons op een asynchroon verzoek wel gelinkt kan worden aan het verzoek.

5. Kennisgeving- en synchronisatieberichten

Dit hoofdstuk behandelt de berichten die veelal gebruikt worden voor gegevenssynchronisatie met initiatief bij de bron (push): kennisgevingberichten en synchronisatieberichten. Er zijn vier soorten kennisgevingberichten:

- Lk01: enkelvoudig en asynchroon
- Lk02: enkelvoudig en synchroon
- Lk03: samengesteld en asynchroon
- Lk04: samengesteld en synchroon

De vier soorten zijn dus alle combinaties van synchroon en asynchroon en van enkelvoudig en samengesteld. Een enkelvoudige kennisgeving gaat over één object. Het type van dit object is gespecificeerd in het stuurgegeven *entiteittype*. Een samengestelde kennisgeving bevat mutaties voor meerdere objecten.

Een enkelvoudige kennisgeving heeft de volgende structuur:

```
<enkelvoudigeKennisgeving>
     <stuurgegevens>
          ...
     </stuurgegevens>
     <body>
          <parametersKennisgeving>
               ...
          </parametersKennisgeving>
          <object entiteittype=”XXX”>
               ...
          </object>
          <object entiteittype=”XXX”>
               ...
          </object>
     </body>
</enkelvoudigeKennisgeving>
```

met de naam enkelvoudigeKennisgeving nog vrij te kiezen. Het element `<parametersKennisgeving>` bevat gegevens waarmee de verwerking van een kennnisgeving wordt gestuurd. Het tweede element `<object>` komt alleen voor in kennisgevingen waarin de gegevens van een object wijzigen. Het eerste element `<object>` bevat dan de gegevens van voor de wijziging en het tweede element de actuele gegevens van het object. Het attribute `entiteittype` op de `<object>` elementen geeft hun entiteittype. Dit dient gelijk te zijn aan de waarde van het element `<entiteittype>` in de berichtstuurgegevens.

Een samengestelde kennisgeving heeft de volgende structuur:

```
<samengesteldeKennisgeving>
     <stuurgegevens>
          ...
     </stuurgegevens>
     <body>
          <parametersKennisgeving>
               ...
          </parametersKennisgeving>
          <enkelvoudigeKennisgeving>
               ...
          </enkelvoudigeKennisgeving>

          ...

          <enkelvoudigeKennisgeving>
               ...
          </enkelvoudigeKennisgeving>
     </body>
</samengesteldeKennisgeving>
```

met de naam samengesteldeKennisgeving nog vrij te kiezen. Een samengestelde kennisgeving heeft een body bestaande uit het element `<parametersKennisgeving>` gevolgd door twee of meer enkelvoudige kennisgevingen. Alle enkelvoudige kennisgevingen binnen een samengestelde kennisgeving dienen als één transactie verwerkt te worden. Als de verwerking van één van de enkelvoudige kennisgevingen faalt, dan dienen alle reeds verwerkte enkelvoudige kennisgevingen te worden teruggedraaid.

Een asynchroon kennisgevingbericht en een asynchroon synchronisatiebericht zijn notificaties. In de stuurgegevens in de body wordt aangegeven of zo'n asynchroon bericht al dan niet verwerkt moet worden in de database van het ontvangende systeem. Synchrone kennisgevingen en synchronisaties zijn geen notificaties maar transacties: ze moeten verwerkt worden in de database van de ontvanger.

Kennisgevingberichten mogen alleen worden gebruikt om actuele gegevens bij de ontvanger te wijzigen of te corrigeren en niet voor het wijzigen c.q. corrigeren van historische gegevens. Er is bewust gekozen voor de formulering 'actuele gegevens bij de ontvanger', omdat via kennisgevingen wel historie bij een ontvanger kan worden opgebouwd. In de eerste toevoegkennisgeving wordt de oudste situatie aangeboden en vervolgens wijzigkennisgevingen totdat de actuele situatie is bereikt. Het is het beste de toevoeg- en wijzigkennisgevingen te bundelen in een samengestelde kennisgeving om het opbouwen van de historie als één transactie af te handelen.

De verwerking van wijzigingen in historische gegevens is erg complex. Historische gegevens kunnen daarom alleen gecorrigeerd worden door middel van het synchronisatiebericht, waarbij een object inclusief zijn historie opnieuw wordt opgebouwd. Het synchronisatiebericht heeft net als de samengestelde kennisgeving een body met enkelvoudige kennisgevingen.

Beëindigde en vervangen relaties worden als historische relaties beschouwd. Na een verhuizing mogen wijzigingen en correcties op het historische verblijfsadres niet meer in een kennisgevingbericht worden opgenomen. Eventuele correcties dienen via een synchronisatiebericht te worden doorgegeven.

Dit hoofdstuk behandelt allereerst de stuurgegevens in de body van kennisgeving- en synchronisatieberichten. Daarna wordt de werking van het asynchrone enkelvoudige kennisgevingbericht besproken. De werking van de drie andere typen kennisgevingen wordt gedefinieerd uitgaande van de asynchrone kennisgeving. Tot slot wordt het synchronisatiebericht besproken, dat gebruikt wordt voor het opnieuw synchroniseren van een object inclusief zijn historie.

5.1 Stuurgegevens voor kennisgeving- en synchronisatieberichten

In paragraaf 2.2 zijn vier typen kennisgevingberichten onderkend:

- *Toevoeging*
  Bij een toevoeging heeft het zendende systeem vastgesteld dat een object waarnaar het bericht verwijst voor het zendende systeem relevant is geworden. Het object hoeft niet zojuist ontstaan te zijn. Een volwassene die zich vestigt in een gemeente wordt relevant voor die gemeente, maar is al geruime tijd geleden geboren. In verreweg de meeste gevallen zal het zendende systeem dit object ook in zijn eigen database hebben toegevoegd met misschien nog niet alle gegevens of met onjuiste gegevens.
- *Wijziging*
  Bij een wijziging heeft het zendende systeem vastgesteld dat er in de werkelijkheid eigenschappen (gegevens) zijn veranderd van het object waarnaar het bericht verwijst en geeft het die wijzigingen door aan de ontvanger. In verreweg de meeste gevallen zal het zendende systeem de gegevens ook in zijn eigen database gewijzigd hebben.
- *Verwijdering*
  Bij een verwijdering heeft het zendende systeem vastgesteld dat het object waarnaar de bericht verwijst, niet meer relevant is voor het zendende systeem. Het object hoeft niet zojuist opgehouden zijn te bestaan. Denk aan een leerling die met een diploma de school verlaat en niet meer relevant is voor de leerplichtadministratie. In verreweg de meeste gevallen zal het zendende systeem het object ook uit zijn database hebben verwijderd.
- *Correctie*
  Bij een correctie heeft het zendende systeem vastgesteld dat de eerder geleverde waarden niet correct zijn. Bij een correctie is het object in het bericht in de werkelijkheid niet gewijzigd. In verreweg de meeste gevallen zal het gaan om het doorgeven van een correctie op de gegevens in de database van het zendende systeem.

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

- ‘T’: *Toevoeging*
- ‘W’: *Wijziging*
- ‘V’: *Verwijdering*
- ‘C’: *Correctie*

De door de StUF-standaard gedefinieerde semantiek voor de mutatiesoorten 'T' en 'V' wijkt af van de gebruikelijke CRUD (Create, Read, Update en Delete) operaties voor een database. De mutatiesoorten 'T' en 'V' zeggen niets over het ontstaan of ophouden te bestaan van het object en ook niet altijd iets over het opvoeren of verwijderen van een record in de database. De update operatie is in StUF gesplitst in de mutatiesoorten 'W' en 'C'. Het is de verantwoordelijkheid van het ontvangende systeem om de verschillende mutatiesoorten te interpreteren. In veel gevallen voldoet een interpretatie in termen van de gebruikelijke CRUD-operaties.

In het (optionele) stuurgegeven *eventcode* kan het verzendende systeem extra informatie geven over de gebeurtenis die ten grondslag ligt aan het versturen van een kennisgevingbericht. Zo'n event of gebeurtenis (bijv. geboorte, huwelijk, overlijden, etc.) dient beschreven te zijn in het sectormodel. De *eventcode* heeft geen consequenties voor de verwerking van het bericht. Als een andere verwerking gewenst is dan voor een kennisgeving zonder *eventcode*, dan dient een vrij bericht gedefinieerd te worden. De *eventcode* is puur ter informatie voor de ontvangende partij. Er is voor dit uitgangspunt gekozen, om niet alle systemen die de StUF-kennisgevingen gebruiken te dwingen consequenties te verbinden aan een *eventcode*.

Een synchroon kennisgevingbericht dient het ontvangende systeem te verwerken in zijn database. Als het ontvangende systeem ook vraag/antwoordberichten ondersteunt, dan dient een vraag naar het object in een synchrone kennisgeving de nieuwe situatie als antwoord op te leveren, zodra het ontvangende systeem een Bv02-bevestiging heeft gestuurd als respons. Een asynchroon kennisgevingbericht kan naast verplicht te verwerken ook informatief bedoeld zijn. Verplicht te verwerken wil voor een asynchrone kennisgeving zeggen dat na verloop van tijd het ontvangende systeem als antwoord op een vraagbericht de nieuwe situatie in de kennisgeving teruggeeft. Of een asynchroon kennisgevingbericht informatief is of verplicht over te nemen geeft het stuurgegeven *indicatorOvername* aan met 'I' (informatief) respectievelijk 'V' (verplicht). Aanvullende afspraken over de omgang met dit stuurgegeven kunnen worden vastgelegd in het sectormodel.

Het laatste stuurgegeven in de body van een kennisgevingbericht is het optionele *tijdstipMutatie*. Dit geeft aan op welk tijdstip de mutatie in de kennisgeving is doorgevoerd in de database van het zendende systeem. Dit kan een ander tijdstip zijn dan het algemene stuurgegeven *tijdstipBericht*, want dit is het tijdstip waarop het bericht is aangemaakt.

De stuurgegevens van een kennisgeving zijn kinderen van het element `<parametersKennisgeving>` dat als eerste opgenomen in de body van een StUF-bericht. Het element `<parametersKennisgeving>` bevat maximaal vier elementen in de volgende structuur:

```
<parametersKennisgeving>
     <mutatiesoort>...</mutatiesoort>
     <eventcode>...</eventcode>
     <indicatorOvername>...</indicatorOvername>
     <tijdstipMutatie>...</tijdstipMutatie>
</parametersKennisgeving>
```

De structuur van `<parametersKennisgeving>` is gedefinieerd in het complexType `<ParametersKennisgeving>` in [StUFXSD].

	mutatiesoort	**eventcode**	**indicatorOvername**	**tijdstipMutatie**
Lk01	V	O	V	O
Lk02	V	O	–	O
Lk03	–	O	V	–
Lk04	–	O	–	–
Sa01, Sh01	–	–	V	–
Sa02, Sh02	–	–	–	–

Tabel 5.1. Het voorkomen van de kennisgeving stuurgegevens binnen de verschillende berichttypen.

Tabel 5.1 geeft aan welke stuurgegevens voor een kennisgeving voorkomen in welke berichttypen. V wil zeggen verplicht, O wil zeggen optioneel en – wil zeggen mag niet voorkomen.

Voor een Lk01- en Lk02-kennisgeving wordt het entiteittype van het object in het berichtstuurgegeven *entiteittype* opgenomen en de *mutatiesoort* in het berichtstuurgegeven *functie*. Voor een Lk03- en Lk04-kennisgeving wordt het berichtstuurgegeven *entiteittype* weggelaten en wordt het berichtstuurgegeven *functie* gevuld met een in het sectormodel te definiëren waarde. Voor alle synchronisatieberichten wordt het entiteittype van het object in het berichtstuurgegeven *entiteittype* opgenomen en wordt het berichtstuurgegeven *functie* weggelaten.

5.2 Respons en foutafhandeling voor kennisgeving- en synchronisatieberichten

Op een asynchrone kennisgeving of synchronisatiebericht verwacht de zender functioneel geen reactie van de ontvanger. De ontvanger dient uiteraard wel conform de regels in paragraaf 4.4.1 synchroon de ontvangst te bevestigen of aan te geven dat verwerking niet mogelijk is. Als de verwerking van een asynchrone kennisgeving of synchronisatiebericht faalt, dan voorziet de StUF-standaard niet in een geautomatiseerde verwittiging van de zender. Dit zal procedureel afgehandeld moeten worden.

Bij een synchroon kennisgevingbericht of synchroon synchronisatiebericht moet de zender direct geïnformeerd worden of de transactie al dan niet geslaagd is. In geval van een geslaagde transactie reageert de ontvanger door het als respons zenden van een Bv02-bevestigingsbericht met een lege body.

Indien de verwerking van een synchrone kennisgeving of synchronisatiebericht om wat voor reden dan ook niet is geslaagd, dan wordt een foutbericht Fo02 verzonden met een van de codes gedefinieerd in Tabel 4.1 voor soort fout 2. Ook een in het sectormodel gedefinieerde foutcode mag gebruikt worden. Als bij de verwerking van een samengestelde synchrone kennisgeving of van een synchroon synchronisatiebericht de verwerking van één van de enkelvoudige kennisgevingen faalde, dan wordt in het element `<details>` in het Fo02-foutbericht het referentienummer van de falende enkelvoudige kennisgeving opgenomen.

5.3 Regels voor enkelvoudige kennisgevingberichten

Deze paragraaf bespreekt de regels voor het vullen van de twee `<object>` elementen met objectgegevens in een enkelvoudige kennisgeving. Zoals in hoofdstuk 3 besproken hanteert de StUF-standaard een contentmodel waarbij een object elementen bevat die staan voor attributen en elementen die staan voor relaties. Deze paragraaf behandelt de regels voor het vullen van het element voor het object zelf, voor de relaties van het object en voor het element `<gerelateerde>` in relaties.

5.3.1 De structuur van objecten in enkelvoudige kennisgevingberichten

Om de verwerking van een kennisgeving richting database niet te gecompliceerd te maken mag een kennisgevingbericht slechts de volgende typen wijzigingen doorgeven:

1. Het toevoegen of verwijderen van een fundamentele entiteit van het type gedefinieerd in het berichtstuurgegeven *entiteittype.*
2. Het wijzigen of corrigeren van attributen van de fundamentele entiteit
3. Het toevoegen of beëindigen van een relatie-entiteit van de fundamentele entiteit of van een relatie-entiteit die uitsluitend via relatie-entiteiten is gekoppeld aan de fundamentele entiteit.
4. Het wijzigen of corrigeren van gegevens van een relatie-entiteit van de fundamentele entiteit of van een relatie-entiteit die uitsluitend via relatie-entiteiten is gekoppeld aan de fundamentele entiteit.
5. Het toevoegen van een fundamentele entiteit die als gerelateerde in een relatie voorkomt.

Samengevat komen deze regels erop neer dat alleen het object waar de kennisgeving betrekking op heeft en relaties (ook via relaties van relaties) van dat object in een kennisgeving kunnen worden opgevoerd, verwijderd en gemuteerd. Gerelateerden mogen worden opgevoerd, maar ze mogen niet worden gemuteerd via een kennisgeving voor het object waarin ze als gerelateerde worden toegevoegd. Een wijziging in een gerelateerde dient via een aparte enkelvoudige kennisgeving te worden doorgegeven. Door gebruik te maken van een samengestelde kennisgeving kan dit toch in één transactie gedaan worden. In een sectormodel mag deze beperking worden opgeheven, zodat ook wijzigingen in gerelateerden en hun relaties kunnen worden doorgegeven.

In het voorbeeld in Figuur 1 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 (nationaliteit)' (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. Van een gerelateerde fundamentele entiteit mogen tenzij anders gespecificeerd in het sectormodel alleen de kerngegevens worden opgenomen. Dankzij deze regels hebben kennisgevingberichten een relatief simpele structuur en is hun verwerking richting database niet te complex.

5.3.2 Het attribute verwerkingssoort

De exacte verwerking van een kennisgevingbericht is complex, omdat er veel variaties zijn zeker waar het gaat om relaties. Binnen kennisgevingen kent StUF daarom het attribute `verwerkingssoort`, waarmee kan worden aangegeven wat er precies verwacht wordt. Het attribute `verwerkingssoort` moet in een kennisgevingbericht worden opgenomen op alle elementen voor een fundamentele entiteit, relatie-entiteit of tabelentiteit. De volgende waarden voor `verwerkingssoort` zijn toegestaan:

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 andere relatie-entiteit
S De sleutel van een entiteit wordt gewijzigd
O De entiteit in het eerste `<object>` element wordt ontdubbeld door het samen te voegen met de entiteit in het tweede `<object>` element. Er wordt ontdubbeld, omdat beide entiteiten bleken te verwijzen naar hetzelfde object in de werkelijkheid.

Het gebruik van het attribute `verwerkingssoort` wordt in de volgende paragrafen nader uitgelegd voor de fundamentele entiteit op het hoogste niveau in het bericht, voor relatie-entiteiten en voor gerelateerde entiteiten. In het vervolg zullen we de fundamentele entiteit op het hoogste niveau in het bericht ook wel aanduiden als topfundamenteel.

5.3.3 Het vullen van de `<object>` elementen

Het toevoegen en verwijderen van objecten en het wijzigen van elementen voor attributen is relatief eenvoudig. De exacte regels verschillen voor een fundamenteel entiteittype en een tabelentiteittype.

Een enkelvoudige kennisgeving mag alleen worden verzonden, als minimaal één van de kerngegevens van de topfundamenteel een waarde heeft of als het attribute `sleutelOntvangend` bekend is.

Fundamenteel entiteittype

Bij een fundamenteel entiteittype kunnen zich de volgende situaties voordoen:

1. Een object is relevant geworden voor het zendende systeem.
2. Elementen voor attributen van het object zijn gewijzigd
3. Elementen voor attributen van het object zijn gecorrigeerd in het zendende systeem
4. De sleutel in het zendende systeem is gewijzigd
5. Het object wordt ontdubbeld door samenvoeging met een ander object (beide objecten in het systeem bleken te verwijzen naar hetzelfde object in de werkelijkheid)
6. Het object is niet langer relevant voor het zendende systeem
7. Alleen relaties zijn gewijzigd

Als alleen relaties zijn gewijzigd, dan is de topfundamenteel alleen nodig voor de identificatie van het object.

Onderstaande tabel vat de regels samen voor de opbouw van het bericht. Het kopje 'Oud/Actueel' verwijst naar de twee `<object>` elementen. Als er één `<object>` element is (mutatiesoort 'T' en 'V'), dan wordt dit aangeduid met 'Actueel'. Als er twee `<object>` elementen zijn, dan wordt het eerste aangeduid met 'Oud' en het tweede met 'Actueel'. De regels hebben betrekking op het vullen van de elementen `beginGeldigheid` en `eindGeldigheid`, het attribute `verwerkingssoort` en van de elementen voor attributen van de topfundamenteel. Het attribute

`sleutelVerzendend` is niet in de tabel opgenomen, omdat dit attribute 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 attribute `sleutelVerzendend` niet opgenomen. De attributes `sleutelOntvangend` en `sleutelGegevensbeheer` zijn niet in de tabel opgenomen omdat de regels ervoor niet afhangen van de mutatiesoort en de verwerkingssoort.

Situatie	Mutatie soort	Oud/ Actueel	begin-Geldigheid	eind-Geldigheid	verwer-kings-soort	Overige elementen
Relevant geworden	T	Actueel	N	N	T	Alle bekende elementen
Wijziging	W	Oud	O	N	W	Kerngegevens als `sleutelOntvangend` ontbreekt + te wijzigen elementen
		Actueel	N	N	W	Kerngegevens als `sleutelOntvangend` ontbreekt + gewijzigde elementen
Correctie	C	Oud	O	O	W	Kerngegevens als `sleutelOntvangend` ontbreekt + te corrigeren elementen
		Actueel	N	N	W	Kerngegevens als `sleutelOntvangend` ontbreekt + gecorrigeerde elementen
Sleutelwijziging	C	Oud	-	-	S	Kerngegevens en zo mogelijk `sleutelOntvangend`
		Actueel	-	-	S	Kerngegevens en zo mogelijk `sleutelOntvangend`
Ontdubbeling	C	Oud	-	-	O	Kerngegevens en zo mogelijk `sleutelOntvangend`
		Actueel	-	-	O	Kerngegevens en zo mogelijk `sleutelOntvangend`
Irrelevant geworden	V	Actueel	-	-	V	Kerngegevens als `sleutelOntvangend` ontbreekt
Identificatie	W of C	Oud	-	-	I	Kerngegevens als `sleutelOntvangend` ontbreekt
		Actueel	-	-	I	Kerngegevens als `sleutelOntvangend` ontbreekt

Tabel 5.2 Regels voor de opbouw van een bericht

Mutatiesoort	De codes voor het veld mutatiesoort gedefinieerd in paragraaf 5.1	
`beginGeldigheid` en `eindGeldigheid` 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 3.2.1	

Tabel 5.3 Legenda tabel: Regels voor de opbouw van een bericht

Het attribute `sleutelVerzendend` moet in de topfundamenteel 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 attribute `sleutelVerzendend` niet opgenomen. De attributes `sleutelOntvangend` en `sleutelGegevensbeheer` zijn optioneel.

De elementen `<beginGeldigheid>` en `<eindGeldigheid>` mogen alleen opgenomen worden, als:
1. in het sectormodel gespecificeerd is dat voor het betreffende fundamentele entiteittype historie relevant is EN
2. de entiteit of de groep minstens één element bevat waarvoor volgens het sectormodel historie relevant is.

In het eerste `<object>` element specificeert het element `<eindGeldigheid>` het eind van de geldigheid van het 'oude' gegeven. Dit is het enige gegeven in het eerste `<object>` element dat een nieuwe waarde mag hebben. In het tweede `<object>` element geven `<beginGeldigheid>` en `<eindGeldigheid>` het tijdvak geldigheid van de nieuwe gegevens aan.

De tijdvakken geldigheid in verschillende groepen hoeven bij een wijziging niet identiek te zijn. Een wijziging in groep A kan ingaan op datum A en een wijziging in groep B op datum B. Binnen een entiteit en een groep mogen uiteraard alleen elementen worden opgenomen die op hetzelfde moment wijzigen, waarvoor historie niet relevant is of die niet wijzigen.

Een correctie vervangt een foutieve waarde door de juiste waarde. Een correctie heeft meestal geen impact op het tijdvak geldigheid, omdat er in de werkelijkheid niets is veranderd. Er zijn twee uitzonderingen:
1. Als de correctie betrekking heeft op het ontstaan of ophouden te bestaan van het object, zal de corresponderende datum van het tijdvak geldigheid ook veranderen.
2. Als `<beginGeldigheid>` of `<eindGeldigheid>` zelf gecorrigeerd worden.

Bij een sleutelwijziging, ontdubbeling en verwijdering wordt het tijdvak geldigheid niet opgenomen, omdat er in de werkelijkheid niets is gebeurd. Bij een sleutelwijziging wordt een sleutel vervangen in de database. Bij een ontdubbeling worden twee objecten met verschillende sleutels die verwijzen naar hetzelfde object in de werkelijkheid ontdubbeld. Het object in het eerste `<object>` element bestaat niet langer en is samengevoegd met het object in het tweede `<object>` element. Ook een ontdubbeling gebeurt alleen in de database. Bij een verwijdering vindt het zendende systeem het object niet langer relevant. In de werkelijkheid is er niets met het object gebeurt.

Als de verwerkingssoort 'I' is, wordt het tijdvak geldigheid niet opgenomen.

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 eind geldigheid relevant zijn, dienen deze te zijn gedefinieerd als elementen binnen de tabelentiteit. Bij tabelentiteiten mogen ook de attributes `sleutelVerzendend`, `sleutelOntvangend` en `sleutelGegevensbeheer` niet voorkomen, omdat tabelentiteiten geen sleutels hebben. Bij een tabelentiteit kunnen de verwerkingssoorten 'S', 'O' en 'I' niet voorkomen.

5.3.4 Het vullen van relatie-entiteiten en gerelateerde entiteiten

Een wijziging kan betrekking hebben op relaties met andere objecten. Een relatie-entiteit wordt alleen in een bericht opgenomen, indien voor het gerelateerde object minimaal voor één kerngegeven een waarde bekend is of als het attribute `sleutelOntvangend` bekend is. Het opnemen van relaties is complex, omdat er veel verschillende situaties zijn. Deze worden hieronder toegelicht. De daarna volgende tabel geeft voor elke situatie de regels voor het opnemen van elementen, attributes, het tijdvak geldigheid en het tijdvak relatie. Waar nodig worden deze regels bij de beschrijving van de situaties toegelicht.

1. *Mutatiesoort 'T' en verwerkingssoort 'T': Een relatie wordt toegevoegd in een toevoegkennisgeving*
   Van een object kunnen relaties relevant zijn en daarom in een toevoegkennisgeving worden opgenomen. Alleen nog niet beëindigde relaties zijn relevant. Van bepaald type relatie worden net zoveel relatie-entiteiten opgenomen als er relevante relaties zijn. Wanneer een persoon bijvoorbeeld vier kinderen heeft, dan wordt vier keer de relatie-entiteit van een persoon naar zijn kinderen opgenomen.

2. *Mutatiesoort 'W' of 'C' en verwerkingssoort 'T': Een relatie wordt toegevoegd in een wijzig- of correctiekennisgeving*
   Nu is er iets veranderd in het object: het object heeft een nieuwe relatie gekregen. Het gaat dus om een kennisgeving met mutatiesoort 'W' of 'C'. In het eerste `<object>` element wordt de relatie-entiteit opgenomen met de attributes `entiteittype`, `verwerkingssoort=”T”` en `StUF:noValue=”geenWaarde”` en met een lege element-inhoud. De ‘nieuwe’ relatie wordt met `verwerkingssoort=”T”` opgenomen in het tweede `<object>` element.

3. *Mutatiesoort 'W' en verwerkingssoort 'W': De gegevens van een relatie-entiteit wijzigen in de werkelijkheid*
   Als een huwelijk wordt ontbonden, worden krijgen de datum, plaats en reden van de huwelijksontbinding binnen het relatie-entiteittype huwelijk een waarde. Er is dan een oud en een actueel voorkomen van de relatie nodig en dus ook een oud en een actueel voorkomen van de topfundamenteel. Gegevens van een relatie kunnen alleen wijzigen, als de relatie niet beëindigd is. `<beginRelatie>` en `<eindRelatie>` kunnen nooit geraakt worden. `<beginRelatie>` kan alleen gecorrigeerd worden en `<eindRelatie>` krijgt alleen een waarde bij het beëindigen van een relatie of wordt na het beëindigen van de relatie gecorrigeerd. `<beginRelatie>` en `<eindRelatie>` worden bij een wijziging van de relatiegegevens alleen in de relatie-entiteit opgenomen, als ze deel uitmaken van de kerngegevens. Omdat er bij een wijziging nog geen `<eindRelatie>` is, zal het element `<eindRelatie>` met een lege elementinhoud met als attribute `StUF:noValue=”geenWaarde”` worden opgenomen.

4. *Mutatiesoort 'C' en verwerkingssoort 'W': De gegevens van een relatie-entiteit worden gecorrigeerd*
   Dit is het geval als de mutatiesoort ‘C’ (Correctie) is. In de werkelijkheid is er niets gebeurd met de relatie. Een administratieve fout in de gegevens van de relatie wordt gecorrigeerd. Ook `<beginRelatie>` en

<eindRelatie> kunnen worden gecorrigeerd. Ze worden in een correctie dus anders behandeld dan in een wijziging.

5. *Mutatiesoort 'W' en verwerkingsoort 'V': 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.
   In het eerste <object> element wordt de te verwijderen relatie opgenomen en in het tweede <object> element wordt een relatie-entiteit opgenomen met de attributes entiteittype, verwerkingssoort="V" en StUF:noValue="geenWaarde" en een lege elementinhoud.

6. *Mutatiesoort 'W' en verwerkingssoort 'E': Een relatie wordt beëindigd*
   Dit betekent dat de relatie niet langer in de werkelijkheid bestaat, bijvoorbeeld het niet langer hebben van een bepaalde verblijfstitel. 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 de relatie niet meer relevant is, 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 <eindRelatie> een nieuwe waarde krijgen. Deze nieuwe waarde wordt gespecificeerd in de relatie-entiteit in het eerste <object> element. In het tweede <object> element wordt een relatie-entiteit opgenomen met de attributes entiteittype, verwerkingssoort="E" en StUF:noValue="geenWaarde" en een lege elementinhoud.
   Een relatie kan ook beëindigd worden voor relatie-entiteiten waarin de elementen <beginRelatie> en <eindRelatie> niet voorkomen. Het verzendende systeem kan dan niet specificeren op welk moment de relatie beëindigd is.
   Het tijdvak geldigheid mag bij een beëindiging niet gespecificeerd worden. De op gegevens zoals ze gelden op het moment van beëindigen blijven geldig.

7. *Mutatiesoort 'W' en verwerkingssoort 'R': 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.
   <beginRelatie> bevat in het eerste <object> element het oorspronkelijke begintijdstip van de relatie en <eindRelatie> het eindtijdstip van de relatie. In het eerste <object> element wordt het tijdvak geldigheid niet opgenomen, omdat het enige element dat wijzigt <eindRelatie> is. Voor het tijdvak geldigheid van een vervangen relatie geldt hetzelfde als voor het tijdvak geldigheid van een beëindigde relatie. Het tweede <object> element bevat in de elementen <beginGeldigheid> en <eindGeldigheid> het tijdvak geldigheid van de nieuwe relatie en in <beginRelatie> het begintijdstip van de nieuwe relatie. Het eindtijdstip heeft geen waarde.

8. *Mutatiesoort 'C' en verwerkingssoort 'R' of 'V': 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. Dit bericht heeft dezelfde opbouw als een vervanging (er komt wel een andere relatie voor in de plaats) of een verwijdering (er komt geen andere relatie voor in de plaats) .

9. *Mutatiesoort 'W' of 'C' en verwerkingssoort 'I':Relatie is opgenomen als kerngegeven of omwille gerelateerde*
   De relatie is alleen opgenomen in het bericht omdat deze deel uitmaakt van de kerngegevens van de topfundamenteel of omdat er gegevens in een gerelateerde entiteit worden gewijzigd. Met de relatie-entiteit zelf gebeurt niets.

10. *Mutatiesoort 'C' en verwerkingssoort 'S': Sleutelwijziging*
    De sleutel in het zendende systeem van de relatie wordt gewijzigd.

Soort kennisgeving	Mut. soort	Oud/ Actueel	Topfunda-menteel	Relatie-entiteit						Gerelateerde entiteit		
			verwerkings-soort	begin-Geldigheid	eind-Geldigheid[9]	verwerkings-soort	beginRelatie	eindRelatie[10]	Rest inhoud	sleutel-Verzendend	verwerkings-soort	Element-inhoud
Toevoegen relatie bij toevoegen object	T	Actueel	T	N	N	T	N	N	Alles	V	I	K/sleutel
Toevoegen relatie bij wijzigen object	W	Oud	W/I	-	-	T	-	-	Leeg	-	-	-
		Actueel	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
		Actueel	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
		Actueel	W/I	N	N	W	C, N	C, N	`K/Sleutel` + gecorrig. geg.	V	I	K/Sleutel
Verwijderen relatie	W[11]	Oud	W/I	-	-	V	K, O	K, Leeg	`K/Sleutel`	V	I	K/Sleutel
		Actueel	W/I	-	-	V	-	-	Leeg	-	-	-
Beëindigen relatie	W	Oud	W/I	-	-	E	O	N	`K/Sleutel`	V	I	K/Sleutel
		Actueel	W/I	-	-	E	-	-	Leeg	-	-	-
Vervangen relatie	W/C	Oud	W/I	-	-	R	O	N	`K/Sleutel`	V	I	K/Sleutel
		Actueel	W/I	N	N	R	N	N	Alles	V	I	K/Sleutel
Correctie toevoeging van een relatie	C	Oud	W/I	-	-	V	K, O	K, O	`K/Sleutel`	V	I	K/Sleutel
		Actueel	W/I	-	-	V	-	-	Leeg	-	-	-
Identificatie	W, C, V	Oud	W/I/V/S	-	-	I	K, O	K, O	`K/Sleutel`	V	I	K/Sleutel
		Actueel	W/I/V/S	-	-	I	K, O	K, O	`K/Sleutel`	V	I	K/Sleutel
Sleutelwijziging/ ontdubbeling	C	Oud	I	-	-	S/O	K	K	`K + sleutel verzen-dend`	V	I	K/Sleutel
		Actueel	I	-	-	S/O	K	K	`K + sleutel verzen-dend`	V	I	K/Sleutel

Tabel 5.4 Regels voor het vullen van de attributen en elementen binnen een relatie-entiteit

[9] `beginGeldigheid` en `eindGeldigheid` 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.

[10] `beginRelatie` en `eindRelatie` worden – mits gedefinieerd in het sectormodel – opgenomen bij de verwerkingssoorten T, R, E. Bij verwerkingssoort W en V worden ze alleen opgenomen, als `beginRelatie` een kerngegeven is. Bij verwerkingssoort 'W' kunnen ze ook worden opgenomen als de mutatiesoort 'C' is.

[11] 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 5.1	
Verwerkingssoort	De codes voor het veld verwerkingssoort gedefinieerd in paragraaf 5.3.2	
`beginGeldigheid` en `eindGeldigheid` NB: Als de één voorkomt is de ander verplicht!	-	Mag niet voorkomen
	O	Oorspronkelijke waarde, als tijdvakGeldigheid wordt ondersteund
	N	Nieuwe waarde, als tijdvakGeldigheid wordt ondersteund
`beginRelatie` en `eindRelatie` 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 `beginRelatie` of `eindRelatie` 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.
	K + sleutel verzendend	Neen in de relatie-entiteit alle kerngegevens op. Neem in de 'oude' relatie-entiteit de oorspronkelijke `sleutelVerzendend` op en in de 'actuele' entiteit de nieuwe `sleutelVerzendend`
`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 5.5 Legenda Tabel 5.4 Regels voor het vullen van de attributen en elementen binnen een relatie-entiteit

11. Mutatiesoort 'C' en verwerkingssoort 'O': Ontdubbeling

De relatie in het eerste `<object>` element wordt ontdubbeld door hem samen te voegen met de relatie in het tweede `<object>` element.

Bovenstaande tabel beschrijft in detail de regels voor het vullen van de attributen en elementen binnen een relatie-entiteit. De vulling van de elementen van de gerelateerde entiteit wordt gespecificeerd na onderstaande tabel. De attributes `sleutelVerzendend`, `sleutelOntvangend` en `sleutelGegevensbeheer` van de relatie-entiteit zijn niet in deze tabel opgenomen, omdat deze attributen optioneel zijn, tenzij de verwerkingssoort 'O' of 'S' is. `sleutelVerzendend` is niet verplicht zoals bij fundamentele entiteiten, omdat de identificatie van een relatie bijna nooit een probleem is, omdat de entiteit vanwaaruit de relatie ligt en de gerelateerde meestal de relatie al uniek identificeren. Daarnaast beschikken veel systemen niet over een `sleutelVerzendend` voor een relatie.

Net zoals bij fundamentele entiteiten worden de kerngegevens van de relatie niet in het bericht opgenomen als `sleutelOntvangend` gespecificeerd is. Bij verwerkingssoort 'O' en 'S' is het attribute `sleutelVerzendend` verplicht. De attributes `sleutelOntvangend` en `sleutelGegevensbeheer` mogen in de relatie worden opgenomen. Bij verwerkingssoort 'O' en 'S' zijn de kerngegevens ook verplicht, als `sleutelOntvangend` in de relatie wordt opgenomen.

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 `<beginGeldigheid>` en `<eindGeldigheid>` niet voorkomen. Zij zijn daarom niet in de tabel opgenomen. De attributes `sleutelOntvangend` en `sleutelGegevensbeheer` zijn ook niet in de tabel opgenomen onder de gerelateerde entiteit. Deze attributes zijn optioneel, terwijl `sleutelVerzendend` verplicht is, als het zendend systeem erover beschikt. Als het attribute `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 5.3.5 wordt aangegeven hoe de gerelateerde entiteit dan gevuld dient te worden.

5.3.5 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 in een toevoegkennisgeving
2. Toevoegen van een relatie in een wijzig- of correctiekennisgeving
3. Vervangen van een relatie in een wijzig- of correctiekennisgeving

Denk bijvoorbeeld aan het door de GBA in het bericht opnemen van de partner bij het specificeren van een huwelijk. De GBA kent de partner niet als een onafhankelijke persoon en zal een voor het ontvangende systeem onbekende partner meegeven. Als het zendende systeem niet zeker weet of het ontvangende systeem een gerelateerde kent, dient het gerelateerde object met 'T' als verwerkingssoort te worden opgenomen. Voor de gerelateerde entiteit dezelfde regels als voor de topfundamenteel met inachtneming van de in het sectormodel gedefinieerde structuur voor de gerelateerde. Als het zendende systeem zeker weet dat het ontvangende systeem de gerelateerde kent, dan wordt de gerelateerde met verwerkingssoort 'I' en alleen de kerngegevens opgenomen.

In een sectormodel kan het gebruik van verwerkingssoort 'T' voor gerelateerden verboden worden. Wanneer het zendende systeem verwacht dat het ontvangende systeem de gerelateerde niet kent, dan wordt in dat geval in een samengestelde kennisgeving eerst een toevoegkennisgeving voor de gerelateerde opgenomen en vervolgens de kennisgeving met de nieuwe relatie met zijn gerelateerde. De gerelateerde in de relatie krijgt nu verwerkingssoort 'I' en wordt opgenomen met alleen de kerngegevens.

In het sectormodel mag ook worden gespecificeerd dat in een wijzig- of correctiekennisgeving gegevens van het gerelateerde object wel 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 een topfundamenteel. 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 topfundamenteel met inachtneming van de beperkingen gedefinieerd in het sectormodel.

Tabel 5.6 specificeert het vullen van de verwerkingssoort in de gerelateerde entiteit.

Soort kennisgeving	Mutatie-soort	Oud/ Actueel	Verwerkingssoort		
			Topfundamenteel	Relatie-entiteit	Gerelateerde entiteit
Toevoegen relatie bij toevoegen object	T	Actueel	T	T	I/T
Toevoegen relatie bij wijzigen object	W	Oud	I/W	T	I/T
		Actueel	I/W	T	I/T
Wijzigen gerelateerd object	W	Oud	I/W	I/W	I/W
		Actueel	I/W	I/W	I/W
Corrigeren gerelateerd object	C	Oud	I/W	I/W	I/W
		Actueel	I/W	I/W	I/W
Vervangen relatie	W	Oud	I/W	R	I
		Actueel	I/W	R	I/T

Tabel 5.6 Invullen attribute `verwerkingssoort`

5.3.6 Aanvullende regels

- Indien in een kennisgevingbericht van 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 het 'oude' als het 'nieuwe' voorkomen opgenomen.
- Een relatie van een object 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 relatie-entiteit ofwel in twee verschillende kennisgevingberichten, bijvoorbeeld bij de opbouw van historie.
- Indien een wijzig- of correctiekennisgeving meerdere relaties van al dan niet hetzelfde relatie-entiteittype bevat, dan dienen de relatie-entiteiten voor deze relaties in exact dezelfde volgorde in de twee `<object>` elementen te

worden opgenomen. Het is dus toegestaan om de gegevens van twee verschillende huwelijken in één kennisgevingbericht te wijzigen.

5.4 Regels voor samengestelde kennisgevingberichten

De body van een samengesteld kennisgevingbericht bestaat uit twee of meer enkelvoudige kennisgevingberichten. Deze enkelvoudige kennisgevingberichten dienen door het ontvangende systeem verwerkt te worden in de volgorde waarin ze in de samengestelde kennisgeving staan. De enkelvoudige kennisgevingberichten dienen elk te voldoen aan de regels zoals hierboven beschreven. Het ontvangende systeem verwerkt de enkelvoudige kennisgevingberichten op exact dezelfde wijze als losse enkelvoudige kennisgevingen, maar wel binnen één databasetransactie. Zodra de verwerking van één van de enkelvoudige kennisgevingen in de samengestelde kennisgeving faalt, dient de verwerking van alle reeds verwerkte enkelvoudige kennisgevingen te worden teruggedraaid. In geval van synchroon samengesteld kennisgevingbericht wordt bovendien een Fo02-foutbericht verstuurd, zie paragraaf 5.2.

Er gelden nog enkele aanvullende regels:
- Een *synchrone* samengestelde kennisgeving mag alleen *synchrone* enkelvoudige kennisgevingen bevatten.
- Een *asynchrone* samengestelde kennisgeving mag alleen *asynchrone* enkelvoudige kennisgevingen bevatten.
- De asynchrone enkelvoudige kennisgevingen dienen allemaal dezelfde *indicatorOvername* te hebben als de asynchrone samengestelde kennisgeving waarin ze zitten.

5.5 Regels voor synchronisatieberichten

StUF kent twee berichtsoorten voor het corrigeren van gegevens:
1. Synchronisatiebericht actueel (berichtcode Sa01 voor de asynchrone variant en Sa02 voor de synchrone variant)
2. Synchronisatiebericht historisch (berichtcode Sh01 voor de asynchrone variant en Sh02 voor de synchrone variant)

Met een synchronisatiebericht kunnen de gegevens van een object van een bepaald entiteittype worden gecorrigeerd. Het stuurgegeven *entiteittype* geeft aan om welk entiteittype het gaat. Een synchronisatiebericht actueel verschilt van de correctiekennisgeving, omdat de zender niet hoeft aan te geven welke gegevens gecorrigeerd worden. Het synchronisatiebericht actueel vraagt aan de ontvanger om zijn actuele gegevens te vervangen door de gegevens in het bericht. Het synchronisatiebericht historisch maakt het mogelijk om historische gegevens te corrigeren. Het synchronisatiebericht historisch is de enige voorziening die StUF hiervoor biedt.

De synchrone en asynchrone varianten van de synchronisatieberichten verschillen alleen qua stuurgegevens. De body van een synchroon synchronisatiebericht is gelijk aan de body van een asynchroon synchronisatiebericht. Ook binnen de body van een synchrone synchronisatiebericht worden de asynchrone varianten van een erin op te nemen synchronisatiebericht actueel of kennisgevingbericht gebruikt.

Voor synchronisatieberichten is van toepassing de foutafhandeling gedefinieerd in de paragrafen 4.4.1 en 4.4.3.

5.5.1 Synchronisatiebericht actueel

Het synchronisatiebericht actueel is bedoeld voor het corrigeren van alleen de actuele gegevens van een object. De body bevat als laatste element `<actueel>` met daarin een toevoegkennisgevingbericht inclusief stuurgegevens. De ontvanger vervangt in zijn systeem alle actuele gegevens van een object door de waarden in het element `<actueel>`. Als in het synchronisatiebericht actueel een bepaald element niet voorkomt of voorkomt met `StUF:noValue="nietOndersteund"`, dan mag de ontvanger een eigen waarde voor dat element handhaven. Bij de verwerking van een synchronisatiebericht actueel worden eventueel aanwezige historische gegevens ongemoeid gelaten. Het tijdvak geldigheid van het actuele voorkomen wordt overgenomen uit het synchronisatiebericht. Het is mogelijk dat door de verwerking van een synchronisatiebericht actueel een gat ontstaat met het tijdvak geldigheid van de meest recente historische voorkomen of dat een andere inconsistentie ontstaat. Het verdient daarom de voorkeur om te synchroniseren met behulp van een synchronisatiebericht historisch, als er ook historische gegevens zijn.

Het element `<actueel>` kan voorafgegaan worden door een element `<gerelateerden>` met daarin één of meer elementen voor toevoegkennisgevingen inclusief stuurgegevens voor een gerelateerde van een bepaald entiteittype die voorkomt in `<actueel>`. In het sectormodel worden de namen voor de elementen voor de verschillende toegestane typen gerelateerden gedefinieerd. De toevoegkennisgevingen voor gerelateerden worden verwerkt als normale toevoegkennisgevingen met `indicatorOvername` 'V'. De elementen `<gerelateerden>` zijn nodig

om eventueel nog niet bij de ontvanger bekende gerelateerden toe te laten voegen. Wanneer zo'n gerelateerde ook gesynchroniseerd moet worden, dan is hiervoor een apart synchronisatiebericht nodig.

Voor de toevoegkennisgevingen in `<actueel>` en `<gerelateerden>` gelden de volgende aanvullende regels:
- De stuurgegevens dienen te voldoen aan de regels die gelden voor berichten verzonden na het synchronisatiebericht actueel en na de eventueel eraan voorafgaande toevoegkennisgevingen in het bericht.
- Het element `<mutatiesoort>` moet voorkomen met als waarde 'T'. Het element `<indicatorOvername>` moet voorkomen met als waarde 'V'. De elementen `<eventcode>` en `<tijdstipMutatie>` mogen niet voorkomen.
- Het element `<object>` in de toevoegkennisgeving bevat alle actuele waarden zoals de zender die kent, de `sleutelZendendSysteem` en zo mogelijk de `sleutelOntvangendSysteem` en/of `sleutelGegevensbeheer`. De `verwerkingssoort` dient op dezelfde wijze gevuld te zijn als bij een toevoegkennisgeving met als uitzondering dat de `verwerkingssoort` van een gerelateerde altijd 'I' dient te zijn. De kerngegevens zijn verplichte elementen ook al is de `sleutelOntvangendSysteem` aanwezig.

5.5.2 Synchronisatiebericht historisch

Het synchronisatiebericht historisch is bedoeld voor het corrigeren van historische en desgewenst ook actuele gegevens van een object. De ontvanger dient in zijn systeem de historische en actuele gegevens van het object te vervangen door de gegevens in het synchronisatiebericht historisch voorzover ze geleverd worden in het synchronisatiebericht historisch. Een ontvanger die geen historische gegevens kent, dient alleen de actuele gegevens te vervangen door de actuele gegevens in het bericht conform de regels voor het synchronisatiebericht actueel.

Als een ontvanger vanuit verschillende bronnen historische gegevens krijgt aangeleverd, dan is de verwerking van het synchronisatiebericht historisch complex. De aanwezige historische gegevens kunnen niet simpelweg vervangen worden, omdat er ook historie kan zijn voor gegevens die niet geleverd worden in het synchronisatiebericht historisch en deze historie dan verloren gaat.

Het synchronisatiebericht historisch bevat als eerste element in de body altijd een element `<actueel>` met als inhoud een synchronisatiebericht actueel inclusief de stuurgegevens. Er is om twee redenen gekozen voor het opnemen van een compleet synchronisatiebericht actueel:
- Een ontvanger die alleen geïnteresseerd is in actuele gegevens hoeft uitsluitend het synchronisatiebericht actueel te verwerken en kan de verdere inhoud van het synchronisatiebericht historisch negeren. Ontvangers die alleen geïnteresseerd zijn in actuele gegevens dienen dus op deze manier een synchronisatiebericht historie te kunnen verwerken.
- Een reeds in de standaard voorkomende constructie wordt hergebruikt en er hoeft geen nieuwe constructie te worden geïntroduceerd.

Dit eerste element `<actueel>` is ook bedoeld om het object aan de hand van actuele gegevens te identificeren.

Na het element `<actueel>` volgt nul of één element `<historie>`. In het `<historie>` element zit een element `<oudste>` met als inhoud een toevoegkennisgeving inclusief stuurgegevens voor het te synchroniseren object. Voor de body van deze toevoegkennisgeving gelden dezelfde regels als voor de body van de toevoegkennisgeving in het element `<actueel>` in het synchronisatiebericht actueel. Alleen bevat het element `<object>` nu de oudste situatie van het te synchroniseren object, dat wil zeggen:
- Alle gegevens die sinds het ontstaan van het object niet zijn gewijzigd;
- Alle gegevens waarvoor historie niet relevant is met hun actuele waarde;
- Alle niet-vervangen en niet-beëindigde relaties met hun oudste situatie;
- Beëindigde relaties waarvoor historie relevant is met hun oudste situatie.

Als een gegeven waarvoor historie relevant is sinds het ontstaan van het object is gewijzigd, dan maakt de waarde op het moment van ontstaan deel uit van de historische situatie. Als een relatie is vervangen, dan maakt de oudste van een verzameling vervangen relaties onderdeel uit van de oudste situatie. Bij relatiegegevens waarvoor historie relevant is, wordt de waarde op het moment van ontstaan van de relatie opgenomen in de oudste situatie. Voor een relatiegegeven waarvoor historie niet relevant is, wordt de actuele waarde opgenomen in de oudste situatie.

Het element `<oudste>` kan worden voorafgegaan door een element `<gerelateerden>` met daarin elementen voor toevoegkennisgevingen inclusief stuurgegevens voor een gerelateerde van een bepaald entiteittype in het `<oudste>` element. De namen voor de elementen voor de verschillende typen gerelateerden dienen in het

sectormodel gespecificeerd te worden. Al deze toevoegkennisgevingen voor gerelateerden dienen actuele gegevens van de gerelateerde te bevatten. Het is dus niet mogelijk om ook gerelateerde objecten te synchroniseren in een synchronisatiebericht historisch. Ze dienen te voldoen aan dezelfde regels als een toevoegkennisgeving voor een gerelateerde in een synchronisatiebericht actueel.

Na het element <oudste> volgen nul of meer elementen <wijziging> met daarin wijzigkennisgevingen inclusief stuurgegevens met de historische wijzigingen van oud naar nieuw, totdat de actuele situatie is bereikt. Elk element <wijziging> mag weer worden voorafgegaan door een element <gerelateerden> met daarin elementen voor toevoegkennisgevingen inclusief stuurgegevens voor een gerelateerde van een bepaald entiteittype in het <wijziging> element.

De body van deze wijzigkennisgevingen bevatten het element <mutatiesoort> met als waarde 'W' en het element <indicatorOvername> met als waarde 'V'. De elementen <eventcode> en <tijdstipMutatie> mogen niet voorkomen. De beide <object> elementen bevatten de wijziging. Voor deze elementen gelden de normale regels voor een wijzigkennisgeving. Er is gekozen voor het gebruik van complete wijzigkennisgevingen, omdat:

- de functionaliteit voor het verwerken van een wijzigkennisgeving kan worden hergebruikt;
- er wordt aangesloten bij de structuur voor een samengesteld kennisgevingbericht (alleen de elementnamen verschillen).

Als het element <historie> na <oudste> geen verdere elementen bevat, dan is het doel van het synchronisatiebericht historie om onterecht opgevoerde historische gegevens te verwijderen.

5.5.3 Foutafhandeling

Als de verwerking van één van de berichten binnen een synchronisatiebericht actueel of historisch faalt, dan dient de verwerking in zijn geheel teruggedraaid te worden.

Voor een synchroon synchronisatiebericht zijn bovenop de foutafhandeling in de paragrafen 4.4.1, 4.4.3 en 5.2 extra foutsituaties gedefinieerd die leiden tot het niet verwerken cq terugdraaien van de verwerking van het synchronisatiebericht. De vulling van het Fo02-foutbericht staat in onderstaande tabel.

De extra foutsituaties zijn:

- De ontvanger kan het object niet kan vinden in zijn systeem: 'Object niet gevonden'.
- De ontvanger kan het object niet uniek kan identificeren in zijn systeem: 'Dubbelen voor object gevonden'.
- Bij verwerking van een synchroon synchronisatiebericht historisch blijkt dat het bericht inconsistenties bevat (bijvoorbeeld gaten in de tijdvakken geldigheid of een verkeerde volgorde ervan): 'Synchronisatiebericht historisch niet consistent'.
- Als door de verwerking van een synchroon synchronisatiebericht actueel inconsistenties ontstaan met reeds aanwezige historische gegevens, dan wordt in bevestigingsbericht het element <melding> gevuld met 'StUF0004: Actuele gegevens nu inconsistent met historische'. Het synchronisatiebericht actueel wordt in dit geval wel verwerkt.

StUF voorziet niet in een mechanisme voor het communiceren van deze foutsituaties in geval van een asynchroon synchronisatiebericht.

Foutsituatie (= omschrijving)	**Soort fout**	**Code**	**Plek**	**Details**
Object niet gevonden	2	StUF064	Server	
Dubbelen voor object gevonden	2	StUF067	Server	
Synchronisatiebericht historisch niet consistent	2	StUF070	Client	

Tabel 5.7: Foutsituaties bij de afhandeling van synchrone synchronisatieberichten

6. Vraag- en antwoordberichten

De vraag/antwoordberichten in StUF zijn bedoeld voor het opvragen van gegevens in de database van het antwoordende systeem. Gegevens kunnen zowel synchroon (direct antwoord) als asynchroon (het antwoordende systeem mag zelf bepalen wanneer de antwoordberichten naar het vragende systeem worden gezonden) worden opgevraagd.

Het antwoordende systeem streeft bij het beantwoorden van een synchrone vraag naar het zo goed mogelijk bedienen van het vragende systeem. Synchrone antwoordberichten kunnen daarom meer dan één object van het gevraagde entiteittype bevatten, zodat op basis van één vraag/antwoord interactie een lijst met objecten kan worden getoond. Een synchroon antwoordbericht hoeft niet het volledige antwoord op een vraag te bevatten. Omwille van de performance kan ervoor gekozen zijn slechts een deel van de gevraagde objecten terug te geven. Het vragende systeem kan de rest van de objecten opvragen in een volgend vraagbericht, een zogenaamde vervolgvraag. In de praktijk spoort dit met de wens van een gebruiker controle te kunnen uitoefenen op de objecten die in een lijst wordt getoond en niet lang te hoeven wachten op een complete verzameling resultaten waar hij toch niet veel mee blijkt te kunnen.

De systematiek van het werken met vervolgvragen heeft bovendien als voordeel dat het antwoordende systeem elk vraagbericht op zich kan beantwoorden en niet hoeft 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.

Synchroon vraag/antwoord wordt vaak gebruikt voor het opvragen van een lijst met objecten of voor het ophalen van de details van één object.

Het antwoord op een asynchrone vraag kan bestaan uit een groot aantal objecten. Bij asynchroon berichtenverkeer is er geen interactie met een gebruiker en moeten alle gevraagde objecten in één keer worden teruggestuurd. Het in één bericht versturen van al deze objecten kan leiden tot problemen:

1. Het bericht wordt te groot voor de berichtverwerkende software.
2. Als er een fout optreedt bij de verzending of de verwerking is, dient het hele bericht opnieuw verstuurd te worden of moet hele bericht opniew verwerkt worden met alle risico's vandien (de transactie voor de verwerking van het bericht wordt te groot).

Deze problemen zijn eenvoudig op te lossen door niet één bericht als antwoord te sturen, maar een bericht per object. Het antwoord op een asynchrone vraag bestaat daarom uit nul of meer asynchrone antwoordberichten die elk één object van het gevraagde entiteittype bevatten. Het is voor de vragende partij wel nuttig om te weten of er geen antwoordbericht is zoekgeraakt en of het laatste object al is ontvangen. Het asynchrone antwoordbericht bevat stuurgegevens, waarmee het vragende systeem kan nagaan of het alle antwoordberichten heeft ontvangen.

Het is ook mogelijk een verzameling asynchrone antwoordberichten te versturen zonder dat een vraag is gesteld. Dit mechanisme kan bijvoorbeeld gebruikt worden voor het ontladen van een database.

Voor antwoordberichten legt 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 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.
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.

In vraag- en antwoordberichten wordt het berichtstuurgegeven *entiteittype* gevuld met het entiteittype van het object waarop het bericht betrekking heeft en wordt het berichtstuurgegeven *functie* weggelaten.

6.1 Stuurgegevens voor vraagberichten

Als gegevens gevraagd worden, dan moet het antwoordende systeem weten waar precies om wordt gevraagd. Het meest complex hierbij zijn de selectiecriteria (om welke objecten wordt gevraagd) en de scope (welke gegevens moeten worden teruggegeven). De selectiecriteria en de scope worden dan ook niet gespecificeerd via stuurgegevens maar in de rest van body van het vraagbericht. De paragrafen 6.3.1 respectievelijk 6.3.3 geven hier de regels voor. In deze paragraaf worden de stuurgegevens in het vraagbericht besproken.

- *sortering*
  Het stuurgegeven *sortering* geeft aan volgens welke sortering de objecten teruggegeven moeten worden. Het waardebereik van *sortering* is 0 – 99. De waarde 0 geeft aan dat het vragende systeem niet geïnteresseerd is in de volgorde. Als het stuurgegeven *sortering* de waarde nul heeft, dan mag het antwoordende systeem de objecten in een door het antwoordende systeem zelf te bepalen volgorde teruggeven.
  Het sectormodel dient de sorteringen groter dan nul voor elk fundamenteel en tabel entiteittype te specificeren door per sortering de elementen en relaties van het fundamentele entiteittype op te sommen waarop achtereenvolgens oplopend of aflopend gesorteerd wordt. Indien een relatie onderdeel is van de sortering, dan wordt voor die relatie de sorteervolgorde gespecificeerd door aan te geven op welke elementen en relaties van de relatie en van de gerelateerde entiteit oplopend of aflopend wordt gesorteerd. Bij het entiteittype *persoon* zou bijvoorbeeld als sortering 1 gespecificeerd kunnen worden dat eerst aflopend wordt gesorteerd op geboortedatum en daarbinnen oplopend op geslachtsnaam en voorletters. Sortering 2 zou kunnen zijn oplopend op achtereenvolgens postcode, huisnummer van het verblijfsadres en vervolgens op geslachtsnaam en voorletters.

- *indicatorVervolgvraag*
  Indien een antwoordbericht niet alle objecten heeft teruggeven die aan de selectiecriteria voldoen en het vragende systeem meer objecten wil ontvangen, dan kan een vervolgvraag worden gesteld door een nieuw vraagbericht te versturen. De waarde `true` voor het stuurgegeven *indicatorVervolgvraag* in een vraagbericht geeft aan dat het om een vervolgvraag gaat en de waarde `false`, dat het gaat het om een nieuwe vraag. In paragraaf 6.3.4 staat hoe in geval van een vervolgvraag wordt aangegeven bij welk object in de selectie gestart moet worden.

- *indicatorHistorisch*
  StUF ondersteunt het opvragen van actuele en historische gegevens. Het stuurgegeven *indicatorHistorisch* in een vraagbericht specificeert of en hoe de vraagsteller historische gegevens wil ontvangen. De *indicatorHistorisch* heeft drie mogelijke waarden:
  'N' geen historie: het antwoordende systeem mag alleen actuele gegevens teruggeven.
  'E' historie op entiteitsniveau
  'G' historie op groepsniveau
  Als de *indicatorHistorisch* niet aanwezig is, dan dient het antwoordende systeem uit te gaan van de waarde 'N'. In paragraaf 6.4.3 wordt nader ingegaan op het opnemen van historische gegevens in een antwoordbericht.

- *maximumAantal*
  Met het stuurgegeven *maximumAantal* kan worden aangegeven hoeveel objecten er maximaal mogen worden teruggestuurd. Zeker bij synchrone vraagberichten is het in het kader van de performance van belang om *maximumAantal* niet te groot te kiezen, bijvoorbeeld altijd kleiner dan 100 of gelijk aan het aantal objecten 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, want bij asynchrone vraag- en antwoordberichten kan de gebruiker niet tussentijds ingrijpen. Als het stuurgegeven *maximumAantal* niet aanwezig is, dan dient een defaultwaarde van 100 te worden gebruikt. In het sectormodel mag per type vraagbericht voor een entiteittype een andere defaultwaarde worden gespecificeerd.

- *indicatorAfnemerindicatie*
  In sommige gevallen wil het vragende systeem niet alleen objecten opvragen, maar ook zeker stellen dat in de toekomst mutaties in die objecten worden doorgegeven door middel van Lk01-kennisgevingberichten. Dit kan gespecificeerd door het stuurgegeven *indicatorAfnemerindicatie* met als waarde `true` in het vraagbericht op te nemen. Als de *indicatorAfnemerindicatie* ontbreekt of de waarde `false` heeft, dan hoeft het antwoordende systeem geen afnemerindicaties te plaatsen. Bij de bespreking van het stuurgegeven *indicatorAfnemerindicatie* in het antwoordbericht wordt aangegeven hoe gereageerd wordt op een waarde `true` voor *indicatorAfnemerindicatie* in het vraagbericht.

- *indicatorAantal*
  Het opvragen van het aantal voorkomens dat bij benadering aan de selectiecriteria voldoet kan in een synchroon vraagbericht worden gespecificeerd door het stuurgegeven *indicatorAantal* met als waarde `true` op te nemen. Bij de bespreking van het stuurgegeven *aantalVoorkomens* in een antwoordbericht wordt gespecificeerd wat er in

het antwoordbericht gedaan moet worden als *indicatorAantal* `true` is. Als *indicatorAantal* ontbreekt of de waarde `false` heeft, dan hoeft het aantal voorkomens niet te worden teruggegeven. In een asynchroon vraagbericht mag dit stuurgegeven niet voorkomen.

De stuurgegevens van een vraagbericht zijn kinderen van het element `<parametersVraag>` dat als eerste opgenomen in de body van een StUF-bericht. Het element `<parametersVraag>` bevat maximaal zes elementen in de volgende structuur:

```
<parametersVraag>
     <sortering>...</sortering>
     <indicatorVervolgvraag>...</indicatorVervolgvraag>
     <indicatorHistorisch>...</indicatorHistorisch>
     <maximumAantal>...</maximumAantal>
     <indicatorAfnemerindicatie>...</indicatorAfnemerindicatie>
     <indicatorAantal>...</indicatorAantal>
</parametersVraag>
```

De structuur van `<parametersVraag>` is gedefinieerd in het complexType `<ParametersVraag>` in [StUFXSD].

Tabel 6.1 geeft aan welke stuurgegevens voor een vraagbericht voorkomen in welke berichttypen. V wil zeggen verplicht, O wil zeggen optioneel en – wil zeggen mag niet voorkomen.

	Lv01	**Lv02**
sortering	V	V
indicatorVervolgvraag	V	V
indicatorHistorisch	O	O
maximumAantal	O	O
indicatorAfnemerindicatie	O	O
indicatorAantal	O	–

Tabel 6.1. Het voorkomen van de vraagbericht stuurgegevens binnen de verschillende berichttypen.

6.2 Stuurgegevens voor antwoordberichten

Bij het geven van een antwoord is het niet altijd voldoende om alleen de gevraagde objecten terug te geven, maar is nog aanvullende informatie gewenst. Via de stuurgegevens in de body van het antwoordbericht wordt deze aanvullende informatie verstrekt. Deze paragraaf behandelt de stuurgegevens voor antwoordberichten.

- *indicatorVervolgvraag*
  Een antwoordbericht hoeft niet alle objecten te bevatten die aan de selectiecriteria voldoen. Dit wordt in het antwoordbericht aangegeven met het stuurgegeven *indicatorVervolgvraag*. De waarde `true` geeft aan dat er nog meer objecten zijn die aan de selectiecriteria voldoen. De waarde `false` geeft aan dat alle objecten zijn verstuurd. In een synchroon antwoordbericht is *indicatorVervolgvraag* verplicht. Bij asynchrone antwoordberichten mag het alleen in het laatste antwoordbericht worden opgenomen en daar is *indicatorVervolgvraag* verplicht.

- *indicatorAfnemerindicatie*
  Als in het vraagbericht het stuurgegeven *indicatorAfnemerindicatie* de waarde `true` heeft, dan vraagt het vragende systeem, of het antwoordende systeem afnemerindicaties plaatst voor de teruggegeven objecten. Het antwoordende systeem is niet verplicht dit te doen en kan in het stuurgegeven *indicatorAfnemerindicatie* aangeven of al dan niet afnemerindicaties zijn geplaatst. De waarde `true` voor *indicatorAfnemerindicatie* geeft aan, dat het antwoordende systeem voor de teruggegeven objecten afnemerindicaties heeft geplaatst namens het vragende systeem. Als *indicatorAfnemerindicatie* ontbreekt of de waarde `false` heeft, dan zijn er geen afnemerindicaties geplaatst.

- *aantalVoorkomens*
  Als in een synchroon vraagbericht het stuurgegeven *indicatorAantal* `true` is, dan dient in de stuurgegevens van het antwoordbericht *aantalVoorkomens* gevuld te worden. Als alle gevraagde objecten in het antwoordbericht kunnen worden teruggegeven (*indicatorVervolgvraag* `false`), dan wordt *aantalVoorkomens* gevuld met het aantal objecten in het bericht. *aantalVoorkomens* wordt dan dus niet bepaald door een telling in de database, maar door een telling in het bericht.
  Als niet alle objecten in het bericht kunnen worden opgenomen (*indicatorVervolgvraag* `true`), dan wordt *aantalVoorkomens* gevuld met aantal objecten dat voldoet aan de gespecificeerde selectiecriteria, voorzover deze selectiecriteria zijn gedefinieerd binnen de opgegeven sortering. Met selectiecriteria buiten de opgegeven sortering hoeft bij het bepalen van het aantal voorkomens geen rekening gehouden te worden. Als een dergelijk selectiecriterium wordt genegeerd, dan wordt in de body van het antwoordbericht het element `<melding>` opgenomen gevuld met 'StUF0003: selectiecriteria buiten sortering genegeerd bij bepalen aantal voorkomens'.
  Het *aantalVoorkomens* geeft slechts bij benadering het aantal dat kan worden teruggegeven. Naast het niet meenemen van selectiecriteria buiten de sortering kunnen ook autorisatieregels bijvoorbeeld leiden tot een ander aantal.
  Als een antwoordend systeem niet de resources beschikbaar heeft voor het tellen van het aantal objecten, dan mag het stuurgegevens *aantalVoorkomens* weggelaten worden. In dat geval dient wel in de body van het antwoordbericht het element `<melding>` gevuld te worden met 'StUF0005: bepalen aantalVoorkomens vergt te veel resources'.

- *sequenceNumber*
  Asynchrone antwoordberichten bevatten één object per bericht, waardoor een antwoord op een asynchroon vraagbericht kan bestaan uit een groot aantal berichten. Het is voor de vragende partij nuttig om te weten of er geen antwoordbericht is zoekgeraakt en of het laatste object al is ontvangen. Het in asynchrone antwoordberichten verplichte stuurgegeven *sequenceNumber* bevat het volgnummer van asynchrone antwoordbericht in de verzameling antwoordberichten die het antwoord vormen op het asynchrone vraagbericht. Het eerste asynchrone antwoordbericht heeft als *sequenceNumber* 1. Voor elk volgend asynchroon antwoordbericht wordt het *sequenceNumber* met 1 opgehoogd. Als het vragende systeem constateert dat een *sequenceNumber* niet precies één groter is dan het *sequenceNumber* van het onmiddellijk ervoor ontvangen antwoordbericht, dan zijn er antwoordberichten zoekgeraakt.
  In synchrone antwoordberichten mag dit stuurgegeven niet voorkomen.

- *indicatorLaatsteBericht*
  Asynchrone antwoordberichten bevatten één object per bericht, waardoor een antwoord op een asynchroon vraagbericht kan bestaan uit een groot aantal berichten. Het is voor de vragende partij nuttig om te weten of er geen antwoordbericht is zoekgeraakt en of het laatste object al is ontvangen. Het stuurgegeven *indicatorLaatsteBericht* geeft in een asynchroon antwoordbericht met de waarde `true` aan dat dit asynchroon antwoordbericht het laatste object bevat dat als antwoord op het vraagbericht zal worden verzonden. Als *indicatorLaatsteBericht* ontbreekt of de waarde `false` heeft, dan dient de vragende partij nog meer berichten te verwachten als antwoord op zijn vraag.
  Het kan gaan om het laatste bericht, omdat het gespecificeerde *maximumAantal* is bereikt of omdat er geen objecten meer zijn die aan de selectiecriteria voldoen. Dit blijkt uit de waarde van het stuurgegeven *indicatorVervolgvraag*. Als het antwoord op een asynchrone vraag bestaat uit slechts één object, dan moet in dat ene antwoordbericht *indicatorLaatsteBericht* met de waarde `true` worden opgenomen.
  In synchrone antwoordberichten mag dit stuurgegeven niet voorkomen.

De stuurgegevens van een antwoordbericht zijn kinderen van het element `<parametersAntwoord>` dat als eerste opgenomen in de body van een StUF-bericht. Het element `<parametersAntwoord>` bevat maximaal zes verschillende elementen in de volgende structuur:

```
<parametersAntwoord>
     <indicatorVervolgvraag>...</indicatorVervolgvraag>
     <indicatorAfnemerindicatie>...</indicatorAfnemerindicatie>
     <aantalVoorkomens>...</aantalVoorkomens>
     <sequenceNumber>...</sequenceNumber>
     <indicatorLaatste>...</indicatorLaatste>
</parametersAntwoord>
```

De structuur van <parametersAntwoord> is gedefinieerd in het complexType <ParametersAntwoord> in [StUFXSD].

Tabel 6.2 geeft aan welke stuurgegevens voor een antwoordbericht voorkomen in welke berichttypen. V wil zeggen verplicht, O wil zeggen optioneel en – wil zeggen mag niet voorkomen.

	La01	**La02**
indicatorVervolgvraag	V	V
indicatorAfnemerindicatie	O	O
aantalVoorkomens	O	O
sequenceNumber	-	V
indicatorLaatsteBericht	-	O

Tabel 6.2. Het voorkomen van de antwoordbericht stuurgegevens binnen de verschillende berichttypen.

6.3 Regels voor vraagberichten

Bij het versturen van een vraagbericht kan het vragende systeem aangeven van welke objecten, in welke volgorde het welke gegevens wil ontvangen. In de berichtstuurgegevens wordt door middel van het stuurgegeven *entiteittype* gespecificeerd om welk type objecten het gaat en in de vraagbericht stuurgegevens wordt door middel van het stuurgegeven *sortering* aangegeven in welke volgorde de objecten worden verwacht. In de body van het vraagbericht wordt door middel van selectiecriteria gedefinieerd welke objecten worden gevraagd. 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. Het specificeren van de selectiecriteria wordt behandeld in paragraaf 6.3.1. Daarnaast kunnen de gevraagde objecten ook worden gespecificeerd door middel van sleutels. Dit wordt behandeld in paragraaf 6.3.2. Het specificeren van de gevraagde gegevens in de body van het vraagbericht wordt behandeld in paragraaf 6.3.3. Het stellen van een vervolgvraag behandeld in paragraaf 6.3.4. In paragraaf tenslotte wordt de algemene structuur van een vraagbericht behandeld en de foutafhandeling.

6.3.1 Het specificeren van selectiecriteria

Selecties kunnen op willekeurige elementen van een object worden gedefinieerd. 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’. Er is voor dit simpele mechanisme gekozen om de implementatie van selecties niet te complex te maken en om de responstijd bij selecties te kunnen garanderen. Daarnaast is opnemen van mechanismen als SQL en XQuery binnen StUF strijdig met het uitgangspunt dat introductie van StUF niet tot grote aanpassingen in bestaande systemen mag leiden. Als het door StUF ondersteunde 'en' selectiemechanisme niet verfijnd genoeg is, dan is het aan de vragende partij om extra selecties uit te voeren op de ontvangen objecten. Deze keuze maakt de implementatie van de standaard simpel en is afdoende voor synchrone vragen bedoeld om interactief een aantal objecten te tonen in een lijst of om precies één object te selecteren. Voor complexe vragen (meestal asynchrone vragen) biedt StUF nu mogelijk niet voldoende functionaliteit. De praktijk zal dit uitwijzen.

Een ‘gelijk’ selectie wordt gedefinieerd door als eerste element na <parametersVraag> in de body van het vraagbericht het element <gelijk> op te nemen. Dit element heeft als attribute entiteittype met het entiteittype uit de berichtstuurgegevens. De elementen van <gelijk> bevatten de selectiecriteria voor een 'gelijk' selectie. Wanneer binnen <gelijk> een element met lege inhoud en de attributes xsi:nil=”true” en StUF:noValue=”geenWaarde” of ”vastgesteldOnbekend” wordt opgenomen, dan betekent dit dat de waarde voor dat veld leeg respectievelijk vastgesteldOnbekend moet zijn. Het eerste 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. Naar de bijzondere waarden StUF:noValue=”geenWaarde” of ”vastgesteldOnbekend” kan alleen gevraagd worden in een gelijk-selectie.

Een ‘vanaf/tot-en-met’ selectie wordt gespecificeerd door na <parametersVraag> of na <gelijk> de elementen <vanaf> en <totEnMet> in de body van het vraagbericht op te nemen. De elementen <vanaf> en <totEnMet> hebben als attribute entiteittype met het entiteittype uit de berichtstuurgegevens. De elementen

van <vanaf> en <totEnMet> bevatten de selectiecriteria voor een 'vanaf/tot-en-met' selectie. Voor elementen binnen <vanaf> en <totEnMet> kunnen low of high values gespecificeerd worden door het element met lege inhoud en de attributes xsi:nil="true" en StUF:noValue="geenWaarde" op te nemen.

Het contentmodel voor de elementen <gelijk>, <vanaf> en <totEnMet> dient gelijk te zijn aan het contentmodel van het antwoordbericht voor dat entiteittype. Bij het zoeken van personen op een bepaald adres bevat <gelijk> bijvoorbeeld een element voor het relatie-entiteittype PERSOON.verblijft op.ADRES met daarin een element voor het entiteittype adres. Binnen het element voor adres worden de gewenste waarden voor de postcode en het huisnummer gespecificeerd. <vanaf> en <totEnMet> dienen precies dezelfde elementen als inhoud te bevatten. Een selectie mag een combinatie van 'gelijk' en 'vanaf/tot-en-met' criteria bevatten. Een element mag hetzij voorkomen binnen <gelijk> hetzij binnen <vanaf> en <totEnMet>, maar niet binnen beide.

Voor elk element dat niet is opgenomen binnen <gelijk> of <vanaf> en <totEnMet> zijn alle waarden toegestaan. Als de elementen <gelijk>, <vanaf> en <totEnMet> niet in het bericht voorkomen, dan dienen alle objecten te worden teruggegeven. Een element binnen <gelijk>, <vanaf> of <totEnMet> met StUF:noValue="waardeOnbekend" en een element binnen <vanaf> en <totEnMet> met StUF:noValue="vastgesteldOnbekend" dient in de verwerking genegeerd te worden.

Bij een gelijk-selectie kan worden aangegeven dat alleen een gedeelte van de waarde overeen hoeft te stemmen. Dit wordt gedaan door op een element binnen <gelijk> het attribute StUF:exact op te nemen met de waarde false. Wanneer het attribute StUF:exact ontbreekt of de waarde true heeft, dan voldoen alleen objecten, waarbij de waarde voor het selectiecriterium exact overeenkomt met de gespecificeerde waarde. Dit mechanisme is bijvoorbeeld nuttig, als je een persoon zoekt maar niet weet of hij Jansen heet of Janssen. Als voor het element <geslachtsnaam> binnen het element <gelijk> Jans en het attribute StUF:exact="false" wordt gespecificeerd, dan worden zowel de Jansen's als de Janssen's teruggeven. Binnen de elementen <vanaf> en <totEnMet> mag het attribute StUF:exact niet voorkomen op de selectiecriteria. Bij het definiëren van het vraagbericht in het sectormodel dient attribute ref="StUF:exact" te worden opgenomen op de elementen voor de selectiecriteria waarop met niet-exacte waarden geselecteerd mag worden. Het attribute StUF:exact is gedefinieerd in [StUFXSD].

Datums verdienen nog extra aandacht, omdat een datum niet volledig hoeft te zijn. Binnen <vanaf> en <totEnMet> mag voor een datumelement geen onvolledige datum worden gespecificeerd. Een object met een onvolledige datum voldoet alleen, als de selectiecriteria de hoogste en de laagste datum van het bereik van de onvolledige datum omvatten. Een datum met de dag onbekend voldoet dus alleen als alle datums in de maand voldoen aan de selectiecriteria. Bij een 'vanaf/tot-en-met' selectie worden objecten met jaarOnbekend dus nooit teruggegeven. Er is voor deze werkwijze gekozen, omdat een onvolledige datum een willekeurige waarde mag hebben. Alleen met deze specificatie is verzekerd dat er geen objecten worden teruggegeven met een door het vragende systeem niet verwachte waarde voor de datum.

Binnen <gelijk> mag voor een datumelement wel een onvolledige datum worden gespecificeerd. In dat geval voldoen alleen objecten met als waarde voor dat datumelement de gespecificeerde onvolledige waarde. Via dit mechanisme kunnen dus ook objecten met als waarde voor de datum 'jaarOnbekend' worden opgevraagd.

StUF schrijft niet voor dat een antwoordend systeem selecties op alle elementen van een object moet ondersteunen. Selecties op elementen uit de gevraagde sortering dienen in elk geval ondersteund te worden, mits er selectiecriteria worden gespecificeerd voor de opeenvolgende elementen in de sortering. Zodra er voor een element in de sortering geen selectiecriteria zijn gespecificeerd, dan hoeft een selectie op volgende elementen in de sortering niet meer te worden ondersteund. Selecties op elementen buiten de elementen van de sortering hoeft een antwoordend systeem ook niet te ondersteunen. Het antwoordende systeem wordt wel geacht te proberen binnen een redelijke responstijd met een antwoord te komen. Als dit niet lukt, dan wordt het antwoordende systeem geacht een antwoord te geven waarbij in elk geval wordt voldaan aan de selectiecriteria uit de sortering. Indien een selectiecriterium is gespecificeerd, dat niet ondersteund hoeft te worden en dat selectiecriterium is in het antwoord genegeerd, dan dient het element <melding> in de body van het antwoordbericht te worden opgenomen gevuld met 'StUF0001:' gevolgd door een spatie en de elementnamen gescheiden door spaties van de genegeerde selectiecriteria, bijvoorbeeld:

```
<melding>StUF0001: huisnummerToevoeging</melding>.
```

Bij het specificeren van de selectiecriteria kunnen zich de volgende foutsituaties voordoen:

Foutsituatie (= omschrijving)	Soort fout	Code	Plek	Details
<vanaf> en <totEnMet> bevatten niet dezelfde elementen	1, 2	StUF076	Client	De naam van het verschillend voorkomende element
Een element komt voor in zowel <gelijk> als <vanaf> en <totEnMet>	1, 2	StUF079	Client	De naam van het element
Het attribute StUF:exact is gebruikt op een element binnen <vanaf> of <totEnMet>	1, 2	StUF082	Client	De naam van het element
Onvolledige datum binnen <vanaf> of <totEnMet>	1, 2	StUF085	Client	De naam van het element met de onvolledige datum.

Tabel 6.3: Foutsituaties bij de specificatie van de selectiecriteria in vraagberichten

6.3.2 Het bevragen op sleutel

Soms zal het vragende systeem de sleutel van een gevraagd object 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 in het element <gelijk> een sleutel of in de elementen <vanaf> en <totEnMet> een range van sleutels mee te geven in plaats van selectiecriteria. Een sleutel worden als het attribute sleutelVerzendend, sleutelOntvangend of sleutelGegevensbeheer in het <gelijk>, <vanaf> of <totEnMet> element opgenomen. Het element heeft verder geen inhoud. Bij een selectie op sleutelwaarde wordt een eventueel gespecificeerde sortering genegeerd. De objecten met sleutels binnen de gevraagde range mogen in een willekeurige volgorde worden teruggeven.

Bij het selecteren op sleutel kunnen zich de volgende foutsituaties voordoen:

Foutsituatie (= omschrijving)	Soort fout	Code	Plek	Details
Meer dan één sleutel gespecificeerd als selectiecriterium	1, 2	StUF088	Client	
Een sleutel en andere elementen gespecificeerd als selectiecriterium	1, 2	StUF091	Client	
Ontvangend systeem registreert sleutel in het verzendende systeem niet	1, 2	StUF094	Server	

Tabel 6.4: Foutsituaties bij de specificatie van de selectiecriteria in vraagberichten

6.3.3 Het specificeren van de gevraagde gegevens

In de berichtbody kan worden gespecificeerd welke gegevens moeten worden teruggegeven. Er zijn drie mogelijkheden:

1. Geef alleen de kerngegevens van het gevraagde entiteittype terug.
2. Geef alle gegevens terug van het gevraagde entiteittype terug, dat wil zeggen alle elementen en relaties gedefinieerd in het schema voor een antwoordbericht en binnen de relaties en gerelateerden ook weer alle elementen en relaties gedefinieerd in het schema voor het antwoordbericht, enzovoorts.
3. Geef de gespecificeerde gegevens terug (hieronder wordt de wijze van specificeren toegelicht).

De gevraagde gegevens worden gespecificeerd in het element <scope>. Dit element wordt in de berichtbody opgenomen na de optionele elementen voor de selectiecriteria (kan dus als eerste element na <parametersVraag>). Het element <scope> bevat verplicht het attribute entiteittype met als waarde het entiteittype gedefinieerd in de berichtstuurgegevens. Als het element <scope> in het vraagbericht ontbreekt, dan mag het antwoordende systeem zelf bepalen welke gegevens worden teruggegeven.

Ten behoeve van de eerste twee mogelijkheden kent StUF binnen het element <scope> het attribute scope. Dit attribute heeft als mogelijke waarden 'kerngegevens' en 'alles'. Als het attribute scope wordt opgenomen mag het element <scope> verder geen inhoud hebben.

Ten behoeve van de derde mogelijkheid worden de gevraagde gegevens gespecificeerd binnen het element <scope> analoog aan de specificatie van de selectiecriteria. Het element <scope> bevat dan als elementen de gevraagde gegevens en relaties conform het contentmodel voor het antwoordbericht van het gevraagde entiteittype. Een gegeven wordt gevraagd door het op te nemen als een element met een lege inhoud en het attributes xsi:nil="true" en StUF:noValue="geenWaarde". Een relatie wordt gevraagd door het element voor de relatie op te nemen met het attribute entiteittype gevuld met de naam van het entiteittype voor de relatie. Hierbinnen worden weer de gevraagde gegevens en relaties opgenomen naast het verplichte element <gerelateerde> voor de gerelateerde. Ook binnen de gerelateerde worden op dezelfde wijze de gevraagde gegevens en relaties opgenomen.
Naar attributen kan niet gevraagd worden. Ook naar het element <tijdvakGeldigheid> kan niet gevraagd worden. Als het stuurgegeven *indicatorHistorisch* de waarde 'E' of 'G' heeft, hoeft naar de elementen <beginRelatie> en <eindRelatie> in relatie-entiteittypen niet gevraagd te worden. Ze worden dan sowieso geleverd. Als *indicatorHistorisch* de waarde 'N' heeft of niet is opgenomen, dan worden <beginRelatie> en <eindRelatie> alleen geleverd, als ze expliciet gevraagd worden.

Wanneer van een persoon de naamsgegevens, de geboortedatum en de adresgegevens worden gevraagd, heeft het element <scope> de volgende inhoud: het attribute entiteittype met de waarde van het berichtstuurgegeven *entiteittype*, de elementen voor de geslachtsnaam, de voorvoegsels, de voorletters en de geboortedatum met een lege inhoud en de attributes xsi:nil="true" en StUF:noValue="geenWaarde". Daarnaast bevat <scope> het element voor het relatie-entiteittype PERSOON.verblijft op.ADRES. Dit element bevat alleen het element <gerelateerde entiteittype="ADR"> met elementen met een lege inhoud en de attributes xsi:nil="true" en StUF:noValue="geenWaarde" voor de gevraagde adresgegevens.

Een extra aandachtspunt bij het maken van een sectormodel zijn de relaties, waarbij voor de gerelateerde een <choice> is gedefinieerd. Deze relaties dienen bij het definiëren van het vraagbericht in het element <scope> te worden opgenomen met een kardinaliteit gelijk aan het aantal keuzen binnen de <choice>. Op deze manier kan in het vraagbericht voor elk type gerelateerde worden gespecificeerd welke gegevens moeten worden teruggegeven. Als er voor een gerelateerde binnen de <choice> geen relatie is opgenomen met die gerelateerde, dan mogen in het antwoord relaties met die gerelateerde niet worden teruggegeven.

Bij het specificeren van de gevraagde gegevens kunnen zich de volgende foutsituaties voordoen:

Foutsituatie (= omschrijving)	**Soort fout**	**Code**	**Plek**	**Details**
Zowel het attribute scope als een inhoud gespecificeerd voor het element <scope>	1, 2	StUF097	Client	
Een gerelateerde uit een choice komt dubbel voor binnen het element <scope>	1, 2	StUF100	Client	De elementnaam van de relatie gevolgd door een spatie en de elementnaam van de gerelateerde

Tabel 6.5: Foutsituaties bij de specificatie van de selectiecriteria in vraagberichten

6.3.4 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 bespreking van het stuurgegeven *indicatorVervolgvraag* in vraag- en antwoordberichten hebben we gezien, dat niet altijd alle objecten die aan de selectiecriteria voldoen worden teruggegeven. Met een vraagbericht met de waarde true in het stuurgegeven *indicatorVervolgvraag* kan om de volgende objecten gevraagd worden. Een dergelijk vraagbericht zullen we hierna met vervolgvraag aanduiden.

StUF heeft als uitgangspunt, dat berichten onafhankelijk van elkaar verwerkt moeten kunnen worden. Het antwoordende systeem hoeft daarom geen informatie meer te hebben over het eerder verzonden antwoord. In een vervolgvraag dient daarom te worden gespecificeerd wat het laatst ontvangen object is. Dit wordt gedaan door na het optionele element

<scope> (kan dus als eerste element na <parametersVraag>) het element <start> op te nemen. Het element heeft als kind het element voor het laatste object uit het antwoordbericht met *indicatorVervolgvraag* `true`. Het laatste object uit het antwoordbericht wordt in het vervolg ook wel startobject genoemd.

Indien dit object als attribute `sleutelVerzendend` bevatte, dan dient de waarde voor `sleutelVerzendend` in het attribute `sleutelOntvangend` te worden opgenomen. Het attribute `sleutelVerzendend` wordt niet opgenomen, tenzij het laatste object in het antwoordbericht `sleutelOntvangend` bevatte, want dan wordt de waarde voor `sleutelOntvangend` opgenomen in het attribute `sleutelVerzendend` en wordt `sleutelOntvangend` niet opgenomen, tenzij het laatste object in het antwoordbericht `sleutelVerzendend` bevatte. Als het laatste object in het antwoordbericht het attribute `sleutelGegevensbeheer` bevatte, dan wordt het attribute `sleutelGegevensbeheer` opgenomen in het startobject in het vraagbericht.

Bij het specificeren van een vervolgvraag kan zich de volgende foutsituatie voordoen:

Foutsituatie (= omschrijving)	**Soort fout**	**Code**	**Plek**	**Details**
IndicatorVervolgvraag is true, maar het element <start> ontbreekt	1, 2	StUF103	Client	

Tabel 6.6: Foutsituaties bij de specificatie van de selectiecriteria in vraagberichten

6.4 Regels voor antwoordberichten

Deze paragraaf behandelt het vullen van antwoordberichten. In paragraaf 6.4.1 wordt allereerst ingegaan op het al dan niet opnemen van objecten in het antwoordbericht. Paragraaf 6.4.2 gaat in het opnemen van elementen voor attributen en relaties binnen een object zonder rekening te houden met historie. Paragraaf 6.4.3 behandelt het opnemen van historische gegevens voor een object gegeven de *indicatorHistorisch* in het vraagbericht. Paragraaf 6.4.4 tenslotte gaat nog kort in op de foutafhandeling. Alvorens dieper op het vullen van het antwoordbericht in te gaan, wordt eerst nog de structuur van het antwoordbericht besproken.

Soms kan een vraagbericht wel verwerkt worden, maar is niet aan alle verwachtingen van de vragende partij voldaan. Om dit te kunnen aangeven wordt het element <melding> met een kardinaliteit 0 tot oneindig gebruikt. Dit element is gedefinieerd in het StUF-schema ([StUFXSD]).Voor een paar gevallen definieert de StUF-standaard het gebruik van dit veld. In een sectormodel kunnen altijd extra meldingen gedefinieerd worden.

Binnen de body van een antwoordbericht wordt eerst het element met <parametersAntwoord> opgenomen. Daarna volgen de eventuele <melding> elementen. Als laatste worden in de body opgenomen nul of één <object> element (asynchroon antwoordbericht) of nul of meer <object> elementen (synchroon antwoordbericht).

6.4.1 Het opnemen van objecten in een antwoordbericht

Qua autorisatie zijn er drie niveau's te onderscheiden:

- *Entiteittypeniveau:* Het vragende systeem is niet geautoriseerd het entiteittype te bevragen. In dit geval wordt een Fo01- of Fo02-foutbericht teruggestuurd met als foutcode StUF052 (zie paragraaf 4.4.3).
- *Objectniveau:* Het vragende systeem is niet geautoriseerd voor een specifiek object. In dit geval wordt het object niet in het antwoordbericht opgenomen.
- *Attribuut/relatie-niveau:* Het vragende systeem is geautoriseerd voor het object, maar niet geautoriseerd voor alle attributen en relaties van het object. Hieronder wordt dit verder besproken.

Een object wordt alleen in een antwoordbericht opgenomen, als aan één van de twee volgende voorwaarden is voldaan:

- Voor minimaal één van de kerngegevens is een waarde bekend.
- Het attribute `sleutelOntvangend` is bekend en minimaal één van de gevraagde gegevens heeft een geldige waarde of de waarde 'vastgesteldOnbekend'.

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 adressen opgenomen

met als attribute een gevulde `sleutelOntvangend` en als enig element `<locatieOmschrijving>` met een lege inhoud zonder dat de waarde 'vastgesteldOnbekend' is.

Een object wordt als een element `<object>` als kind in het `<body>` element opgenomen. Het `<object>` element heeft een verplicht attribute `entiteittype` gevuld met het entiteittype van het object. De attributes `sleutelVerzendend`, `sleutelOntvangend` en `sleutelGegevensbeheer` mogen in het `<object>` element worden opgenomen. In een synchroon antwoordbericht kan `<body>` meerdere `<object>` elementen bevatten. Het element `<body>` van een asynchroon antwoordbericht bevat altijd slechts één `<object>` element. Als er geen enkel object in het antwoordbericht kan worden opgenomen, dan wordt een antwoordbericht met een lege body (`<body/>` of `<body></body>`) als respons gezonden.

Meerdere objecten worden teruggegeven in de volgorde gespecificeerd in het stuurgegeven *sortering*. Als het stuurgegeven sortering de waarde 0 bevat of als er op sleutel bevraagd is (zie paragraaf 6.3.2), dan mag het antwoordende systeem zelf de volgorde van de objecten bepalen. Als het stuurgegeven *sortering* een sortering bevat die het antwoordende systeem niet ondersteunt, dan wordt een Fo01- of Fo02-foutbericht teruggestuurd met als foutcode StUF118 (zie ook paragraaf 6.4.4). Het is namelijk niet noodzakelijk dat een systeem alle in het sectormodel gespecificeerde sorteringen ondersteunt.

Per entiteittype en per soort vraagbericht (synchroon of asynchroon) kan het antwoordende systeem een maximum aantal terug te zenden objecten kennen. Indien meer objecten aan de selectiecriteria voldoen en het maximum aantal van het antwoordende systeem is kleiner dan aantal gespecificeerd in het stuurgegeven *maximumAantal* in het vraagbericht, dan zendt het antwoordende systeem het eigen maximale aantal objecten. In een synchroon antwoordbericht wordt het element `<melding>` dan gevuld met 'StUF0002: Meer objecten gevraagd dan teruggegeven' en wordt het stuurgegeven *indicatorVervolgvraag* met `true` gevuld. In geval van een asynchrone vraag wordt in het laatste asynchrone antwoordbericht het element `<melding>` gevuld met 'StUF0002: Meer objecten gevraagd dan teruggegeven' en worden de stuurgegevens *indicatorVervolgvraag* en *indicatorLaatsteBericht* met `true` gevuld.

6.4.2 Het vullen van objecten in een antwoordbericht

In deze paragraaf wordt het vullen van elementen binnen een object besproken. De hier gegeven regels gelden zowel voor voorkomens van een object met actuele gegevens als voor voorkomens met historische gegevens. In de volgende paragraaf wordt nader ingegaan op het onderscheid tussen actuele en historische gegevens.

In een antwoordbericht verzonden naar aanleiding van een vraagbericht moeten worden opgenomen de gegevens waarom in het element `<scope>` in het vraagbericht is gevraagd. Als het element `<scope>` ontbreekt in het vraagbericht, dan mag het antwoordende systeem zelf bepalen welke gegevens worden teruggegeven. Voor asynchrone antwoordberichten zonder direct bijbehorend vraagbericht dient in het sectormodel gespecificeerd te worden welke gegevens in het bericht moeten worden opgenomen.

In de volgende gevallen kan cq mag het antwoordende systeem geen geldige waarde verstrekken:

- Het antwoordende systeem ondersteunt het gevraagde gegeven niet. In dat geval wordt het element teruggegeven met een lege inhoud en met de attributes `xsi:nil="true"` en `StUF:noValue="nietOndersteund"`.
- Het vragende systeem is niet geautoriseerd voor het gegeven. In dat geval wordt het element teruggegeven met een lege inhoud en met de attributes `xsi:nil="true"` en `StUF:noValue="nietGeautoriseerd"`.
- Het antwoordende systeem kent geen waarde voor het gegeven. In dat geval wordt het element teruggegeven met een lege inhoud en met de attributes `xsi:nil="true"` en `StUF:noValue="waardeOnbekend"`.

Deze regels dienen in bovenstaande volgorde te worden toegepast. Een relatie wordt alleen in een object opgenomen, als voor minimaal één van de kerngegevens van de gerelateerde een waarde bekend is of als het attribute `sleutelOntvangend` van de relatie of de gerelateerde bekend is.

In een antwoordbericht dient in een element voor het fundamentele entiteittype of tabelentiteittype waarop de vraag betrekking heeft het attribute `sleutelVerzendend` gevuld te worden met de sleutel in het antwoordende (=verzendende) systeem.

6.4.3 Het omgaan met historische gegevens

Indien in de stuurgegevens *indicatorHistorisch* de waarde 'N' heeft of niet voorkomt of indien voor geen van de gevraagde elementen in het sectormodel historie is gedefinieerd, dan worden alleen de actuele gegevens in het bericht opgenomen. Dat wil zeggen in het bericht worden één (asynchroon antwoordbericht) of meer (synchroon antwoordbericht) objecten van het in de berichtstuurgegevens gevraagde entiteittype opgenomen met de nu geldende gevraagde gegevens. Verder worden alle gevraagde relaties opgenomen waarvoor het element `<eindRelatie>` geen waarde heeft, of een waarde die in de toekomst ligt of waarbij het element `<eindRelatie>` niet voorkomt in het relatie-entiteittype in het sectormodel. In geen van de objecten 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 `<eindRelatie>`.

Indien in de stuurgegevens *indicatorHistorisch* de waarde 'E' of 'G' heeft, dan worden naast de actuele gegevens ook historische gegevens in het bericht opgenomen, als voor minstens één van de gevraagde elementen in het sectormodel historie is gedefinieerd en zo'n element in de loop van de tijd van waarde veranderd is. Hieronder wordt beschreven hoe binnen het `<object>` element de historische gegevens gevuld moet worden.

Als het stuurgegeven *indicatorHistorisch* de waarde 'E' heeft, dan bevat `<object>` allereerst de actuele waarden voor de attributen van het entiteittype, de relaties inclusief historische relaties en het `<tijdvakGeldigheid>` op entiteitniveau. Het element `<beginGeldigheid>` wordt gevuld met het tijdstip van de wijziging die het kortst geleden is van een element waarvoor in het sectormodel historie is gedefinieerd. Het element `<eindGeldigheid>` krijgt een lege inhoud en de attributes `xsi:nil="true"` en `StUF:noValue="geenWaarde"`. Binnen groepen gedefinieerde `<tijdvakGeldigheid>` elementen worden niet opgenomen.

De relaties worden gegroepeerd naar entiteittype. In relatie-entiteiten waarvoor in het sectormodel historie is gedefinieerd, is `<tijdvakRelatie>` verplicht. Relaties met een gevulde `<eindRelatie>` worden aflopend gesorteerd naar `<eindRelatie>` opgenomen na de actuele relaties voor een entiteittype. Wanneer een relatie-entiteit historie heeft (bijv. een huwelijk) voor een of meer van zijn elementen, dan bevat een relatie-entiteit na het element voor de gerelateerde en na de elementen voor de eigen attributen met de actuele gegevens en relaties ook één of meer `<historie>` elementen met de historische gegevens. Gerelateerde entiteiten worden altijd met alleen de actuele waarden opgenomen. Dit geldt ook voor historische relaties, waarvan de gerelateerde is gewijzigd, nadat de relatie beëindigd is.

Na de eventuele relaties worden de historische waarden voor de attributen opgenomen door één of meer elementen `<historie>` op te nemen. Binnen een element `<historie>` worden als elementen uitsluitend de attributen (niet de relaties!) van het entiteittype opgenomen die op `<eindGeldigheid>` van het historische voorkomen van waarde zijn veranderd. Dit geldt ook voor elementen in een groep met een eigen `<tijdvakGeldigheid>`. Al deze elementen krijgen als waarde de tot `<eindGeldigheid>` van het historische voorkomen geldige waarde. Daarnaast wordt het element `<tijdvakGeldigheid>` op entiteitniveau opgenomen. Hierbinnen wordt `<beginGeldigheid>` gevuld met het tijdstip van de wijziging die de kortste tijd voor eindGeldigheid heeft plaatsgevonden van een element waarvoor in het sectormodel historie is gedefinieerd. Geen enkel op groepniveau gedefinieerd `<tijdvakGeldigheid>` wordt opgenomen. De `<historie>` elementen worden aflopend gesorteerd naar het element `<beginGeldigheid>` in `<tijdvakGeldigheid>`.

De specificatie dat alleen van waarde veranderende attributen mogen worden opgenomen in `<historie>` elementen, heeft twee belangrijke consequenties:

1. Een attribuut dat gedurende de hele levenscyclus van een object niet van waarde verandert, komt alleen voor als element in het actuele voorkomen en niet binnen de `<historie>` elementen.
2. In een sectormodel binnen een element `<historie>` mag geen enkel element verplicht zijn met uitzondering van `tijdvakGeldigheid`.

Onderstaand voorbeeld illustreert het opnemen van historie als *indicatorHistorisch* 'E' is.

```
<object entiteittype="PRS">
   <geslachtsnaam>Berg</geslachtsnaam>
   <voorvoegsels>van den></voorvoegsels>
   <voorletters>JP</voorletters>
   <burgerlijkeStaat>gehuwd</burgerlijkeStaat>
   <geboortedatum>19770708</geboortedatum>
```

```
   <tijdvakGeldigheid>
      <beginGeldigheid>20050423</beginGeldigheid>
      <eindGeldigheid xsi:nil="true" StUF:noValue="geenWaarde"/>
   </tijdvakGeldigheid>
   <verblijfsadres entiteittype="PRSADRVBL">
      <gerelateerde entiteittype="ADR">
         <straatnaam>Donk</straatnaam>
         <postcode>5612DF</postcode>
         <woonplaats>Eindhoven</woonplaats>
         <huisnummer>24</huisnummer>
      </gerelateerde>
      <tijdvakRelatie>
         <beginRelatie>20050601</beginRelatie>
         <eindRelatie xsi:nil="true" StUF:noValue="geenWaarde"/>
      </tijdvakRelatie>
   </verblijfsadres>
   <verblijfsadres entiteittype="PRSADRVBL">
      <gerelateerde entiteittype="ADR">
         <straatnaam>Vallestap</straatnaam>
         <postcode>5654BX</postcode>
         <woonplaats>Nuenen</woonplaats>
         <huisnummer>33</huisnummer>
      </gerelateerde>
      <tijdvakRelatie>
         <beginRelatie>19991108</beginRelatie>
         <eindRelatie>20050601</eindRelatie>
      </tijdvakRelatie>
   </verblijfsadres>
   <verblijfsadres entiteittype="PRSADRVBL">
      <gerelateerde entiteittype="ADR">
         <straatnaam>Beatrixstraat</straatnaam>
         <postcode>5686AF</postcode>
         <woonplaats>Nuenen</woonplaats>
         <huisnummer>105</huisnummer>
      </gerelateerde>
      <tijdvakRelatie>
         <beginRelatie>19770708</beginRelatie>
         <eindRelatie>19991108</eindRelatie>
      </tijdvakRelatie>
   </verblijfsadres>
   <historie>
      <burgerlijkeStaat>ongehuwd</burgerlijkeStaat>
      <tijdvakGeldigheid>
         <beginGeldigheid>20010903</beginGeldigheid>
         <eindGeldigheid>20050423</eindGeldigheid>
      </tijdvakGeldigheid>
   </historie>
   <historie>
      <geslachtsnaam>Poepenstaart</geslachtsnaam>
      <voorvoegsels xsi:nil="true" StUF:noValue="geenWaarde"/>
      <tijdvakGeldigheid>
         <beginGeldigheid>19770708</beginGeldigheid>
         <eindGeldigheid>20010903</eindGeldigheid>
      </tijdvakGeldigheid>
   </historie>
</object>
```

Het bovenstaande voorbeeld gaat over ene JP van den Berg geboren op 8-7-1977. De niet veranderde geboortedatum staat bij de actuele gegevens. Op 23-4-2005 is JP van den Berg getrouwd, waardoor zijn burgerlijke staat veranderde en op 3-9-2001 blijkt zijn naam gewijzigd te zijn van 'Poepenstaart' in 'van den Berg'. Deze persoon heeft tot 8-11-1999 gewoond in de Beatrixstraat is toen verhuisd naar de Vallestap en is op 1-6-2005 weer verhuisd naar Donk 24 in Eindhoven, waar hij nu nog woont.

Als het stuurgegeven *indicatorHistorisch* de waarde 'G' heeft, dan worden historische gegevens voor de attributen op een andere manier opgenomen. Historische relaties worden op precies dezelfde manier opgenomen als wanneer het stuurgegeven *indicatorHistorisch* de waarde 'E' heeft.

Voor alle groepen die in het sectormodel een `<tijdvakGeldigheid>` bevatten, worden de historische gegevens binnen die groep opgenomen door middel van `<historie>` elementen. De `<historie>` elementen zijn de laatste elementen in de groep en ze worden aflopend gesorteerd naar het element `<beginGeldigheid>` erbinnen. Alle gevraagde elementen worden in elk `<historie>` element opgenomen en niet alleen de elementen die op

<eindGeldigheid> wijzigen. Voor alle elementen die geen lid zijn van een groep met een <tijdvakGeldigheid> en waarvoor in het sectormodel historie is gedefinieerd, wordt de historie opgenomen binnen <historie> elementen op entiteitniveau per tijdstip van de wijziging. Ook hier geldt dat in elk <historie> element alle gevraagde gegevens worden opgenomen.

Het is toegestaan om de *indicatorHistorisch* de waarde 'G' te geven, ook al komt er geen enkele groep met een <tijdvakGeldigheid> voor binnen het entiteittype. In dat geval worden alle gevraagde gegevens opgenomen in elk <historie> element op entiteitniveau en niet alleen de op <eindGeldigheid> veranderende gegevens.

Uitgaande van een groep van de naamsgegevens met daarbinnen een <tijdvakGeldigheid> ziet het voorbeeld er als volgt uit, als *indicatorHistorisch* de waarde 'G' heeft. Omdat er voor de relaties niets veranderd zijn deze in het voorbeeld weggelaten.

```
<object entiteittype=”PRS”>
   <naamGrp>
      <geslachtsnaam>Berg</geslachtsnaam>
      <voorvoegsels>van den</voorvoegsels>
      <voorletters>JP</voorletters>
      <tijdvakGeldigheid>
         <beginGeldigheid>20010903</beginGeldigheid>
         <eindGeldigheid xsi:nil=”true” StUF:noValue=”geenWaarde”/>
      </tijdvakGeldigheid>
      <historie>
         <geslachtsnaam>Poepenstaart</geslachtsnaam>
         <voorvoegsels xsi:nil=”true” StUF:noValue=”geenWaarde”/>
         <voorletters>JP</voorletters>
         <tijdvakGeldigheid>
            <beginGeldigheid>19770708</beginGeldigheid>
            <eindGeldigheid>20010903</eindGeldigheid>
         </tijdvakGeldigheid>
      </historie>
   </naamGrp>
   <burgerlijkeStaat>gehuwd</burgerlijkeStaat>
   <geboortedatum>19770708</geboortedatum>
   <tijdvakGeldigheid>
      <beginGeldigheid>20050423</beginGeldigheid>
      <eindGeldigheid xsi:nil=”true” StUF:noValue=”geenWaarde”/>
   </tijdvakGeldigheid>
   ...
   <historie>
      <burgerlijkeStaat>ongehuwd</burgerlijkeStaat>
      <geboortedatum>19770708</geboortedatum>
      <tijdvakGeldigheid>
         <beginGeldigheid>19770708</beginGeldigheid>
         <eindGeldigheid>20050423</eindGeldigheid>
      </tijdvakGeldigheid>
   </historie>
</object>
```

We zien nu binnen het element <naamGrp> de historie voor de naamsgegevens. Binnen deze groep worden ook de niet veranderde voorletters opnieuw opgenomen. Op entiteitniveau worden in <historie> de veranderde burgerlijke staat en de niet veranderde geboortedatum opgenomen.

Een antwoordend systeem hoeft het teruggeven van historische gegevens niet te ondersteunen. In dat geval wordt een Fo01- of Fo02-foutbericht teruggegeven zoals gedefinieerd in onderstaande tabel.

Foutsituatie (= omschrijving)	**Soort fout**	**Code**	**Plek**	**Details**
IndicatorHistorisch 'E' wordt niet ondersteund	1, 2	StUF106	Server	
IndicatorHistorisch 'G' wordt niet ondersteund	1, 2	StUF109	Server	

Tabel 6.7: Foutsituaties bij de specificatie van de selectiecriteria in vraagberichten

6.4.4 Foutafhandeling

Bij het afhandelen van vraagberichten zijn naast de situaties in paragraaf 4.4.3 en paragraaf 6.3 en naast de reeds hierboven besproken situaties (autorisatie, sortering en historie) nog de volgende foutsituaties onderkend:

- Indien het ontvangende systeem het afhandelen van asynchrone vragen niet ondersteunt, dan wordt een Fo01-foutbericht met foutcode StUF112 teruggezonden.
- Indien de gebruiker binnen het zendende systeem (zendende organisatie, zendende applicatie, zendende administratie en zendende gebruiker uit de berichtstuurgegevens) onbekend is in het antwoordende systeem en nodig voor autorisatie, dan wordt het foutbericht StUF115 teruggezonden.
- Indien beantwoording van een asynchrone vraag teveel systeemresources vraagt, dan zendt het antwoordende systeem een Fo01-foutbericht met foutcode StUF121. Door het zenden van een foutbericht met foutcode StUF121 kan het antwoordende systeem voorkomen dat het te zwaar belast wordt. Tevens geeft het antwoordende systeem hiermee aan het vragende systeem te kennen, dat de vraag waarschijnlijk beter op een andere manier gesteld kan worden.
  De voorkeur boven het zenden van dit foutbericht heeft het niet teruggeven van alle gevraagde objecten zoals beschreven in de laatste alinea van paragraaf 6.4.1.
- Indien de beantwoording van een synchrone vraag teveel systeemresources vergt en niet mogelijk is binnen de connection time out voor het onderliggende protocol, dan mag gereageerd worden zoals beschreven in paragraaf 4.4. Het heeft echter te voorkeur te reageren door niet alle gevraagde objecten terug te geven zoals beschreven in de laatste alinea van paragraaf 6.4.1.

Tabel 6.8 geeft een overzicht van de hierboven genoemde foutsituaties.

Foutsituatie (= omschrijving)	**Soort fout**	**Code**	**Plek**	**Details**
Antwoordend systeem ondersteunt afhandelen asynchrone vragen niet	1	StUF112	Server	
Gebruiker binnen zendend systeem onbekend in antwoordend systeem en nodig voor autorisatie	1, 2	StUF115	Server	
Antwoordend systeem ondersteunt gevraagde sortering niet	1, 2	StUF118	Server	
Beantwoording asynchrone vraag vergt teveel systeemresources	1	StUF121	Server	

Tabel 6.8: Overige foutsituaties bij het afhandelen van vraagberichten

7. Vrije berichten

De betekenis van kennisgeving- en synchronisatieberichten en ook de vraag/antwoordberichten wordt voor het grootste deel voorgeschreven door de StUF-standaard. De ontwerper van een sectormodel bepaalt primair de structuur van de objecten in deze berichten. De functionaliteit van de berichten schrijft de StUF-standaard voor. Dit kan omdat voor het doorgeven van gebeurtenissen, het synchroniseren van gegevens en het opvragen van gegevens in een database de functionaliteit heel beperkt is zolang deze zich beperkt tot objecten van één type, bijvoorbeeld personen of verblijfsobjecten.

In een service geöriënteerde architectuur is meer of andere functionaliteit nodig. Bij het doorgeven van een gebeurtenis als een verhuizing is de beperking tot één object bijvoorbeeld onwenselijk. In één bericht wil je de verhuizing van een heel gezin kunnen doorgeven. Hierbij wil je ook de mogelijkheid hebben om de aangever en de verhuizende personen en het oude en nieuwe adres als afzonderlijke objecten in het bericht op te nemen. Er is dan in plaats van een kennisgeving of een samengestelde kennisgeving een ander soort bericht nodig.

Bij het beschikbaar stellen van services op een basisregistratie zijn in een aantal gevallen specialistische services met simpele parameters eenvoudiger in het gebruik dan generieke services. Een generieke service biedt veel functionaliteit maar verlangt van de servicegebruiker dat deze de sortering, de gevraagde gegevens en de selectiecriteria definieert binnen niet altijd even simpele structuren. Voor een specialistische service hoeft de servicegebruiker veel minder te definiëren, maar heeft de gebruiker ook minder functionaliteit tot zijn beschikking, tenzij de serviceaanbieder een groot aantal specialistische services aanbiedt. Het bieden van veel specialistische services is vaak niet eenvoudiger dan het bieden van een paar generieke services met een wat complexer interface.

De StUF-standaard heeft voor het doorgeven van gebeurtenissen en het synchroniseren en opvragen van gegevens een aantal redelijk generieke services gedefinieerd. De StUF-standaard wil echter ook de implementatie van andere functionaliteit of van specialistische functionaliteit ondersteunen. Daartoe kent de StUF-standaard het zogenaamde vrije bericht. Vrije berichten zijn altijd onderdeel van een sectormodel. Het is de verantwoordelijkheid van de berichtontwerper om de functionaliteit van vrije berichten te definiëren. In de standaard is de beschrijving van de vrije berichten daarom veel beperkter dan de beschrijving van kennisgeving- en vraag/antwoordberichten.

7.1 Interactiepatronen en berichtcodes

De StUF-standaard kent voor vrije berichten de interactiepatronen notificatie en verzoek/respons in de varianten synchroon en asynchroon. Deze interactiepatronen worden gerealiseerd met vier verschillende berichtcodes:

- Di01: een op een webservice binnenkomend vrij bericht, waarop geen onmiddellijke respons wordt verwacht, maar alleen een bevestiging van ontvangst. Het Di01-bericht kan zowel gebruikt worden voor notificaties als voor asynchrone verzoek/respons interacties. Bij ontvangst dient een Di01-bericht door de webservice op dezelfde manier te worden afgehandeld als een asynchroon kennisgeving- of vraagbericht.
- Du01: De asynchrone respons op een ontvangen Di01-bericht conform het contract van de service. Op het verzenden van een Du01-bericht dient de zender van het Di01-bericht hetzelfde te reageren als op het verzenden van een La02-bericht als antwoord op een asynchroon Lv02-bericht.
- Di02 en Du02: een synchroon verzoek/respons paar met Di02 het verzoek en Du02 de respons die binnen de http time out conform het contract van de service wordt verwacht. De afhandeling van het Di02/Du02 paar is hetzelfde als de afhandeling van het synchrone vraag/antwoord paar Lv01/La01.

Voor de vrije berichten dient minimaal de in paragraaf 4.4.3 gedefinieerde foutafhandeling te worden ondersteund. Andere fouten kunnen worden afgehandeld door in het sectormodel extra foutsituaties voor het Fo01- respectievelijk Fo02-foutbericht te beschrijven of door hiervoor speciale vrije berichten als respons te definiëren.

7.2 Gereserveerde elementen en structuur van het vrije bericht

Op het hoogste niveau binnen de body van een vrij bericht mogen alleen in de StUF-standaard gedefinieerde elementen (gereserveerde elementen) of elementen voor een entiteittype uit het sectormodel voorkomen.

De StUF-standaard kent op het hoogste niveau in de body van een vrij bericht de volgende elementen met een door de standaard gedefinieerde betekenis of functionaliteit:

- 0 of meer elementen voor één entiteit van een entiteittype uit het sectormodel voor het doorgeven van gegevens van objecten van een entiteittype
- `<update>` met de structuur van de body van een kennisgeving voor het doorgeven van wijzigingen in gegevens

- <vraag> met de structuur van de body van een vraagbericht voor het opvragen van gegevens
- <zaakinfo> met een structuur gedefinieerd in het complexType <Zaakinfo> in het sectormodel Zaken voor het doorgeven van zaakinformatie.

De elementen voor entiteittypen uit het sectormodel zijn herkenbaar aan het attribute entiteittype. De naam van de elementen met een attribute entiteittype is vrij, opdat de functie van het element binnen het bericht kan worden gespecificeerd door middel van de naam van het element. In de volgende paragrafen worden de gereserveerde elementen en de structuur van het vrije bericht in meer detail besproken.

De berichtontwerper heeft hiermee de beschikking over een aantal structuren voor het definiëren van de gewenste functionaliteit. Waar mogelijk dienen deze structuren gebruikt te worden. Als de gewenste functionaliteit niet kan worden gerealiseerd met behulp van deze elementen, dan dient deze gerealiseerd te worden binnen het gereserveerde element <parameters>. Voor het teruggeven van meldingen in een responsbericht mag op het hoogste niveau in de body van een vrij bericht het element <melding> worden opgenomen zoals gedefinieerd in de [StUFXSD].

Een vrij bericht heeft daarmee de volgende structuur:

- 0 of 1 element <parameters> met een vrij te definiëren structuur
- 0 of meer elementen <melding> met in het sectormodel gedefinieerde meldingen als respons op een vrij verzoek
- 0 of meer elementen voor één entiteit van een entiteittype uit het sectormodel
- 0 of meer elementen <update>
- 0 of één element <vraag>
- 0 of 1 element <zaakinfo>.

In het StUF0300-schema is de structuur van het vrije bericht als complexType <VrijBericht> gedefinieerd zonder de elementen voor entiteiten van entiteittypen uit het sectormodel. Deze elementen kunnen niet in het complexType worden opgenomen, omdat ze een willekeurige naam mogen hebben. Via het extension mechanisme kunnen ze in een sectormodel aan <VrijBericht> of een kopie cq restriction ervan worden toegevoegd. Het element <zaakinfo> is opgenomen zonder type, omdat hier verwezen moet worden naar het sectormodel Zaken.

7.2.1 Het opnemen van losse gegevens en meldingen

Omdat de stuurgegevens niet uitgebreid mogen worden, moeten alle gegevens in het bericht die geen onderdeel zijn van een entiteittype uit het sectormodel – de zogenaamde parameters – worden opgenomen in de berichtbody. Hierboven is al de eis geformuleerd dat als kind-elementen van de body alleen elementen voor een entiteittype of gereserveerde elementen mogen worden opgenomen. Voor het opnemen van parameters binnen de berichtbody voorziet de StUF-standaard daarom in het gereserveerde element <parameters>. Dit element dient als eerste element in de body voor te komen. Het element <parameters> mag geen attributen hebben, maar verder is de inhoud van dit element volledig vrij.

Een vrij bericht kan meerdere objecten bevatten en ook per object kunnen extra parameters wenselijk zijn. Deze parameters kunnen aan een object worden meegegeven door als laatste element binnen het object het element <parameters> op te nemen. Ook hier geldt dat het element <parameters> geen attributen mag hebben, en dat de inhoud verder volledig vrij is.

Voor het doorgeven van een melding dat de verwerking van een vrij bericht niet geheel succesvol is verlopen kunnen nul of meer elementen <melding> in de body van het responsbericht worden opgenomen. Dit element is gedefinieerd in het StUF-schema ([StUFXSD]).

7.2.2 Elementen voor een entiteittype uit het sectormodel

De waarden van eigenschappen van in het sectormodel gedefinieerde entiteittypen dienen in het vrije bericht te worden opgenomen als elementen binnen het in het sectormodel gedefinieerde complexType voor dat entiteittype. Ze mogen niet als losse elementen in een vrij bericht worden opgenomen. Bij het specificeren van de postcode van het verblijfsadres van een persoon als selectiecriterium wordt deze postcode dus als volgt in het vrije bericht opgenomen:

```
<persoon entiteittype="PRS">
    <verblijftOpAdres entiteittype="PRSADRVBL">
        <gerelateerde entiteittype="ADR">
```

```
            <postcode>5672BJ</postcode>
        </gerelateerde>
    </verblijftOpAdres>
</persoon>
```

De reden hiervoor is dat de betekenis van zo'n element reeds in het sectormodel is gedefinieerd en daarom bij de definitie van het vrije bericht niet meer gespecificeerd hoeft te worden. Deze eis maakt hergebruik van delen van een parser mogelijk.

7.2.3 Het wijzigen van gegevens

Met een kennisgeving kunnen wijzigingen worden doorgegeven met als beperking dat de wijziging betrekking heeft op één object. Een vrij bericht kent deze beperking niet en kan wijzigingen in meerdere objecten doorgeven. De StUF-standaard biedt hiervoor met het gereserveerde element `<update>` de reeds voor een kennisgeving gedefinieerde functionaliteit. Als er meer dan één wijzigend object in het bericht moet worden opgenomen, dan kunnen meerdere `<update>`-elementen opgenomen. Het is natuurlijk ook mogelijk af te zien van het gebruik van het `<update>`-element en zelf de semantiek voor het doorgeven van de wijziging te definiëren.

Het element `<update>` heeft exact dezelfde structuur als de body van een kennisgeving. Het element `<update>` bevat dus naast het element `<parametersKennisgeving>` één (mutatiesoort 'T' of 'V') of twee (mutatiesoort 'W' of 'C') objecten. Zo'n object mag alleen elementen bevatten die zijn gedefinieerd voor een kennisgeving voor dat entiteittype. Daarnaast bevatten de elementen alle door StUF voor een kennisgeving voorgeschreven attributen. Eventuele extra parameters kunnen worden meegegeven door het element `<parameters>` als laatste element binnen `<update>` op te nemen.

Als de functionaliteit binnen het element `<update>` voldoende is, dan dient het in een vrij bericht gebruikt te worden en is het ongewenst dat de berichtontwerper de functionaliteit zelf specificeert.

7.2.4 Het opvragen/selecteren van gegevens

Vrije berichten zullen geregeld gedefinieerd worden voor het opvragen van gegevens, omdat de functionaliteit van vraag/antwoord of de structuur van het antwoordbericht niet voldoet. Om hergebruik van functionaliteit te faciliteren biedt de StUF-standaard hiervoor het gereserveerde element `<vraag>` met precies dezelfde structuur als de body van een vraagbericht. In een vrij bericht mag slechts één `<vraag>` element voorkomen. Eventuele extra parameters om de verwerking van de vraag te sturen kunnen worden opgenomen in het eerste element `<parameters>` in het vrije bericht. Het is natuurlijk ook mogelijk af te zien van het gebruik van het `<vraag>`-element en zelf de semantiek voor het stellen van een vraag te definiëren.

Als de functionaliteit binnen het element `<vraag>` voldoende is, dan dient het in een vrij bericht gebruikt te worden en is het ongewenst dat de berichtontwerper de functionaliteit zelf specificeert.

7.2.5 Zaakinformatie

StUF-berichten zullen regelmatig gebruikt worden voor het doorgeven van events of het doen van verzoeken in het kader van de afhandeling van een zaak. Om het werken met zaken te faciliteren kent de StUF-standaard het element `<zaakinfo>`, waarin gegevens over een zaak kunnen worden opgenomen. Als attribute binnen `<zaakinfo>` wordt `versieSectormodel` opgenomen met daarin de versie van het sectormodel waar binnen `<zaakinfo>` gebruik van wordt gemaakt. Binnen `<zaakinfo>` komt in elk geval het element `<zaak>` voor met de kerngegevens van de zaak, waarbinnen de event of het verzoek zich voordoet. Naast de kerngegevens mag het element `<zaak>` ook nog andere gegevens van de zaak bevatten.

Het lijkt verstandig meer gereserveerde elementen binnen `<zaakinfo>` te definiëren voor bijvoorbeeld de gewenste datum gereed, de beoogde uitvoerder en dergelijke. Deze elementen dienen gedefinieerd te worden binnen het sectormodel Zaken in de vorm van een complexType `<Zaakinfo>`.

Structuur vrij bericht

8. Protocolbindingen

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. StUF heeft als ambitie de inhoud ("payload") van berichten te standaardiseren en laat de definitie van het communicatieprotocol vrij. De hoofdstuk beschrijft de binding van StUF-berichtenverkeer aan twee protocollen:

1. bestand via een zelf te kiezen uitwisselingsmechanisme (ftp, draagbaar medium, e.d.)
2. WSDL 1.1 (Web Services Description Language) met SOAP1.1 en http als onderliggend transportmechanisme

De uitwisseling via bestand is nodig om grote hoeveelheden berichten batchmatig efficiënt te kunnen uitwisselen. Denk bijvoorbeeld aan het initieel vullen van een database met behulp van StUF-berichten. Bestandsuitwisseling zal gebruikt worden, als interactieve uitwisseling onaantrekkelijk of onmogelijk is, bijvoorbeeld omdat er geen datanetwerk voorhanden is of omdat de capaciteit en performance van het netwerk onvoldoende is of omdat het netwerk te duur is.

Voor het interactief uitwisselen is gekozen voor WSDL 1.1 vanwege haar laagdrempeligheid en haar brede toepassing in de praktijk. StUF-berichtenverkeer kan ook aan andere communicatieprotocollen gebonden worden zoals bijvoorbeeld ebMS (zie [EBMS]) uit de ebXML-familie. Voor de binding aan ebMS is een voorstel beschikbaar op het StUF-forum. Dit voorstel is nog niet in de standaard opgenomen, omdat er nog geen partijen zijn begonnen met de implementatie ervan.

8.1 Uitwisseling via een bestand

De via bestand over te dragen StUF -berichten worden in de gewenste verwerkingsvolgorde weggeschreven in een XML-bestand met als root-element `<StUF-berichtenSet>`). Binnen het `<StUF-berichtenSet>` element mogen alleen asynchrone berichten worden opgenomen. De ontvangst van berichten geleverd in een bestand hoeft niet bevestigd te worden. Omdat een sectormodel eigen namen voor berichten definieert, is het niet mogelijk om in het StUF-schema een nadere definitie van het element `<StUF-berichtenSet>` te geven.

In een bestand kunnen grote hoeveelheden StUF-berichten worden opgeslagen. StUF geeft geen voorschriften voor het splitsen van een groot bestand in deelbestanden. Technieken als TAR, ZIP en RAR bieden functionaliteit om een bestand te splitsen. De ontvanger zal de bestanden zonodig weer samenvoegen en aan de geadresseerde applicatie aanbieden. We zullen in dit hoofdstuk verder geen aandacht besteden aan deze vorm van uitwisseling.

8.2 Berichtuitwisseling op basis van WSDL, SOAP en http

In deze binding wordt de uitwisseling van StUF-berichten gedefinieerd met behulp van WSDL (versie 1.1, zie [WSDL]). De service- en de berichtdefinities dienen te voldoen aan het WS-I Basic Profile 1.1 [WSIBP]. Deze paragraaf gaat in op het gebruik van de WSDL-structuren `<message>`, `<portType>`, `<binding>` en `<service>`, nadat enkele uitgangspunten zijn besproken.

Een SOAP-envelope bevat nul of meer `<header>` elementen en precies één `<body>` element. Het `<header>` element is primair bedoeld voor de sturing van het transport en hoog niveau foutafhandeling. Het `<body>` element is bedoeld voor de payload. Het `<header>` element wordt niet gevuld, omdat de OSB Koppelvlak Standaard WUS [OSBWUS] voorschrijft dat dit slechts gebruikt mag worden voor in de WS-I standaard (respectievelijk de onderliggende OASIS/W3C standaards) gedefinieerde velden. Het complete StUF-bericht inclusief de berichtstuurgegevens wordt opgenomen in het `<body>` element.

Functioneel kent StUF de interactiepatronen notificatie (er wordt geen respons terugverwacht) en verzoek/respons in de smaken synchroon en asynchroon. De StUF-standaard schrijft in verband met Quality of Service voor dat er op een asynchrone notificatie en op een asynchroon verzoek of een asynchrone respons gereageerd wordt met een Bv03-bevestigingsbericht of een Fo03-foutbericht. Op het niveau van de servicedefinitie in het wsdl-bestand is er dus ook op asynchrone berichten een respons.

SOAP voorziet via het attribute `encodingStyle` in de mogelijkheid verschillende vormen van serialisatie te ondersteunen. StUF maakt gebruikt van de standaard serialisatie voor XML-berichten en verbiedt daarom in overeenstemming met WS-I BP 1.1 het gebruik van het attribute `encodingStyle` binnen een wsdl voor StUF-berichten.

Als transportprotocol wordt http gebruikt. Binnen http wordt op protocolniveau op elk verzoek gereageerd met een respons. Bij gebruik van http kan daarom voor wat betreft time outs op de connectie worden aangesloten op de connection time out foutafhandeling van http. Als op http niveau een connection time out fout wordt gegeven, dan dient de zender ervan uit te gaan, dat het verzonden bericht niet is aangekomen en het op een later tijdstip opnieuw te versturen.

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 (zie ook paragraaf 4.4).

8.2.1 Het gebruik van het <message> element binnen StUF

Elk StUF-bericht dient als `<part>` binnen een `<message>` element gedefinieerd te worden. De `name` voor het `<part>` is “body”. Er moet via het attribute `element` verwezen worden naar het element met het bericht in het sectormodel, omdat anders niet voldaan wordt aan het WS-I Basic Profile 1.0. Het attribute `name` binnen `<message>` heeft als waarde de naam van het element waarnaar binnen `<part>` verwezen wordt.

StUF schrijft voor dat er voor elke combinatie van `<berichtcode>` en `<entiteittype>` uit de berichtstuurgegevens een apart berichtelement wordt gedefinieerd. Als het `<entiteittype>` leeg is, dan dient voor elke combinatie van `<berichtcode>` en `<functie>` een apart berichtelement te worden gedefinieerd. Deze keuze is gemaakt om drie redenen:

1. Services dienen zo scherp mogelijk te worden gedefinieerd
2. Geautomatiseerde codegeneratie op basis van de wsdl wordt hiermee eenvoudiger
3. Het is mogelijk om in de wsdl accuraat aan te geven welke services exact ondersteund worden.

De definitie van de `<message>` elementen voor de in StUF-standaard gedefinieerde bevestigingsberichten en foutberichten zijn opgenomen in het bestand StUF0300xx.types.wsdl met als namespace “http://www.egem.nl/StUF/wsdl/stuf0300”. xx staat voor het interne versienummer van de bestanden voor deze namespace. Dit bestand kan worden geïmporteerd in de wsdl-bestanden behorend bij een sectormodel.

8.2.2 Het gebruik van het <portType> element binnen StUF

Binnen het `<portType>` element worden in de vorm van `<operation>` elementen de ondersteunde interactiepatronen gedefinieerd. Een `<operation>` voor een StUF-bericht bevat als elementen `<input>`, `<output>` en nul of één keer `<fault>`. Het attribute `name` van `<operation>` en het attribute `message` van `<input>` worden gevuld met het attribute `name` van de corresponderende `<message>`, binnen `<input>` uiteraard gekwalificeerd met de target namespace. Voor de verschillende onderkende portTypes wordt hieronder beschreven hoe het attribute `name` wordt gevuld op de elementen `<output>` en `<fault>`.

De StUF-standaard schrijft voor een StUF end node in ieder geval het gebruik van de volgende portTypes voor:

- OntvangAsynchroon
  Het element `<portType>` heeft als waarde voor het `name` attribute 'OntvangAsynchroon'. Binnen dit `<portType>` worden `<operation>` elementen gedefinieerd voor alle asynchrone berichten, die het systeem kan ontvangen. Het element `<output>` heeft als inhoud: `<output message=”StUFwsdl:Bv03”/>` en het element `<fault>` heeft als inhoud `<fault name=”fout” message=”StUFwsdl:Fo03”/>` met StUFwsdl de namespace qualifier voor de namespace “http://www.egem.nl/StUF/wsdl/stuf0300”. Als het systeem ook asynchrone verzoek/respons berichten ondersteunt, mag niet vergeten worden `<operation>` elementen te definiëren voor de Bv01- en Fo01-berichten.
- VerwerkSynchroneKennisgeving
  Het element `<portType>` heeft als waarde voor het `name` attribute 'VerwerkSynchroneKennisgeving'. Binnen dit `<portType>` worden `<operation>` elementen gedefinieerd voor alle synchrone kennisgeving- en synchronisatieberichten die het systeem ondersteund. Het element `<output>` heeft als inhoud: `<output message=”StUFwsdl:Bv02”/>` en het element `<fault>` heeft als inhoud `<fault name=”fout” message=”StUFwsdl:Fo02”/>` met StUFwsdl de namespace qualifier voor de namespace “http://www.egem.nl/StUF/wsdl/stuf0300”.
- BeantwoordVraag
  Het element `<portType>` heeft als waarde voor het `name` attribute 'BeantwoordVraag'. Binnen dit `<portType>` worden `<operation>` elementen gedefinieerd voor alle synchrone vraagberichten die het

systeem kan beantwoorden. In het element <output> wordt het attribute message gevuld met de elementnaam van het antwoordbericht gekwalificeerd met de targetnamespace corresponderend met het vraagbericht. Het element <fault> heeft als inhoud <fault name="fout" message="StUFwsdl:Fo01"/> met StUFwsdl de namespace qualifier voor de namespace "http://www.egem.nl/StUF/wsdl/stuf0300".

- VerwerkTriggerbericht
  Het portType <portType name="VerwerkTriggerbericht"> voor het verwerken van het triggerbericht is gedefinieerd in de namespace "http://www.egem.nl/StUF/wsdl/stuf0300", zie aldaar.
- VerwerkSynchroonVrijBericht
  Het element <portType> heeft als waarde voor het name attribute 'VerwerkSynchroonVrijBericht'. Binnen dit <portType> worden <operation> elementen gedefinieerd voor alle synchrone vrije verzoekberichten die het systeem ondersteunt. In het element <output> wordt het attribute message gevuld met de elementnaam van het vrije responsbericht gekwalificeerd met de targetnamespace corresponderend met het vrije verzoekbericht. Het element <fault> heeft als inhoud <fault name="fout" message="StUFwsdl:Fo01"/> met StUFwsdl de namespace qualifier voor de namespace "http://www.egem.nl/StUF/wsdl/stuf0300".

Een intermediaire node is vrij in het definiëren van de portTypes voor het ontvangen van asynchrone StUF-berichten. Binnen een portType worden <operation> elementen gedefinieerd voor de asynchrone berichten, die de intermediaire node kan ontvangen. Het element <output> heeft als inhoud: <output message="StUFwsdl:Bv04"/>. Het element <fault> hoeft niet voor te komen. Als het voorkomt, dan dienen er afspraken gemaakt te worden tussen de intermediaire node en de aanbieder van StUF-berichten aan de intermediaire node.

De OSB Koppelvlak Standaard WUS [OSBWUS] schrijft voor dat er per <portType> een wsdl-file wordt gemaakt. In deze wsdl-file kunnen voor dat <portType>, dan ook de <binding> en de <service> beschreven worden. [OSBWUS] verbiedt dus het groeperen van <operation> elementen uit verschillende <portType> elementen in één <service>.

StUF kiest op grond van de [OSBWUS] voorschriften voor de volgende werkwijze bij het definiëren van wsdl's. In een wsdl met als naam de code voor het sectormodel inclusief zescijferig versienummer (vier voor de versie van het sectormodel plus twee voor verschillende versies met verbeteringen) gevolgd door '.types.wsdl' worden alle <message> elementen voor een sectormodel opgenomen, bijvoorbeeld 'bg020501.types.wsdl'. In bestanden met de naam van het sectormodel inclusief zescijferig versienummer gevolgd door '.xxx.wsdl' met xxx de de waarde van het name attribute worden het <portType>, <binding> en <service> element voor een <portType> gespecificeerd.

Een systeem is verplicht een hierboven gedefinieerd <portType> element te ondersteunen, als het minstens één van de erbinnen te definiëren interactiepatronen ondersteunt.

Indien een implementerend systeem andere <portType> elementen wil definiëren, dan mag dit naast de door StUF voorgeschreven <portType> elementen, die in elk geval ondersteund moeten worden. De reden hiervoor is dat StUF wil zorgen voor interoperabiliteit tussen StUF-compliant systemen zonder dat er per StUF-compliant systeem nog van alles en nog wat geconfigureerd moet worden. Door een implementerend systeem zelf gedefinieerde <portType> elementen mogen <operation> elementen bevatten voor interacties die reeds voorkomen in de door StUF voorgeschreven <portType> elementen.

8.2.3 Het gebruik van het <binding> element binnen StUF

Het <binding> element specificeert met welk protocol en hoe exact binnen dat protocol een interactiepatroon is geïmplementeerd, dat binnen een <portType> in een <operation> is gedefinieerd. Een <binding> wordt altijd gemaakt voor één <portType>. Per <portType> zijn meerdere <binding> elementen toegestaan. StUF schrijft voor dat voor de StUF gedefinieerde <portType> element precies één <binding> element wordt gebruikt met als naam SOAP samengevoegd met de naam voor het <portType>. Voor het <portType> met als waarde voor het name attribute OntvangAsynchroon wordt het <binding> element dan

```
<binding name="SOAPOntvangAsynchroon" type="tns:OntvangAsynchroon">
```

met tns de namespace qualifier van de targetnamespace. Voor zelf gedefinieerde <portType> elementen mogen meerdere <binding> elementen worden gebruikt.

StUF heeft gekozen voor een binding aan SOAP en http. Deze binding is gedefinieerd in de namespace "http://schemas.xmlsoap.org/wsdl/soap". Voor deze namespace wordt in het vervolg 'soap' als namespace qualifier gebruikt. StUF kiest voor document style berichten en niet voor rpc style. Dit wordt gespecificeerd door als kind in het <binding> element op te nemen

```
<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>.
```

Daarnaast wordt voor de te binden <operation> elementen uit het <portType> een <operation name="nmtoken"> element binnen het <binding> element opgenomen met nmtoken de naam van het <operation> element binnen het <portType>. Binnen dit <operation> element wordt gespecificeerd hoe de berichten verzonden worden. Het eerste element binnen een <operation> is

```
<soap:operation soapAction=""/>
```

Het attribute soapAction wordt gevuld met "http://www.egem.nl/StUF" in plaats van met een lege string, zoals de OSB Koppelvlak Standaard WUS dit voorschrijft [OSBWUS], omdat XMLSpy een lege string niet accepteert.

Vervolgens wordt in de elementen <input>, <output> en <fault> aangegeven wat wordt opgenomen in het <header> en <body> element van de SOAP-envelope. Het <output> element bevat de normale respons op het inkomend bericht in <input> en <fault> de respons in geval van een foutsituatie. De inhoud van de SOAP-body voor het <input> en <output> element wordt gespecificeerd in het element

```
<soap:body use="literal"/>
```

Het parts attribute binnen <soap:body> is niet nodig, omdat de <messsage> behorend bij de <operation> slechts één <part> bevat dat als kind element in de SOAP-body wordt opgenomen. Het verplichte use attribute in <soap:body> geeft met de waarde literal samen met het ontbreken van het encodingStyle attribute aan dat er geen additionele encoding-regels zijn.

De inhoud van <fault> elementen binnen een <operation> beschrijft hoe het SOAP-fault <detail> element moet worden gevuld in geval van een fout. Een <operation> element in een <binding> bevat altijd precies één <soap:fault> element met als inhoud

```
<fault name="fout">
     <soap:fault name="fout" use="literal"/>
</fault>
```

omdat binnen <operation> elementen in een <portType> voor StUF altijd slechts één <fault> element voorkomt met name="fout". De waarde van het name attribute binnen <fault> is vrij te kiezen en de waarde van het name attribute binnen het <soap:fault> element dient te verwijzen naar het name atttribute van een <fault> element binnen de <operation> in het <portType>. Voor het gemak zijn beide gelijk gekozen. Het element <faultcode> van de SOAP-fault wordt gevuld met de namespace qualifier voor de namespace "http://schemas.xmlsoap.org/soap/envelope/ " gevolgd door de waarde van <plek> binnen het foutbericht, bijvoorbeeld "SOAP-ENV:Client". Het element <faultstring> van de SOAP-fault wordt gevuld met de waarde van <omschrijving> binnen het foutbericht. Het element <faultactor> wordt niet opgenomen in de SOAP-fault.

Een voorbeeld van een <operation> element voor de ontvangst van asynchrone berichten staat hieronder:

```
<operation name="aangifteBinnenlandseVerhuizing">
     <soap:operation soapAction=""/>
     <input>
          <soap:body use="literal"/>
     </input>
```

```
     <output>
          <soap:body use="literal"/>
     </output>
     <fault name="fout">
          <soap:fault name="fout" use="literal"/>
     </fault>
</operation>
```

Het enige dat hierin varieert is de waarde van het `name` attribute voor de `<operation>`.

8.2.4 Het gebruik van het <service> element binnen StUF

Binnen het `<service>` element kunnen in de vorm van `<port>` elementen één of meer gerelateerde 'communication endpoints' voor een `<binding>` worden gedefinieerd. Gegeven de beperking in [OSBWUS] dat een wsdl slechts één `<portType>` element mag bevatten, zal een `<service>` ook altijd slechts één `<port>` element bevatten.

Het attribute `name` op het `<service>` element en het attribute name op het `<port>` element krijgen dezelfde waarde als het `name` attribute van het `<portType>`. Het atttibute `binding` op het `<port>` element krijgt als waarde de namespace qualifier voor de targetnamespace gevolgd door de waarde van het `name` attribute van het `<binding>` element. Op het element `<soap:address>` specificeert het attribute `location` waar de webservice te vinden is. De waarde van het location attribute dient te eindigen met een '/' gevolgd door de waarde van het `name` attribute van het `<portType>`, `<service>` en `<port>` element. Hiervoor dient een URI te staan die in alle door StUF voorgeschreven portType-wsdl's voor één sectormodel identiek is. Verschillende sectormodellen mogen deze URI delen.

Een voorbeeld van het <service> element voor het portType OntvangAsynchroon staat hieronder.

```
<service name="OntvangAsynchroon">
     <port name="OntvangAsynchroon" binding="tns:SOAPOntvangAsynchroon">
          <soap:address location="http://example.com/OntvangAsynchroon"/>
     </port>
</service>
```

Tabel met mogelijke foutberichten

Foutsituatie (<omschrijving>)	**Soort fout**	**<code>**	**<plek>**	**<details>**
Versie StUF niet ondersteund	2,3	StUF001	Server	De dichtstbijzijnde[12] wel ondersteunde versie StUF.
Sectormodel niet ondersteund	2,3	StUF004	Server	-
Versie sectormodel niet ondersteund	2,3	StUF007	Server	De dichtstbijzijnde wel ondersteunde versie sectormodel
Combinatie van ontvangende organisatie, applicatie en administratie onbekend	2,3	StUF010	Client	
Combinatie van zendende organisatie, applicatie en administratie onbekend	2,3	StUF013	Client	
Combinatie zender en referentienummer niet uniek	2,3	StUF016	Client	
TijdstipBericht niet groter dan voorgaand TijdstipBericht van zender	2,3	StUF019	Client	
Berichtcode onbekend	2,3	StUF022	Client	
Berichtcode niet ondersteund	2,3	StUF025	Server	
Entiteittype onbekend binnen sectormodel	2,3	StUF028	Client	
Entiteittype niet ondersteund	2,3	StUF031	Server	
Functie onbekend binnen sectomodel	2,3	StUF034	Client	
Functie niet ondersteund	2,3	StUF037	Server	
Combinatie van berichtcode, entiteittype en functie niet ondersteund	2,3	StUF040	Server	
Crossreferentienummer niet bekend	2,3	StUF043	Client	
Opslaan bericht niet mogelijk	3	StUF046	Server	
Proces voor afhandelen synchroon bericht niet beschikbaar	2	StUF049	Server	
Het zendende systeem is niet geautoriseerd voor de gevraagde combinatie van berichtcode, entiteittype en functie	1,2	StUF052	Client	
Berichtbody is niet conform schema in sectormodel	1,2	StUF055	Client	
Proces voor afhandelen bericht geeft fout	1,2	StUF058	Server	Desgewenst een omschrijving van de fout in het afhandelende proces
Starten berichtverzending niet mogelijk binnen 5 minuten	2	StUF061	Server	
Object niet gevonden	2	StUF064	Server	
Dubbelen voor object gevonden	2	StUF067	Server	
Synchronisatiebericht historisch niet	2	StUF070	Client	

[12]Onder dichtstbijzijnde wordt hier verstaan de laagst ondersteunde versie, als de versie in het bericht lager is dan de ondersteunde versie en de hoogst ondersteunde versie, als de versie in het bericht hoger is dan de ondersteunde versie.

Foutsituatie (<omschrijving>)	**Soort fout**	**<code>**	**<plek>**	**<details>**
consistent				
<vanaf> en <totEnMet> bevatten niet dezelfde elementen	1, 2	StUF076	Client	De naam van het verschillend voorkomende element
Een element komt voor in zowel <gelijk> als <vanaf> en <totEnMet>	1, 2	StUF079	Client	De naam van het element
Het attribute StUF:exact is gebruikt op een element binnen <vanaf> of <totEnMet>	1, 2	StUF082	Client	De naam van het element
Onvolledige datum binnen <vanaf> of <totEnMet>	1, 2	StUF085	Client	De naam van het element met de onvolledige datum.
Meer dan één sleutel gespecificeerd als selectiecriterium	1, 2	StUF088	Client	
Een sleutel en andere elementen gespecificeerd als selectiecriterium	1, 2	StUF091	Client	
Ontvangend systeem registreert sleutel in het verzendende systeem niet	1, 2	StUF094	Server	
Zowel het attribute scope als een inhoud gespecificeerd voor het element <scope>	1, 2	StUF097	Client	
Een gerelateerde uit een choice komt dubbel voor binnen het element <scope>	1, 2	StUF100	Client	De elementnaam van de relatie gevolgd door een spatie en de elementnaam van de gerelateerde
IndicatorVervolgvraag is true, maar het element <start> ontbreekt	1, 2	StUF103	Client	
IndicatorHistorisch 'E' wordt niet ondersteund	1, 2	StUF106	Server	
IndicatorHistorisch 'G' wordt niet ondersteund	1, 2	StUF109	Server	
Antwoordend systeem ondersteunt afhandelen asynchrone vragen niet	1	StUF112	Server	
Gebruiker binnen zendend systeem onbekend in antwoordend systeem en nodig voor autorisatie	1, 2	StUF115	Server	
Antwoordend systeem ondersteunt gevraagde sortering niet	1, 2	StUF118	Server	
Beantwoording asynchrone vraag vergt teveel systeemresources	1	StUF121	Server	
Het antwoordende systeem is niet in staat het verzoek af te handelen binnen de connection time out	2	StUF960	Server	

Tabel 1: Samenvatting onderkende StUF-foutberichten

OPEN ISSUES:

Issues voor nieuwe versies van de StUF standaard:

- Wensen voor extra functionaliteit:
  - XQuery-syntax en -semantiek voor vraagberichten

- Aandachtspunten voor bestaande functionaliteit:
  - Het tevens opnemen van de toekomstige situatie(s) in de “historische” gegevens. Vervangen van het begrip “historisch” voor het begrip “periodegebonden”.

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

[StUFXSD] StUF Community / StUF Standaard / StUF 03.00 / stuf030001.xsd.
http://www.egem.nl/mijnegem

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

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

WSIBP] Versie 1.1
www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html

[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

[OSBWUS] Versie 0.9
OSB Koppelvlak Standaard WUS Versie 0.9, documentversie 0.55

Begrippenlijst

Antwoordbericht	Bericht dat als respons wordt verstuurd op een inkomend vraagbericht
Attribute	Een kenmerk van een XML-element dat een waarde kan hebben.
Attribuut	(Essentieel) kenmerk van een object
Bericht	Een string waarin in de vorm van XML informatie is gecodeerd, die tussen zender en ontvanger kan worden uitgewisseld
Bevestigingsbericht	Een bericht dat gebruikt wordt om de ontvangst van een asynchroon bericht te bevestigen of om de min of meer geslaagde verwerking van een verzoek te bevestigen.
Element	XML-element te gebruiken om een informatie-element mee te onderscheiden in een XML-document.
Entiteit	Representatie van een een in de werkelijkheid bestaand ding of eenheid binnen een informatiesysteem of bericht.
Fundamentele entiteit	Een entiteit die een autonoom in de werkelijkheid bestaand object met veranderende eigenschappen representeert
Relatie-entiteit	Een entiteit die een relatie tussen twee autonoom in de werkelijkheid bestaande objecten representeert
Tabelentiteit	Een entiteit die een niet in de werkelijkheid bestaand autonoom object representeert of een object met onveranderlijke eigenschappen
Entiteittype	Een typering van een groep objecten naar een of meer gemeenschappelijke kenmerken
Foutbericht	Bericht als reactie op een notificatie of een verzoek met de reden waarom het ontvangen bericht niet verwerkt kan worden.
Gemeentelijke Functioneel Ontwerp (GFO)	Een beschrijving van de relevante objecten en hun relaties voor een bepaald gemeentelijk beleidsdomein.
http	HyperText Transfer Protocol)is een standaard protocol voor verzending van webpagina's op het Internet
Interactiepatroon	Een patroon voor de interactie tussen een zender en ontvanger van berichten. StUF ondersteunt alleen de patronen notificatie en verzoek/respons
Kennisgeving(bericht)	Een notificatiebericht voor het informeren van de ontvanger over een gebeurtenis of voor het doorvoeren van een transactie in een database.
Notificatie(bericht)	Een bericht waarop de zender van de ontvanger geen antwoord verwacht. (In wsdl-termen een one-way bericht)
Object	Een voorwerp (fysiek of imaginair) in de werkelijke wereld
Ontvanger	Systeem dat een door een zender verzonden bericht ontvangt.
Respons(bericht)	Bericht waarmee gereageerd wordt op een inkomend verzoekbericht
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)
Sleutel	(Attribute sleutel): unieke identificerende waarde voor een entiteit
SOAP	Simple Object Access Protocol; SOAP definieert een standaard structuur voor het verpakken van tussen een zender en ontvanger uit te wisselen informatie
Template	Formaat (sjabloon)
Triggerbericht	Bericht waarmee de zender het ontvangende systeem vraagt binnen 5 minuten de voor de zender klaarstaande berichten te verzenden.
Verzoek(bericht)	Een bericht waarop de zender van de ontvanger een respons verwacht in de vorm van een bericht.
Voorkomen	Term om een concreet item in een verzameling aan te duiden. Dhr Janssen is bijvoorbeeld een voorkomen van het entiteittype Persoon.
Vraagbericht	Bericht dat om het in de vorm van een antwoordbericht teruggeven van de gegevens van één of meer objecten vraagt.

Vrij bericht	Bericht waarvan de semantiek niet vastligt in de StUF-standaard, maar dat wel voldoet aan een aantal door de StUF-standaard voorgeschreven kenmerken.
WSDL	Web Services Description Language. Een op XML gebaseerde specificatietaal om web-diensten 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
Zender	Systeem dat een bericht verstuurt naar een ontvangend systeem.