Ministerie van Binnenlandse Zaken en Koninkrijksrelaties

Migratievoorziening Technisch ontwerp Conversie

1.3

Datum 11-1-2017
Status Definitief

Inhoudsopgave

Versiehistorie

Datum	Versie	Omschrijving	Auteur
12-05-2014	0.1	Initiële versie	operatie BRP
14-05-2014	0.5	Review commentaar verwerkt	operatie BRP
17-06-2014	0.6	Module namen aangepast n.a.v. NFR	operatie BRP
24-06-2014	0.7	Uitbreiding conversie autorisatie	operatie BRP
14-7-2014	0.8	Foutcodes conversie autorisaties en afnemersindicaties toegevoegd.	operatie BRP
17-9-2014	0.9	Definitief gemaakt t.b.v. release 2.2	operatie BRP
13-07-2015	0.10	Foutcodes aangepast/uitgebreid t.b.v. release 3.1	operatie BRP
28-09-2015	1.0	Referenties bijgewerkt	operatie BRP
26-01-2015	1.1	Versie en referenties bijgewerkt	operatie BRP
03-08-2016	1.2	Wijzigingen n.a.v. refactor Java-model en verplaatsing entiteiten naar Algemeen	operatie BRP
11-01-2017	1.3	Referentie Lo3 aangepast	operatie BRP

Reviewhistorie

Datum	Versie	Omschrijving	Reviewers
13-05-2014	0.1	Inhoudelijke review	operatie BRP

1 Inleiding

1.1 Beknopte omschrijving

Dit technisch ontwerp beschrijft de Conversie software bibliotheek, die zorgt voor het converteren van gegevens tussen de gegevensmodellen van GBA en BRP.

1.2 Referenties

#	Document	Organisatie	Versie	Datum
[LO-GBA]	Logisch Ontwerp GBA	BPR	3.10	08-10-2016
[LO-BRP]	Logisch Ontwerp BRP	OperatieBRP		
[SAD]	Software Architectuur Document Migratievoorziening BRP	OperatieBRP		
[CONV]	Documentatie Bidirectionele Conversie	OperatieBRP		
[UC102]	MV UC102 Vul autorisatietabelregels in BRP initieel	OperatieBRP		
[UC103]	MV UC103 Vul afnemersindicaties in BRP initieel	OperatieBRP		

1.3 Leeswijzer

Dit document is bedoeld voor ontwikkelaars of software architecten die kennis nodig hebben over de interne structuur van de conversie software bibliotheek.
Benodigde voorkennis is een goede bekendheid met de gegevensmodellen van zowel GBA als BRP, zoals beschreven in het logisch ontwerp GBA en BRP [LO-GBA, LO-BRP]. Tevens is goede kennis van de specificatie van de conversie software nodig, zoals beschreven in Documentatie Bidirectionele Conversie [CONV], Use Case 102 [UC102] en Use Case 103 [UC103]. Ten slotte is kennis van de architectuur van de migratievoorziening [SAD] wenselijk om de conversie in de juiste context te plaatsen.

1.4 Indeling

Na de inleidende hoofdstukken begint dit document met uitleg over de data structuren die gebruikt worden in het hoofdstuk ‘Conversie Model’ gevolgd door uitleg over de opbouw van de conversie functionaliteit in ‘Conversie Regels’.
Hierop volgt een aantal hoofdstukken over de services die de conversie software bibliotheek aanbiedt. Hierin worden de verwachtingen beschreven waaraan deze services moeten voldoen.
Dit document sluit af met een overzicht van de ‘Ontwerpbeslissingen’ en een aantal opmerkingen over de ‘Test Hulpmiddelen’ die wordt gebruikt.

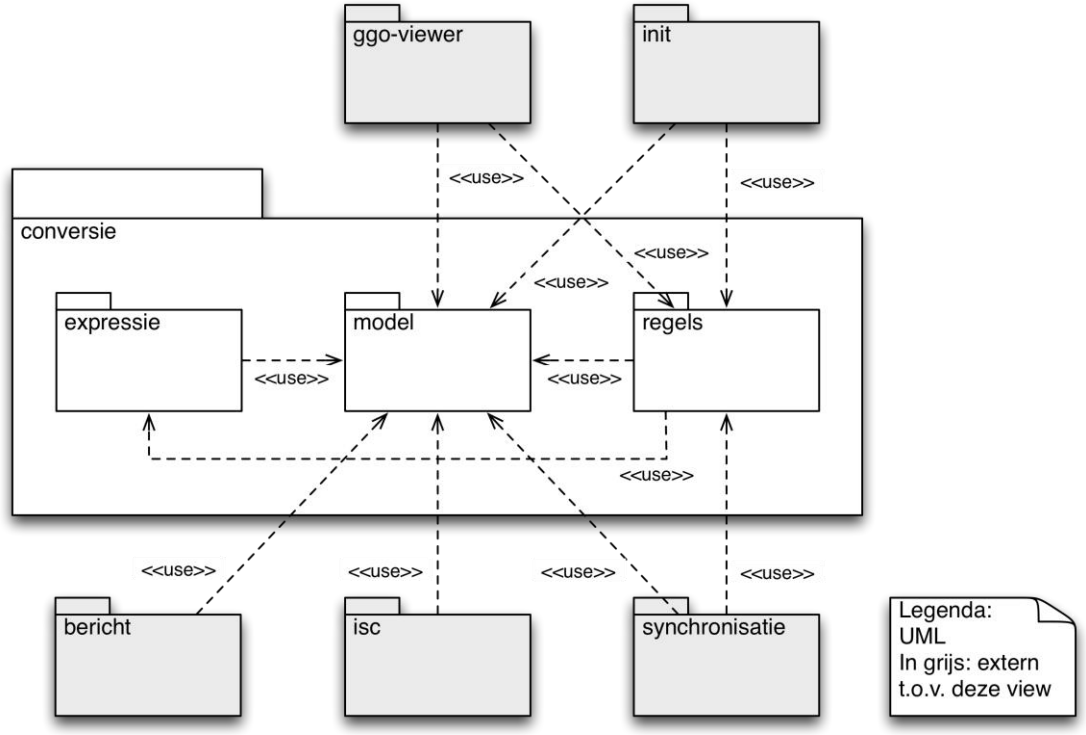
1.5 Terminologie

Binnen dit document worden de termen GBA en LO3 beiden gebruikt. GBA is de naam van het oude system en LO3 verwijst naar de logisch gegevensmodel binnen het GBA. Binnen de technische beschrijvingen wordt primair de term LO3 gebruikt omdat dit overeenkomt met de naamgeving in de broncode.

2 Context

In de technische architectuur is ervoor gekozen om alle kennis omtrent conversie tussen de LO3 en BRP gegevensmodellen te concentreren in één module. Deze module wordt vervolgens als software bibliotheek gebruikt binnen alle andere componenten die gegevens moeten converteren tussen de twee gegevensmodellen. In het software architectuur document [SAD] is aangegeven welke componenten gebruik maken van de Conversie software bibliotheek.

In de onderstaande afbeelding zijn ter illustratie de afhankelijkheden tussen software bibliotheken weergegeven, in de paragraaf 'View-deel 3: Conversie' van de sectie 'Logical View' van het [SAD] kan meer detailinformatie hieromtrent worden gevonden.



Afbeelding 1 Conversie in context

Dit document richt zich alleen op de zaken binnen de Conversie software bibliotheek.

2.1 Maven Structuur

De conversie software bibliotheek is opgedeeld in 2 maven projecten.

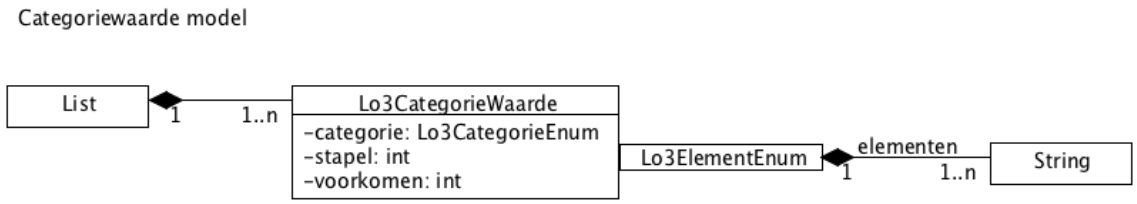
Project	Subproject	Beschrijving
migr-conversie	migr-conversie-model	Bevat de data-modellen voor de conversie.
migr-conversie	migr-conversie-regels	Bevat de conversie logica die gegevens tussen de GBA en BRP modellen heen en weer converteert. Bevat tevens de extern toegankelijke services van Conversie.
migr-conversie	migr-conversie-expressie	Bevat de conversie logica die de GBA autorisatie rubrieken en voorwaarderegels converteert naar de BRP expressietaal.

3 Conversie Model

Dit hoofdstuk beschrijft de verschillende modellen die worden gebruikt bij de conversie.

3.1 Persoonslijst modellen

3.1.1 LO3 categoriewaarde model



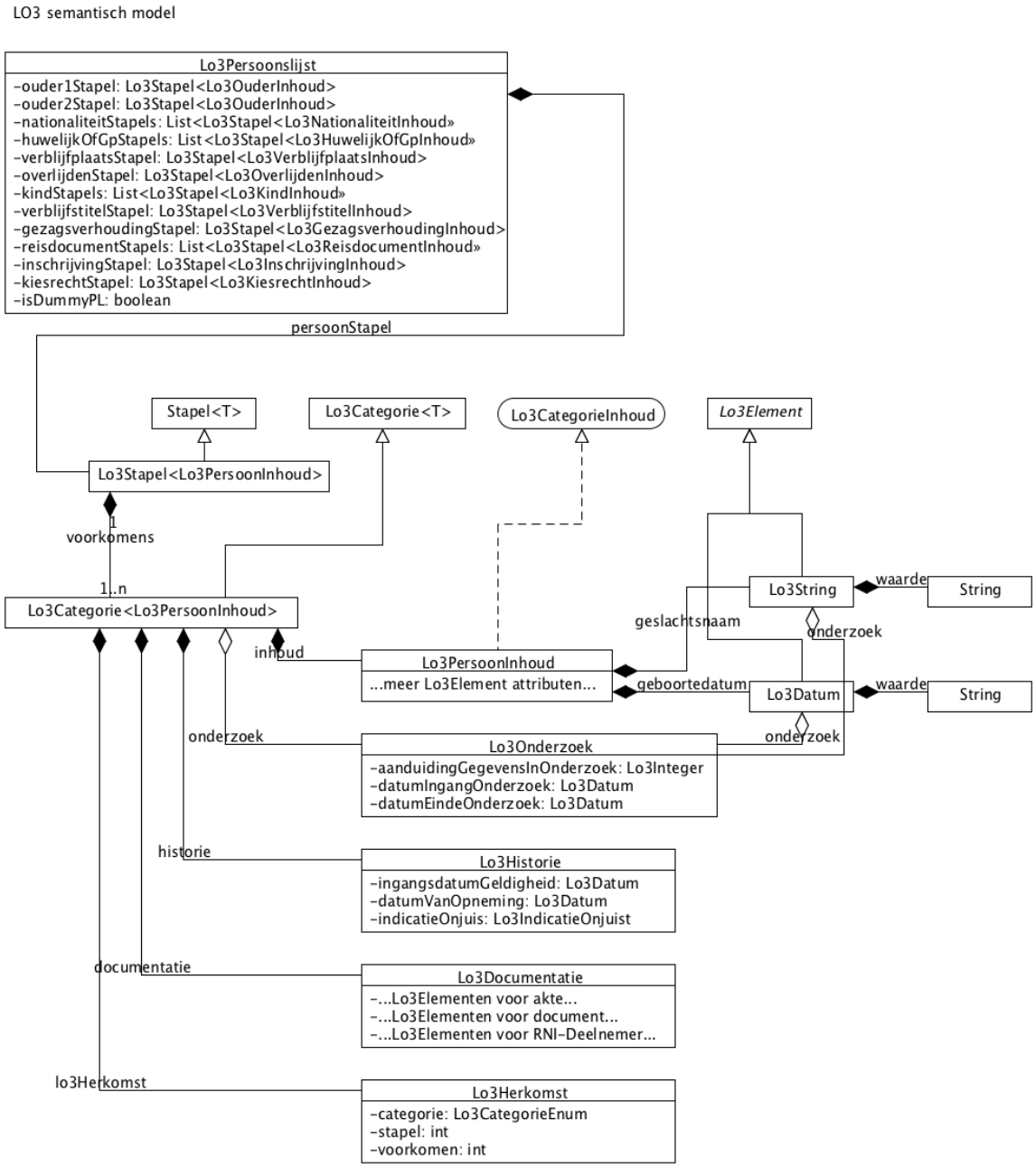
Afbeelding 2 LO3 categoriewaarde model

Het categoriewaarde model is een zeer eenvoudig model om de gegevens van een LO3 persoonslijst in op te slaan.
Het model bestaat eenvoudigweg uit een lijst met Lo3CategorieWaarde objecten.
Elk Lo3CategorieWaarde object bevat herkomstgegevens (categorie, stapelnummer, en voorkomennummer. Zie voor details hierover de beschrijving van het LO3 semantisch model) en een Map van Lo3ElementEnum naar String. Deze map bevat alle inhoudelijke gegevens uit het categorie voorkomen.

Dit model is een tussenstap tussen een Lg01 berichtformaat persoonslijst en het volledige semantische LO3 model dat in de volgende sectie wordt beschreven. De eenvoud van het model laat toe dat het een persoonslijst kan representeren die syntactische fouten bevat, zoals een niet-numerieke waarde in een numeriek veld.

Dit model is veranderlijk. Dat wil zeggen dat er geen ingebouwde bescherming is tegen tussentijdse wijzigingen in de objecten van het model. Dit in tegenstelling tot de overige modellen. Dit is gedaan omdat dit een zeer eenvoudig en tijdelijk model is waar geen complexe handelingen mee worden verricht.

3.1.2 LO3 semantisch model



Afbeelding 3 LO3 semantisch model

Dit is het belangrijkste LO3 model. Het is opgezet als een onveranderlijk. Zie hiervoor de ontwerpbeslissing 'Ontwerpbeslissingen'.

3.1.2.1 Lo3Persoonslijst

Het LO3 semantisch model wordt gerepresenteerd als een object van het type Lo3Persoonslijst. Dit model is gebaseerd op het logisch model van de LO3 persoonslijst, zoals dat ook wordt gebruikt in het GBA en is gedocumenteerd in [LO-GBA].
Gegevens waarvan in het [LO-GBA] is aangegeven dat er maximaal één actuele categorie op de persoonslijst kan voorkomen worden binnen het Lo3Persoonslijst object als een enkelvoudige Lo3Stapel opgenomen.

Gegevens waarvan in het [LO-GBA] is aangegeven dat er meer dan één actuele categorie op de persoonslijst kan voorkomen worden binnen het Lo3Persoonslijst object als een lijst van Lo3Stapels opgenomen.
In een object van het type Lo3Persoonslijst worden alleen gegevens van de categorienummers 01 t/m 13 en de bijbehorende historische categorienummers opgenomen. Specifiek worden categorienummers 14/64 en 15 niet opgenomen omdat deze niet als persoonsgegevens worden geconverteerd naar BRP.

3.1.2.2 Stapels en Voorkomens

Er is één uitbreiding in het conversie model ten opzichte van het model in [LO-GBA] en dat is de toevoeging van het concept Stapel.
Een Stapel is gedefinieerd als zijnde één actuele categorie met alle daarbij behorende historische categorieën uit de persoonslijst.
Binnen elk categorienummer krijgt elke stapel een stapelnummer dat de herkomst van de gegevens in het oorspronkelijke bericht aangeeft. Binnen elk categorienummer krijgt de eerste stapel het stapelnummer 0, en wordt vervolgens met één opgehoogd voor elke nieuwe stapel (en dus elk nieuw actueel voorkomen) die voor dat categorienummer wordt aangemaakt.
Een enkele actuele of historische categorie binnen een Stapel wordt aangeduid als een Voorkomen. Elk voorkomen heeft ook een voorkomennummer om de herkomst te identificeren. Het actuele voorkomen (de actuele categorie) krijgt het voorkomennummer 0 en wordt voor elk volgende voorkomen met 1 opgehoogd.
Omdat in LO3 een voorkomen alleen een datum aanvang geldigheid heeft, en geen datum einde geldigheid, wordt het beëindigen van een gegeven aangegeven door het opnemen van een leeg voorkomen. Hierdoor zal er altijd een actueel voorkomen in een stapel zijn opgenomen.

3.1.2.3 Categorie

De gegevens uit één categorie (voorkomen) worden opgeslagen in een Lo3Categorie object. Binnen dit object worden 5 gegevens opgeslagen:

- Lo3Historie (LO3 groepen 84 ‘onjuist’, 85 ‘geldigheid’ en 86 ‘opneming’)
- Lo3Documentatie (LO3 groepen 81 ‘akte’, 82 ‘document’ en 88 ‘RNI’)
- Lo3Onderzoek (LO3 groep 83 ‘procedure’)
- Lo3Herkomst (afgeleide gegevens om de brongegevens te kunnen identificeren)
- Inhoud, een subtype van Lo3CategorieInhoud (omvat alle overige gegevens uit de categorie)

3.1.2.4 Herkomst

Middels een combinatie van categorienummer, stapelnummer en voorkomennummer kan van elk voorkomen de exacte herkomst worden vastgesteld. Dit drietal gegevens wordt samengevat in een Lo3Herkomst object.

3.1.2.5 Inhoud

De inhoud van een categorie/voorkomen wordt vastgelegd in een subtype van Lo3CategorieInhoud. Elk actueel categorienummer heeft een eigen subtype dat precies de toegestane elementen uit die categorie bevat, uitgezonderd elementen uit groepen 81 t/m 86 die hierboven al zijn beschreven. Elk element heeft een specifiek type dat de betekenis van het element vastlegt, zoals Lo3Huisnummer of Lo3Datum. Alle element typen zijn een subtype van Lo3Element.

3.1.2.6 Onderzoek

Als er een onderzoek (groep 83 ‘procedure’) aanwezig is op een categorie, dan wordt dat niet alleen in een attribuut van het Lo3Categorie object aangegeven. Tevens wordt dat Lo3Onderzoek object direct aan elk individueel element dat bij het onderzoek betrokken is, gekoppeld. Dit kan ook in geval van een leeg element (onderzoek naar ontbrekende gegevens). Voor een leeg element wordt een instantie van het juiste elementtype aangemaakt dat inhoudelijk leeg is, maar wel aan het Lo3Onderzoek is gekoppeld.

Als zowel de element waarde leeg is, en er is geen onderzoek op het element, dan wordt het attribuut in de inhoud zelf met **null** gevuld. Er wordt geen element aangemaakt met een lege inhoud en geen onderzoek.

3.1.3 Tussenmodel

Het tussenmodel wordt maar op één moment gebruikt als tussenstap in de conversie van LO3 naar BRP, namelijk tussen de conversie van de inhoud naar BRP en de conversie van de historie naar BRP.
Het tussenmodel gebruikt de BRP groepen structuur, maar dan met een LO3 historie.
Zie voor meer details de beschrijving van het LO3 semantisch model en het BRP model.

3.1.3.1 TussenPersoonslijst

Het hoofdobject is een TussenPersoonslijst, die bevat TussenStapel objecten en lijsten van TussenStapel objecten. In plaats van een attribuut per LO3 categorie is er een attribuut per BRP groep. Zie [LO-BRP] voor informatie over BRP groepen.
De structuur van de gegevens volgt het BRP model. Gegevens over relaties worden volgens de BRP structuur opgenomen, en de IST objecten zijn ook opgenomen. Zie [CONV] en [LO-BRP] voor meer informatie over relaties en IST.

3.1.3.2 TussenStapel

De TussenStapel is equivalent aan de Lo3Stapel.

3.1.3.3 TussenGroep

De TussenGroep is equivalent aan een Lo3Categorie met 2 verschillen. Ten eerste is de inhoud een subtype van BrpGroepInhoud. Ten tweede is er geen koppeling naar een onderzoek op groep niveau.

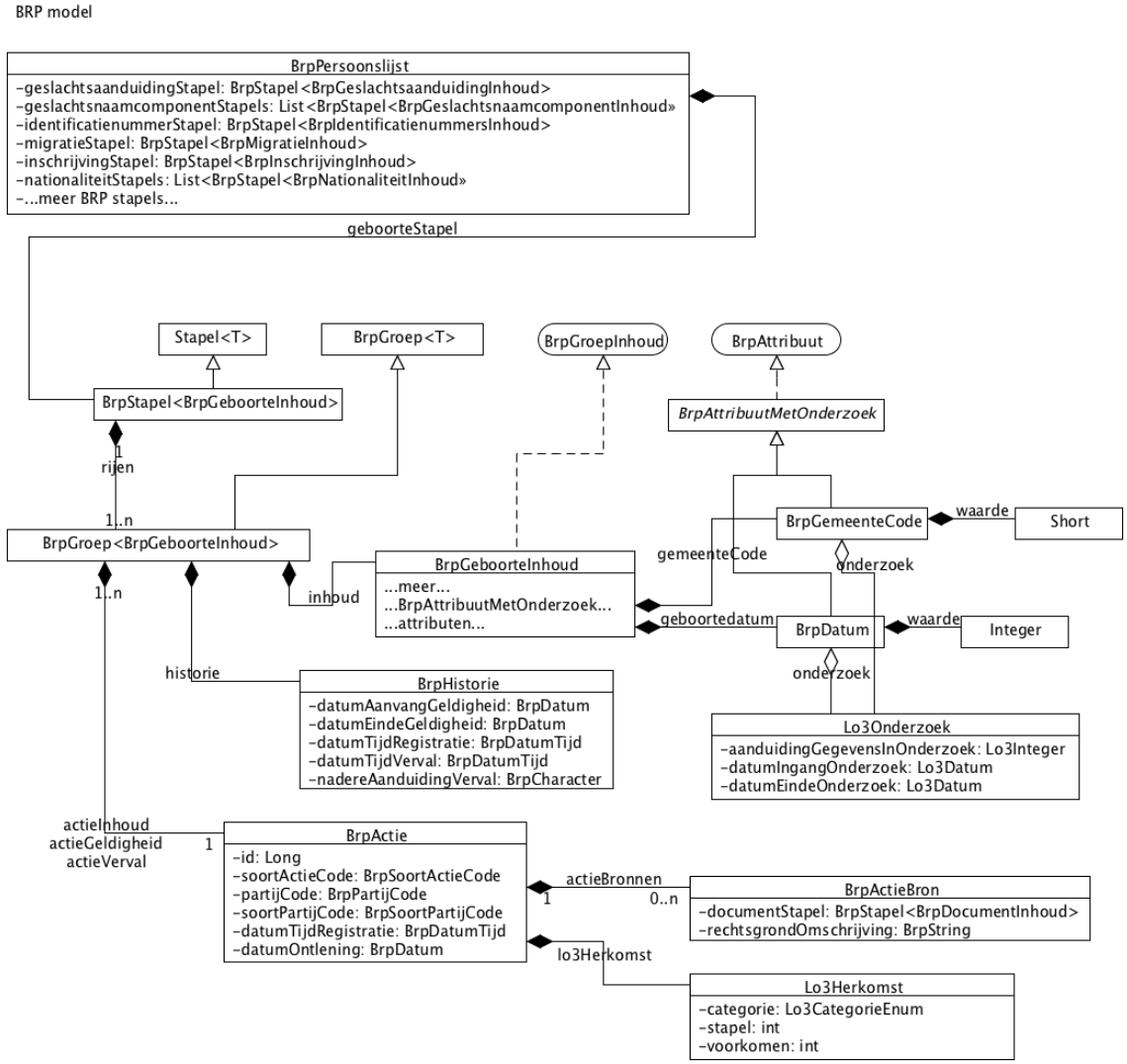
3.1.3.4 Inhoud

De inhoud van een groep bestaat uit een subtype van BrpGroepInhoud. Elke BRP groep heeft een eigen subtype. Alle attributen in een groep zijn een subtype van BrpAttribuutMetOnderzoek.

3.1.3.5 Onderzoek

Onderzoek wordt aan de BRP attributen gekoppeld op dezelfde manier als aan de LO3 Elementen.

3.1.4 BRP model



Afbeelding 4 BRP model

Het BRP model is ook een semantisch model, net als het LO3 semantisch model. In dit geval is het model echter gebaseerd op het BRP logisch model, zoals beschreven in [LO-BRP]. De structuur van de gegevens is daar beschreven in het hoofdstuk 'Gegevensstructuur', en de details over elk gegeven zijn opgenomen in de bijlage 'Gegevenswoordenboek'.

3.1.4.1 BrpPersoonslijst

Het BRP semantisch model wordt gerepresenteerd als een object van het type BrpPersoonslijst. Gegevens waarvan in het [LO-BRP] is aangegeven dat er maximaal één actuele groep op de persoonslijst kan voorkomen worden binnen het Lo3Persoonslijst object als een enkelvoudige BrpStapel opgenomen.
Gegevens waarvan in het [LO-BRP] is aangegeven dat er meer dan één actuele groep op de persoonslijst kan voorkomen worden binnen het BrpPersoonslijst object als een lijst van BrpStapels opgenomen.
Gegevens over relaties worden volgens de BRP structuur opgenomen, en de IST objecten zijn ook opgenomen. Zie [CONV] en [LO-BRP] voor meer informatie over relaties en IST

3.1.4.2 Stapels en Rijen

Waar in het LO3 model de term 'voorkomen' gebruikt wordt, wordt in het BRP model de term 'rij' gebruikt. Door verschillende termen te gebruiken wordt beter duidelijk waar we het over hebben.

Het BRP model gebruikt hetzelfde Stapel concept als het LO3 semantisch model, om alle historische Rijen en eventueel één actuele Rij over één gebeurtenis te bundelen.
Een Stapel is gedefinieerd als zijnde één actuele categorie met alle daarbij behorende historische categorieën uit de persoonslijst.
Omdat in BRP een groep zowel een begindatum als een einddatum kan hebben, is het niet noodzakelijk dat een Stapel altijd een actuele groep bevat.

3.1.4.3 Groep

De gegevens uit één rij worden opgeslagen in een BrpGroep object. Binnen dit object worden de volgende gegevens opgeslagen:

- Inhoud, een subtype van BrpGroepInhoud (omvat alle inhoudelijke gegevens uit de groep)
- BrpHistorie (omvat datum/tijd gegevens over geldigheid, registratie en verval)
- Actie inhoud
- Actie geldigheid
- Actie verval

3.1.4.4 Inhoud

De inhoud van een rij wordt vastgelegd in een subtype van BrpGroepInhoud. Elke groep in [LO-BRP] heeft een eigen subtype dat precies de attributen uit die groep bevat, uitgezonderd de historie attributen. Elk attribuut heeft een specifiek type dat de betekenis van het attribuut vastlegt, zoals BrpGemeenteCode of BrpDatum. Alle attribuut typen zijn een subtype van BrpAttribuutMetOnderzoek. Er bestaat ook een supertype BrpAttribuut (zonder onderzoek), maar deze wordt niet voor persoonslijst gegevens gebruikt.

3.1.4.5 Onderzoek

Als een attribuut bij een onderzoek betrokken is dan wordt een Lo3Onderzoek object direct aan dat attribuut gekoppeld. Dit kan ook in geval van een leeg attribuut (onderzoek naar ontbrekende gegevens). Voor een leeg attribuut wordt een instantie van het juiste attribuuttype aangemaakt dat inhoudelijk leeg is, maar wel aan het Lo3Onderzoek is gekoppeld.
Als zowel de attribuut waarde leeg is en er is geen onderzoek op het attribuut, dan wordt het attribuut veld in de inhoud zelf met **null** gevuld. Er wordt geen attribuut aangemaakt met een lege inhoud en geen onderzoek.

3.1.4.6 Historie

De historie gegevens voor een BRP groep worden opgenomen in een BrpHistorie object. Er wordt geen apart objecttype gebruikt om het verschil aan te geven tussen groepen die alleen registratie en verval kennen (formele historie) en groepen die ook aanvang en einde geldigheid kennen (materiële historie). In een BrpHistorie object kunnen beide soorten historie worden opgenomen.

3.1.4.7 Acties

In objecten van het type BrpActie wordt de documentatie (document of akte) en andere gegevens ter verantwoording van de inhoudelijke gegevens van de groep opgenomen. Er worden aparte acties aangemaakt voor de verantwoording van de inhoudelijke gegevens (actie inhoud), de einde geldigheid van de gegevens (materieel einde, actie geldigheid) en het vervallen van de gegevens (formeel einde, actie verval).

3.2 Autorisatie modellen

De drie modellen voor autorisaties zijn gebaseerd op dezelfde principes als de modellen voor persoonsgegevens. Dat wil zeggen, aan de LO3 kant wordt gebruik gemaakt van Stapels en Voorkomens met een CategorieInhoud.
In het tussenmodel en het BRP model wordt gebruik gemaakt van Stapels en Rijen met een GroepInhoud.
Enigszins afwijkend is dat er een complexere structuur dan een eenvoudige lijst van verschillende stapels in het model zit. Zie voor details de specificatie [UC102] en de broncode.

3.3 Afnemersindicatie modellen

De drie modellen voor afnemersindicaties zijn gebaseerd op dezelfde principes als de modellen voor persoonsgegevens. Dat wil zeggen, aan de LO3 kant wordt gebruik gemaakt van Stapels en Voorkomens met een CategorieInhoud.
In het tussenmodel en het BRP model wordt gebruik gemaakt van Stapels en Rijen met een GroepInhoud.
Enigszins afwijkend is dat er net als voor autorisaties een complexere structuur dan een eenvoudige lijst van verschillende stapels in het model zit. Zie voor details de specificatie [UC103] en de broncode.

4 Conversie Regels

4.1 Conversie Persoonslijst

De conversie regels voor persoonslijsten zijn de technische realisering van de specificaties in de Documentie Bidirectionele Conversie [CONV].
Bij de implementatie van de conversie regels is er bij complexe logica vaak voor gekozen om de code zo nauw mogelijk de functionele specificatie te laten volgen. Dit betekent dat de structuur van de code soms afwijkt van de normale standaarden binnen het project. Met name kunnen methodes soms langer zijn dan normaal en kunnen er complexere conditionele regels binnen één methode voorkomen dan standaard is toegestaan. Door echter dichter bij de functionele specificatie te blijven qua structuur is het gemakkelijker om aanpassingen in de specificatie over te nemen in de code.
De conversie implementatie gaat apart om met inhoudelijke conversie van gegevens en met de conversie van historie, net als dit in [CONV] apart wordt gespecificeerd.

4.1.1 Conversie Inhoud van LO3 naar BRP

De conversie van inhoudelijke gegevens gebeurt in de klasse Lo3InhoudNaarBrpConversieStap, en in de verschillende klassen die van daaruit worden aangeroepen. Deze zijn georganiseerd per LO3 categorie en implementeren de over het algemeen eenvoudige conversie van de inhoudelijke gegevens zoals aangegeven in [CONV]. Gegevens worden vaak opgesplitst in meerdere groepen en de conversie implementatie klassen maken TussenGroepen aan en voegen die toe aan de Builder voor de TussenPersoonslijst.

4.1.2 Conversie Historie van LO3 naar BRP

De conversie van historie wordt uitgevoerd de de klasse Lo3HistorieConversie. Deze klasse bevat een algemeen technisch gedeelte van de conversie. De eigenlijke historie conversie algoritmes zoals beschreven in [CONV] zijn geïmplementeerd in de klassen van Lo3HistorieConversieVariantLB21 - LB24. De code aan het einde van de klassenaam correspondeert met de code voor het algoritme in [CONV]. De specificatie van welk algoritme voor welke BRP groep wordt gebruikt is opgenomen in het Spring configuratie bestand conversie-beans.xml.
De algoritmes zijn zo dicht mogelijk bij de formulering van de specificatie in [CONV] gehouden qua structuur. Commentaar regels verwijzen naar de nummering van de stappen in [CONV].
De historie conversie is veruit het meest complexe deel van de conversie.

4.1.3 Conversie Inhoud van BRP naar LO3

De terug conversie van de inhoudelijke gegevens gebeurt niet in een aparte stap, maar wordt aangeroepen tijdens de terug conversie van historie. Dit is omdat het samenvoegen van groepen tot één categorie invloed heeft op de te maken historie.
De code voor de inhoudelijke conversie staat wel los van de historie conversie code en is te vinden in de subklassen van de klasse BrpCategorieConverteerder.
Deze code is, net als de inhoudelijke conversie van LO3 naar BRP, georganiseerd per LO3 categorie.

4.1.4 Conversie Historie van BRP naar LO3

De code voor de terug conversie van de historie zit in de klasse BrpGroepConverteerder. In tegenstelling tot de conversie van LO3 naar BRP is bij de conversie van BRP naar LO3 sprake van slechts één algoritme.
De hele terug conversie van BRP naar LO3 (historie plus inhoud) wordt aangestuurd vanuit de BrpConverteerder klasse.
De historie conversie is veruit het meest complexe deel van de conversie.

4.2 Conversie Autorisatie

De conversie regels voor autorisaties zijn de technische realisering van de specificaties in [UC102].

De conversie implementatie gaat apart om met inhoudelijke conversie van gegevens en met de conversie van historie.
De conversie van inhoudelijke gegevens gebeurt in de klasse Lo3InhoudNaarBrpConversieStap en in de verschillende klassen die van daaruit worden aangeroepen.
De conversie van historie wordt uitgevoerd door de klasse Lo3HistorieConversie. Deze klasse bevat een algemeen technisch gedeelte van de conversie. Voor autorisaties wordt specifiek gebruik gemaakt van Lo3HistorieConversieVariantLB22. De code aan het einde van de klassenaam correspondeert met de code voor het algoritme in [CONV]. De specificatie van welk algoritme voor welke BRP groep wordt gebruikt is opgenomen in het Spring configuratie bestand conversie-beans.xml.

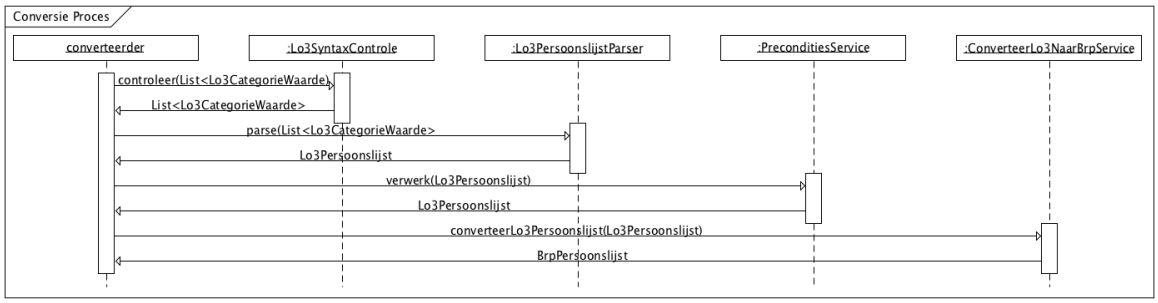
4.3 Conversie Afnemersindicatie

De conversie regels voor afnemersindicaties zijn de technische realisering van de specificaties in [UC103].
De conversie implementatie gaat apart om met inhoudelijke conversie van gegevens en met de conversie van historie.
De conversie van inhoudelijke gegevens gebeurt in de klasse Lo3InhoudNaarBrpConversieStap en in de verschillende klassen die van daaruit worden aangeroepen.
De conversie van historie wordt uitgevoerd door de klasse Lo3HistorieConversie. Deze klasse bevat een algemeen technisch gedeelte van de conversie. Voor afnemersindicaties wordt specifiek gebruik gemaakt van Lo3HistorieConversieVariantLB22. De code aan het einde van de klassenaam correspondeert met de code voor het algoritme in [CONV]. Voor de groep BrpAfnemersIndicatie wordt gebruikt gemaakt van een eigen specifiek algoritme Lo3HistorieConversieVariantLB25. Dit algoritme staat beschreven in [UC103].
De specificatie van welk algoritme voor welke BRP groep wordt gebruikt is opgenomen in het Spring configuratie bestand conversie-beans.xml.

5 Services (Persoonsgegevens, LO3 naar BRP)

Dit hoofdstuk beschrijft de services die de Conversie bibliotheek aanbiedt voor het converteren van persoonsgegevens van het LO3 model naar het BRP model. De model klassen die deze services als input nemen of als output produceren worden nader beschreven in het hoofdstuk 'Conversie Model'.

5.1 Conversie Proces



Afbeelding 5 Conversie proces

Voor het converteren van een Persoonslijst van LO3 naar BRP worden een aantal services achter elkaar aangeroepen om de verschillende stappen van de conversie te doorlopen.
Dit proces is nodig om het correct functioneren van de conversie te garanderen voor ongecontroleerde gegevens.
De eerdere stappen in het proces zorgen ervoor dat in vervolgstappen geen onverwachte foutsituaties ontstaan.
Het resultaat van de aanroep naar de Lo3SyntaxControle moet nog worden vertaald naar een Lo3Persoonslijst voordat deze kan worden doorgegeven aan de PreconditiesService. Dit gebeurt door een aanroep van de parse() methode van de Lo3PersoonslijstParser uit het project migr-bericht-model.

5.2 Logging

Logging van fouten en bijzondere situaties tijdens alle stappen van het conversie proces is een bijzonder belangrijk aspect van de conversie software. Elk verlies van of verandering aan informatie dat ontstaan als een Lo3Persoonslijst naar BRP wordt geconverteerd moet verklaarbaar zijn aan de hand van de meldingen die in de logging worden opgeslagen. Verlies of verandering van informatie wordt geconstateerd door de resulterende BrpPersoonslijst terug te converteren naar een Lo3Persoonslijst.

5.2.1 Implementatie

De logging implementatie maakt gebruik van een ThreadLocal in de Logging klasse. Zie ook ontwerpbeslissing 'Ontwerpbeslissingen'.
Hiermee heeft de aanroepende klasse echter wel de verantwoordelijkheid om de logging voor het verwerken van een persoonslijst te initialiseren.
Deze logging informatie wordt door de synchronisatie service samen met de BrpPersoonslijst in de BRP database opgeslagen.

5.2.2 Logging Initialiseren

Voor het verwerken van een persoonslijst moet de volgende methode in de klasse **Logging** worden aangeroepen:
public static void initContext()

5.2.3 Foutniveaus

De verschillende foutniveaus die kunnen worden gelogd zijn opgenomen in de enum klasse LogSeverity.

Het foutniveau SUPPRESSED is speciaal. Dit niveau wordt nooit initieel aan een melding toegekend, maar een ERROR kan omgezet worden naar SUPPRESSED als de betreffende fout volgens de opschoonregels uit de set gegevens wordt weggelaten.

5.3 Lo3SyntaxControle

Controleer een LO3 persoonslijst op syntax fouten.

5.3.1 Aanroep

Voordat de Lo3SyntaxtControle methode kan worden aangeroepen moet de Logging context zijn geïnitialiseerd. Zie hiervoor de sectie 'Logging Initiali'.

public List<Lo3CategorieWaarde> controleer(final List<Lo3CategorieWaarde> categorieWaarden) throws OngeldigePersoonslijstException

5.3.2 Invoer

Een geordende lijst met Lo3CategorieWaarde objecten. Elk object representeert een categorie voorkomen in het Lo3 datamodel.

5.3.3 Uitvoer

De uitvoer bestaat uit 2 delen, een expliciete lijst met gecontroleerde Lo3CategorieWaarde objecten, en een impliciet deel dat bestaat uit de logging die is toegevoegd aan de LoggingContext.

5.3.4 Proces

De controleer() methode loopt door alle Lo3CategorieWaarde objecten heen en controleert of deze aan een aantal syntax regels voldoen.
Er wordt gecontroleerd of elk element in de Lo3CategorieWaarde onderdeel is van een groep die is toegestaan binnen deze categorie, zoals gedefinieerd in de klasse Lo3CategorieEnum.
Voor elk element wordt gecontroleerd of de inhoud voldoet aan de type en lengte regels die zijn vastgelegd in de klasse Lo3ElementEnum.
Als er in een Lo3CategorieWaarde object fouten zitten van het niveau ERROR of erger en het object heeft de markering Onjuist, dan wordt dit object niet toegevoegd aan de uitvoer waarden en worden de betreffende fouten in niveau gereduceerd tot SUPPRESSED.

5.3.5 Foutafhandeling

Als er na het verwerken van alle Lo3CategorieWaarde objecten nog fouten over zijn van niveau ERROR of erger, dan gooit de controleer() methode een OngeldigePersoonslijstException.

5.4 PreconditiesService

Controleer een Lo3Persoonslijst op preconditie fouten. De precondities zijn gedefinieerd in [CONV].

5.4.1 Aanroep

Voordat de PreconditiesService methode kan worden aangeroepen moet de Logging context zijn geïnitialiseerd. Zie hiervoor de sectie 'Logging Initiali'.

public Lo3Persoonslijst verwerk(final Lo3Persoonslijst lo3Persoonslijst) throws OngeldigePersoonslijstException

5.4.2 Invoer

Een Lo3Persoonslijst object.

5.4.3 Uitvoer

De uitvoer bestaat uit 2 delen, een expliciet Lo3Persoonslijst object, en een impliciet deel dat bestaat uit de logging die is toegevoegd aan de LoggingContext.

5.4.4 Proces

De verwerk() methode doorloopt de hele Lo3Persoonslijst en controleert of deze aan alle precondities voldoet.
Als er in een Lo3Persoonslijst fouten zitten van het niveau ERROR of erger en die fouten kunnen worden opgelost door het weghalen van categorie voorkomens die als Onjuist zijn gemarkeerd, dan worden die onjuiste voorkomens niet toegevoegd aan de uitvoer waarde, en worden de betreffende fouten in niveau gereduceerd tot SUPPRESSED.
Als er in een stapel na het verwijderen van onjuiste voorkomens nog fouten van het niveau ERROR zitten en alle voorkomens in die stapel die juist zijn, zijn ook leeg, dan is de hele stapel semantisch leeg en wordt de lege stapel verwijdert en dus niet toegevoegd aan de uitvoer waarde. Zie paragraaf 3.2 punt 2 in [CONV].
Als er na het verwijderen van onjuiste voorkomens en lege stapels nog steeds fouten van het niveau ERROR voorkomen en de Lo3Persoonslijst is opgeschort met reden 'F', dan wordt er als uitvoer een dummy persoonslijst aangemaakt met alleen een aantal basisgegevens uit de actuele Persoon (01), Inschrijving (07) en Verblijfplaats (08) categorieën. Zie paragraaf 3.2 punt 3 in [CONV].

5.4.5 Foutafhandeling

Als er na het verwerken van de hele Lo3Persoonslijst nog fouten over zijn van niveau ERROR, dan gooit de verwerk() methode een OngeldigePersoonslijstException.

5.5 ConverteerLo3NaarBrpService

Converteer een Lo3Persoonslijst zonder syntax of preconditie fouten naar een BrpPersoonslijst.

5.5.1 Aanroep

Voordat de ConverteerLo3NaarBrpService methode kan worden aangeroepen moet de Logging context zijn geïnitialiseerd. Zie hiervoor de sectie 'Logging Initiali'.

public BrpPersoonslijst converteerLo3Persoonslijst(final Lo3Persoonslijst lo3Persoonslijst) throws InputValidationException

5.5.2 Invoer

Een Lo3Persoonslijst object dat voldoet aan alle syntax en preconditie regels.

5.5.3 Uitvoer

De uitvoer bestaat uit 2 delen, een expliciet BrpPersoonslijst object, en een impliciet deel dat bestaat uit de logging van bijzondere situaties tijdens de conversie die is toegevoegd aan de LoggingContext.

5.5.4 Proces

De converteerLo3Persoonslijst() methode transformeert de Lo3Persoonslijst in 3 stappen naar een BrpPersoonslijst.
Als eerste stap wordt er geconverteerd naar een TussenPersoonslijst, waarbij de inhoudelijke gegevens worden vertaald en opgesplitst naar BRP groepen, maar elke BRP groep behoudt de volledige LO3 historie van de categorie waarvan deze is afgeleid. Meer details over deze stap zijn te vinden in de sectie 'Conversie Inhoud' van het hoofdstuk 'Conversie Regels'.
Als tweede stap wordt de TussenPersoonslijst door de historie conversie omgezet naar een BrpPersoonslijst. Hierbij wordt per BRP groep de historie geconverteerd, en worden meerdere rijen waar mogelijk samengevoegd zodat een correcte BRP historie voor de groep ontstaat. Meer details over deze stap zijn te vinden in de sectie 'Conversie Historie' van het hoofdstuk 'Conversie Regels'.
Als laatste stap worden een aantal afgeleide gegevens toegevoegd aan de BRPPersoonslijst om deze te completeren. Dit betreft de Voornamen en Geslachtsnaam componenten.

5.5.5 Foutafhandeling

Als de persoonslijst goed gecontroleerd is voor conversie (door de syntax controle en de precondities), dan treden er geen verwachte fouten meer op tijdens de conversie. Alle onverwachte fouten zullen resulteren in een RuntimeException.

6 Services (Persoonsgegevens, BRP naar LO3)

6.1 Conversie Proces

In tegenstelling tot de conversie van LO3 naar BRP gebeurt de terug conversie vanuit BRP naar LO3 in één enkele stap middels de ConverteerBrpNaarLo3Service.
Er is geen syntax controle omdat de informatie in BRP in een gestructureerd formaat wordt opgeslagen. Dit voorkomt automatisch syntax fouten.
Er is geen aparte stap voor het controleren van precondities, dit gebeurt binnen de ConverteerBrpNaarLo3Service.
Ook wordt er geen Logging gebruikt zoals bij de conversie van LO3 naar BRP.

6.2 ConverteerBrpNaarLo3Service

Converteer een BrpPersoonslijst naar een Lo3Persoonslijst.

6.2.1 Aanroep

public Lo3Persoonslijst converteerBrpPersoonslijst(final BrpPersoonslijst brpPersoonslijst)

6.2.2 Invoer

Een BrpPersoonslijst object.

6.2.3 Uitvoer

Een Lo3Persoonslijst object.

6.2.4 Proces

De converteerBrpPersoonslijst() methode valideert de BrpPersoonslijst intern om de precondities voor de terug conversie van BRP naar LO3 te controleren. Zie ontwerpbeslissing ‘Ontwerpbeslissingen’. De precondities zijn gedefinieerd in [CONV].
De conversie zelf wordt in één stap uitgevoerd. Hierbij wordt de historie conversie uitgevoerd aan de hand van de Acties die aan de BrpPersoonslijst gekoppeld zijn. Binnen de historie conversie wordt voor individuele rijen een inhoudelijke conversie aangeroepen waarmee de inhoudelijke gegevens naar het LO3 model worden vertaald. Zie voor meer informatie hierover de secties ‘Conversie Inhoud’ en ‘Conversie Historie’ van het hoofdstuk ‘Conversie Regels’.
Er volgen dan nog een aantal relatief simpele nabewerking stappen die de Lo3Persoonslijst laten voldoen aan de inhoudelijke en sortering regels van LO3.

6.2.5 Foutafhandeling

Als er een preconditie wordt overtreden in de BrpPersoonslijst, dan leidt dit tot het gooien van een PreconditieException. PreconditieException is een subklasse van RuntimeException en wordt niet expliciet afgevangen.
Alle andere fouten tijdens conversie resulteren in een RuntimeException.

7 Services (Autoristaties en Afnemersindicaties)

7.1 Conversie Proces

De conversie van autorisaties en afnemersindicaties verloopt op soortgelijke wijze als de conversie van persoonsgegevens, maar in iets vereenvoudigde vorm.
Er is geen expliciete syntaxcontrole stap omdat de bron hiervoor gestructureerde XML is.
Er is ook geen terug conversie van BRP naar LO3. Dit is niet nodig omdat voor levering van gegevens gebruik wordt gemaakt van de BRP leveringsfunctionaliteit.
Wel wordt er gebruik gemaakt van dezelfde logging functionaliteit als bij de conversie van persoonsgegevens.

7.2 PreconditiesService (Autorisatie)

Controleer een LO3 autorisatie op preconditie fouten. De precondities zijn gedefinieerd in [UC102].

7.2.1 Aanroep

Voordat de PreconditiesService methode kan worden aangeroepen moet de Logging context zijn geïnitialiseerd. Zie hiervoor de sectie 'Logging Initiali'.

public void verwerk(final Lo3Autorisatie autorisatie) throws OngeldigeAutorisatieException

7.2.2 Invoer

Een Lo3Autorisatie object.

7.2.3 Uitvoer

De uitvoer bestaat alleen uit een impliciet deel dat bestaat uit de logging die is toegevoegd aan de LoggingContext.

7.2.4 Proces

De verwerk() methode doorloopt de Lo3Autorisatie en controleert of deze aan alle precondities voldoet.

7.2.5 Foutafhandeling

Als er in een Lo3Autorisatie fouten zitten van het niveau ERROR dan gooit de verwerk() methode een OngeldigeAutorisatieException.

7.3 PreconditiesService (Afnemersindicatie)

Controleer een LO3 afnemerindicatie op preconditie fouten. De precondities zijn gedefinieerd in [UC103].

7.3.1 Aanroep

Voordat de PreconditiesService methode kan worden aangeroepen moet de Logging context zijn geïnitialiseerd. Zie hiervoor de sectie 'Logging Initiali'.

public Lo3Afnemersindicatie verwerk(final Lo3Afnemersindicatie afnemersindicatie) throws OngeldigeAfnemersindicatieException

7.3.2 Invoer

Een Lo3Afnemersindicatie object.

7.3.3 Uitvoer

De uitvoer bestaat uit 2 delen, een expliciet Lo3Afnemersindicatie object, en een impliciet deel dat bestaat uit de logging die is toegevoegd aan de LoggingContext.

7.3.4 Proces

De verwerk() methode doorloopt de Lo3Afnemersindicatie en controleert of deze aan alle precondities voldoet.
Als er in een Lo3Afnemersindicatie fouten zitten van het niveau ERROR en deze fouten kunnen worden opgelost door het weghalen van categorie voorkomens die als Onjuist zijn gemarkeerd, dan worden deze onjuiste voorkomens niet toegevoegd aan de uitvoer waarde en worden de betreffende fouten in niveau gereduceerd tot SUPPRESSED.

7.3.5 Foutafhandeling

Als er na het verwerken van de hele Lo3Afnemersindicatie nog fouten over zijn van niveau ERROR, dan gooit de verwerk() methode een OngeldigeAfnemersindicatieException.

7.4 ConverteerLo3NaarBrpService (Autorisatie)

Converteer een LO3 autorisatie zonder preconditie fouten naar een BRP autorisatie.

7.4.1 Aanroep

Voordat de ConverteerLo3NaarBrpService methode kan worden aangeroepen moet de Logging context zijn geïnitialiseerd. Zie hiervoor de sectie 'Logging Initiali'.

public BrpAutorisatie converteerLo3Autorisatie(final Lo3Autorisatie lo3Autorisatie)

7.4.2 Invoer

Een Lo3Autorisatie object dat voldoet aan alle preconditie regels.

7.4.3 Uitvoer

De uitvoer bestaat uit 2 delen, een expliciet BrpAutorisatie object, en een impliciet deel dat bestaat uit de logging van bijzondere situaties tijdens de conversie die is toegevoegd aan de LoggingContext.

7.4.4 Proces

De converteerLo3Autorisatie() methode transformeert de Lo3Autorisatie in 2 stappen naar een BrpAutorisatie.
Als eerste stap wordt er geconverteerd naar een TussenAutorisatie, waarbij de inhoudelijke gegevens worden vertaald en opgesplitst naar BRP groepen, maar elke BRP groep behoudt de volledige LO3 historie van de categorie waarvan deze is afgeleid.
Tijdens de conversie van de inhoudelijke gegevens worden de regels zoals beschreven in [UC102] aangehouden. Hierbij worden ook de autorisatie filterrubrieken geconverteerd naar BrpExpressie objecten, terwijl de sleutelrubrieken en voorwaarderegels geconverteerd worden naar de brp expressietaal.

Als tweede stap wordt de TussenAutorisatie door de historie conversie omgezet naar een BrpAutorisatie. Hierbij wordt per BRP groep de historie geconverteerd, en worden meerdere rijen waar mogelijk samengevoegd zodat een correcte BRP historie voor de groep ontstaat.

7.4.5 Foutafhandeling

Als de autorisatie goed gecontroleerd is voor conversie (door de precondities), dan treden er geen fouten meer op tijdens de conversie. Alle onverwachte fouten zullen resulteren in een RuntimeException.

7.5 ConverteerLo3NaarBrpService (Afnemersindicatie)

Converteer een LO3 afnemersindicatie zonder preconditie fouten naar BRP afnemersindicaties.

7.5.1 Aanroep

Voordat de ConverteerLo3NaarBrpService methode kan worden aangeroepen moet de Logging context zijn geïnitialiseerd. Zie hiervoor de sectie 'Logging Initiali'.

public BrpAfnemersindicaties converteerLo3Afnemersindicaties(final Lo3Afnemersindicatie lo3Afnemersindicaties)

7.5.2 Invoer

Een Lo3Afnemersindicatie object dat voldoet aan alle preconditie regels.

7.5.3 Uitvoer

De uitvoer bestaat uit 2 delen, een expliciet BrpAfnemersindicaties object, en een impliciet deel dat bestaat uit de logging van bijzondere situaties tijdens de conversie die is toegevoegd aan de LoggingContext.

7.5.4 Proces

De converteerLo3Afnemersindicaties() methode transformeert de Lo3Afnemersindicatie in 4 stappen naar een BrpAfnemersindicaties object.
Als eerste stap worden verschillende stapels voor dezelfde afnemer samengevoegd tot één stapel.
Als tweede stap wordt er geconverteerd naar een TussenAfnemersindicaties object, waarbij de inhoudelijke gegevens worden vertaald en opgesplitst naar BRP groepen, maar elke BRP groep behoudt de volledige LO3 historie van de categorie waarvan deze is afgeleid.
Als derde stap wordt de TussenAfnemersindicaties door de historie conversie omgezet naar een BrpAfnemersindicaties object. Hierbij wordt per BRP groep de historie geconverteerd en worden meerdere rijen waar mogelijk samengevoegd zodat een correcte BRP historie voor de groep ontstaat.
Als laatste stap word een materieel einde van een afnemersindicatie omgezet naar een inhoudelijk veld datumTijdEindeVolgen.

7.5.5 Foutafhandeling

Als de afnemersindicatie goed gecontroleerd is voor conversie (door de precondities), dan treden er geen fouten meer op tijdens de conversie. Alle onverwachte fouten zullen resulteren in een RuntimeException.

7.6 Overzicht foutcodes conversie autorisaties

Bij het converteren van autorisaties van LO3 naar BRP kunnen de onderstaande foutcodes met bijbehorende betekenis optreden en in de logging voorkomen.

Foutcode	Betekenis
AUT001	Autorisaties: Afnemersindicatie is verplicht.
AUT002	Autorisaties: Ingangsdatum mag niet (gedeeltelijk) onbekend zijn.
AUT003	Autorisaties: Ingangsdatum is verplicht.
AUT004	Autorisaties: Afnemersnaam is verplicht.
AUT005	Autorisaties: Verstrekkingsbeperking is verplicht.
AUT006	Autorisaties: De einddatum mag niet voor de begindatum liggen.
AUT007	Autorisaties: Er kan geen expressie gevonden worden voor de sleutelrubriek.
AUT008	Autorisaties: Er kan geen expressie gevonden worden voor de filterrubriek.
AUT009	Autorisaties: de voorwaarde op aantekening (15.42.10) is omgezet naar een standaard waarde.
AUT010	Autorisaties: de voorwaarde op blokkering (07.66.20) is omgezet naar een standaard waarde.
AUT011	Autorisaties: Einddatum mag niet (gedeeltelijk) onbekend zijn.

7.7 Overzicht foutcodes conversie afnemersindicaties

Bij het converteren van afnemersindicaties van LO3 naar BRP kunnen de onderstaande foutcodes met bijbehorende betekenis optreden en in de logging voorkomen.

Foutcode	Betekenis
AFN001	Afnemersindicaties: a-nummer is verplicht.
AFN002	Afnemersindicaties: Er moet in minimaal 1 categorievoorkomen een

Foutcode	Betekenis
	afnemerindicatie gevuld zijn en alle gevulde afnemersindicaties in een stapel moeten gelijk zijn.
AFN003	Afnemersindicaties: Ingangsdatum mag niet (gedeeltelijk) onbekend zijn.
AFN004	Meerdere stapels gevonden met dezelfde afnemersindicatie; deze stapel wordt genegeerd.
AFN005	Stapel met meerdere verschillende afnemersindicaties.
AFN006	Historie binnen stapel is ongeldig. Actuele categorie geeft geen levering aan; volledige stapel wordt genegeerd.
AFN007	Historie binnen stapel is ongeldig. Actuele categorie geeft levering aan; enkel actuele categorie wordt geconverteerd.
AFN008	Beëindiging (lege categorie) kan niet gekoppeld worden aan een (nog niet beëindigde) start (gevulde categorie).
AFN009	Persoon niet gevonden op basis van a-nummer.
AFN010	*Partij niet gevonden op basis van afnemersindicatie.*
AFN011	Geen leveringsautorisatie gevonden voor partij.
AFN012	Afnemersindicatie kan niet worden opgeslagen.
BIJZ_CONV_AFN001	Meerdere (overlappende) stapels voor 1 afnemer bij persoonslijst. Deze stapel is genegeerd.
BIJZ_CONV_AFN002	Meerdere (overlappende) stapels voor 1 afnemer bij persoon. Deze (meest actuele) stapel is geconverteerd.
BIJZ_CONV_AFN003	Afnemersindicatie stapel gevonden zonder afnemersindicatie. Stapel wordt genegeerd.
BIJZ_CONV_AFN004	Historie binnen stapel is ongeldig. Actuele categorie geeft geen levering aan; volledige stapel wordt genegeerd.
BIJZ_CONV_AFN005	Historie binnen stapel is ongeldig. Actuele categorie geeft levering aan; enkel actuele categorie wordt geconverteerd.
BIJZ_CONV_AFN006	Meerdere (niet overlappende) stapels voor 1 afnemer bij persoon. Stapels zijn samengevoegd.
VERW_AFN001	Er kan geen abonnement worden gevonden om deze afnemersindicatie aan te koppelen.
VERW_AFN001	Er is een fout opgetreden bij het opslaan van de afnemer indicatie. Zie logbestand voor details.

8 Ontwerpbeslissingen

8.1 Scheiding BRP fysiek model t.o.v. Conversie Model

Bij het opzetten van het BRP model is bewust gekozen om dit niet direct te baseren op het fysieke gegevensmodel van de BRP zoals dat in de database wordt gebruikt. Het BRP model is daardoor niet afhankelijk van de specifieke Entiteit klassen die worden gebruikt om met de database te communiceren.
Dit heeft als voordeel dat wijzigingen die beperkt zijn tot de fysieke representatie van de gegevens, maar die geen wijziging van het logisch model van BRP betreffen, geen invloed hebben op de conversie software.

8.2 Onveranderlijkheid in modellen

Er is voor gekozen om in de semantische modellen het model onveranderlijk (immutable) te maken. Dat wil zeggen dat de veld waarden van een object in het model na het aanmaken van dat object niet meer gewijzigd kunnen worden. Een wijziging in een deel van het model kan alleen gemaakt worden door een nieuw object aan te maken, en een nieuw model op te bouwen waarbij het gewijzigde object in plaats van het oorspronkelijke object wordt opgenomen.
Dit zorgt ervoor dat gegevens uit een persoonslijst veilig gedeeld kunnen worden met andere delen van deze persoonslijst.
Bijvoorbeeld, als bij de conversie van LO3 naar BRP de inhoud van een categorie opgesplitst wordt in meerdere groepen, dan worden dezelfde historie en documentatie objecten gedeeld door meerdere tussengroepen.
Doordat de model objecten onveranderlijk zijn, is er geen risico dat latere stappen in de conversie gegevens aanpassen die gedeeld worden met andere groepen/rijen.
Onveranderlijkheid is ook een voordeel voor de thread-safety van gegevens, maar dat is hier minder van belang omdat één persoonslijst niet door meerdere threads tegelijk wordt bewerkt.

8.3 Thread-safe conversie regels

Het is de bedoeling dat de conversie services veilig vanuit meerdere threads kunnen worden aangeroepen binnen één proces.
Dit om de schaalbaarheid van de conversie software te verbeteren door niet alleen horizontale schaalbaarheid te ondersteunen, maar ook verticale schaalbaarheid.
Dit principe is in de structuur van de software verwerkt.

8.4 Conversie Onderzoek

Voor de conversie van Onderzoek gegevens is ervoor gekozen om de onderzoek gegevens mee te dragen met alle individuele elementen/attributen. Dit heeft tot gevolg dat de representatie van elementen/attributen een stuk complexer is, en ook de software die werkt met deze element/attributen, met veel meer situaties rekening moet houden.
Het afgewezen alternatief was geweest om de onderzoek gegevens apart te converteren. De onderzoek gegevens moeten echter nauw moeten aansluiten bij de inhoudelijke gegevens. Dit zou dus leiden tot een tweede implementatie van de volledige set conversie regels. Zulke duplicatie van logica is te veel extra bouwwerk, en heeft een te groot risico op verschillen tussen de twee implementaties.

8.5 Precondities voor conversie van BRP naar LO3

Voor de conversie van BRP naar LO3 is ervoor gekozen om alle preconditie fouten tot uitval van de persoonslijst te laten leiden.
Een preconditie fout betekent namelijk dat er een fout in de persoonslijst zit, en dat die niet aan alle regels van de BRP voldoet.
Omdat één van de doelstellingen van Operatie BRP is om de gegevenskwaliteit te verbeteren, is het beter om de persoonslijst te laten uitvallen, waarna actie kan worden ondernomen om de fout in de persoonslijst op te lossen.

8.6 ThreadLocal Logging

Voor logging wordt gebruik gemaakt van de ThreadLocal constructie van Java. Hiermee krijgt iedere thread een eigen kopie van het veld. Deze kopie kan worden aangesproken via de static methodes in de klasse Logging.
Deze keuze is gemaakt omdat de logging op veel verschillende plekken moet worden aangesproken. Het alternatief voor een ThreadLocal zou zijn op het logging object overal mee te geven als parameter aan alle conversie methoden. Dit zou een grote vervuiling van de code met zich meebrengen.

9 Test Hulpmiddelen

9.1 Test Projecten

Voor de conversie software zijn test hulpmiddelen ontwikkeld die zowel door ontwikkelaars als door testers gebruikt wordt. Het primaire test project is migr-test-persoon-naarbrp. Dit project wordt gebruikt voor het inlezen van LO3 persoonslijsten in excel formaat, het converteren van deze persoonslijsten naar BRP en het terug converteren van de BRP persoonslijst naar LO3. Bij alle stappen worden vergelijkingen uitgevoerd om te verifiëren dat het resultaat aan de verwachtingen voldoet, of dat het overeenkomt met de oorspronkelijke gegevens voor de conversie.
Een tweede test project is migr-test-persoon-naarlo3. Dit project voert tests uit van excel bestanden met brongegevens in BRP formaat, en converteert die naar LO3.
Het derde test project is migr-test-persoon-database. Dit project voert de conversie van LO3 naar BRP uit net zoals migr-test-persoon-naarbrp, maar doet geen terug conversie van BRP naar LO3. In plaats daarvan ondersteund dit project het uitvoeren van SQL-gebaseerde tests op de opgeslagen resultaten van een testgeval.

9.2 Gebruik

De tests in deze projecten kunnen zowel met een in-memory database gedraaid worden, als met een echte PostgreSQL database.
Beide projecten hebben een RegressieTest klasse die een volledige regressie test draait. Ook zijn en extra Test klassen die andere test sets in de projecten gebruiken. In de Test klassen wordt ingesteld of de in-memory database of een externe database gebruikt wordt.
De configuratie van de externe database is te vinden in src/test/non-packaged-resources.
De test klassen kunnen gedraaid worden als een gewone JUnit testklasse.

9.3 Effect op productie code

Voor het implementeren van deze test projecten zijn zowel de klassen van het LO3 semantisch model, als de klassen van het BRP model geannoteerd met Simple XML annotaties. Hierdoor kunnen deze modellen in- en uitgelezen worden als XML, wat wordt gebruikt om de rapportages van de test projecten te maken. Ook de verwachte resultaten worden in dit XML formaat opgeslagen.